Proceedings of the

Twenty-Third Annual Symposium of the Pattern Recognition Association of South Africa

29-30 November 2012
Pretoria, South Africa

Proceedings of the

Twenty-Third Annual Symposium of the Pattern Recognition Association of South Africa

29-30 November 2012
Pretoria, South Africa

Hosted by:
Council for Scientific and Industrial Research (CSIR)
http://www.prasa.org

Edited by: Alta de Waal
ISBN 978-0-620-54601-0
Publication date: 29 November 2012

Member of the International Association of Pattern Recognition (IAPR)

Organisation

PRASA 2012 was organised by the Council for Scientific and Industrial Research (CSIR).

Organising Committee

Alta de Waal
Asheer Bachoo
Jason de Villiers
Natasha Govender

Review process

Full-length papers accepted to PRASA have passed a strict double-blind peer review process. The submission and review procedure is as follows:

- Authors submit full camera-ready papers to the conference
- Each paper is evaluated by at least two reviewers
- The programme committee members decide whether to accept or reject the paper based on their comments
- Reviews are returned to the authors. For papers that were conditionally accepted, the authors are permitted to make changes as required
- The authors resubmit the papers for final publication.

List of reviewers:

Jason de Villiers
Natasha Govender
Asheer Bachoo
Fred Nicolls
Aby Louw
Alta de Waal
Charl van Heerden
Daniel Willet
Daniel van Niekerk
Etienne van der Poel
Etienne Barnard
Febe de Wet
Frans van den Bergh
Gerhard de Jager
Andre Nel
Hans Roos
Jaco Badehorst
Jules Tapamo
Keith Forbes
Louis Coetzee
Fintan Wilson
Marelie Davel
Neil Muller
Philip Robinson
Serestina Viriri
Helmer Strik
Herman Kamper
Stefan van der Walt
Willie Brink
Yanxia Sun
Yuko Roodt
Zenghui Wang
Zygmunt Szpak
Bernardt Duvenhage
Jaco Cronje
Christiaan van der Walt
Willem Basson
Neil Kleynhans
Thipe Modipa
Thomas Niesler
Inger Fabris-Rotelli

Table of Contents

Optimisation of acoustic models for a target accent using decision-tree state clustering

Herman Kamper and Thomas Niesler
Department of Electrical and Electronic Engineering
Stellenbosch University, South Africa
kamperh@sun.ac.za, trn@sun.ac.za

***Abstract*—In this paper we extend the decision-tree state clustering algorithm normally used to construct tied-state hidden Markov models to allow for the explicit optimisation on a particular target accent. Although the traditional algorithm guarantees overall likelihood improvements when clustering states from multiple accents, per-accent improvements are not guaranteed. We develop a tractable formulation of the targeted optimisation strategy by basing the decision-tree cluster splitting criterion on a likelihood calculated exclusively on the target accent. We find that this approach leads to deterioration compared to the traditional modelling approaches. However, when combining targeted and non-targeted approaches by linear weighting, small but consistent improvements over the traditional approaches are observed.**

I. Introduction

Accented speech is often prevalent in multilingual societies. The processing of such speech is therefore a necessary but challenging task. In previous work [1] we considered different approaches for modelling the five accents of South African English (SAE). In particular, we considered multi-accent acoustic modelling which allows selective data sharing between accents. This is achieved by including accent-based questions in the decision-tree state clustering process normally used to construct tied-state hidden Markov models (HMMs).

Although multi-accent acoustic modelling enables selective sharing, the likelihood criterion used during the decision-tree state clustering process is calculated on data from all accents. The process therefore guarantees an overall likelihood improvement, but not per-accent improvements. In some practical scenarios it might, however, be desirable to obtain the best possible acoustic model set for a particular accent. This leads to the question of whether the multi-accent decision-tree state clustering approach can be extended to optimise the likelihood on a particular target accent. Selective sharing would still be allowed across accents, but data will only be shared if it is advantageous for the target accent. In this paper we develop, evaluate and analyse such techniques.

We base our investigation on databases for the five accents of SAE identified in the literature [1], [2]. The acoustic modelling approaches developed in [1] will serve as baselines in the evaluation of the proposed targeted modelling approaches.

II. Related Research

Several studies have considered acoustic modelling of different accents of the same language. One approach is to simply train separate accent-specific models that allow no sharing between accents [3]. An alternative is to pool data from all accents considered, resulting in a single accent-independent acoustic model set [4]. Adaptation techniques in which models trained on one accent are adapted using data from another accent have also been considered [5], [6].

Recently, selective data sharing across accents through the use of appropriate decision-tree state clustering algorithms has received some attention [1], [7]. In these studies the multilingual modelling approach first proposed by Schultz and Waibel [8] was extended to apply to multiple accents of the same language. In this paper we extend the multi-accent acoustic modelling approach to allow targeted optimisation on an individual accent from the set of accents considered.

III. General Experimental Methodology

A. Training and test sets

Our experiments were based on the African Speech Technology (AST) databases [9]. These consist of annotated telephone speech recorded over fixed and mobile telephone networks and contain a mix of read and spontaneous speech. As part of the AST Project, five English accented speech databases were compiled corresponding to the five South African accents of English identified in the literature [2]: Afrikaans English (AE), Black South African English (BE), Cape Flats English (CE), White South African English (EE) and Indian South African English (IE). These databases were transcribed both phonetically, using a common IPA-based phone set consisting of 50 phones, as well as orthographically.

Each of the five databases was divided into training, development and evaluation sets. As indicated in Tables I and II, the training sets each contain between 5.5 and 7 hours of speech from approximately 250 speakers while the evaluation sets contain approximately 25 minutes from 20 speakers for each accent. The development sets were used only for the optimisation of the recognition parameters before final testing on the evaluation data. For the development and evaluation sets the ratio of male to female speakers is approximately equal and all sets contain utterances from both land-line and mobile phones. There is no speaker-overlap between any of the sets. The average length of an utterance is approximately 2 seconds.

B. General acoustic modelling procedure

Speech recognition systems were developed using the HTK tools [10]. Speech audio data was parametrised as 13 Mel-

TABLE I
TRAINING SETS FOR EACH ACCENT.

Accent	Speech (h)	No. of utterances	No. of speakers	Phone tokens
AE	7.02	11 344	276	199 336
BE	5.45	7779	193	140 331
CE	6.15	10 004	231	174 068
EE	5.95	9878	245	178 954
IE	7.21	15 073	295	218 372
Total	31.78	54 078	1240	911 061

TABLE II
EVALUATION SETS FOR EACH ACCENT.

Accent	Speech (min)	No. of utterances	No. of speakers	Phone tokens
AE	24.16	689	21	10 708
BE	25.77	745	20	11 219
CE	23.83	709	20	11 180
EE	23.96	702	18	11 304
IE	25.41	865	20	12 684
Total	123.13	3710	99	57 095

frequency cepstral coefficients (MFCCs) with their first and second order derivatives to obtain 39 dimensional observation vectors. Cepstral mean normalisation was applied on a per-utterance basis. The parametrised training sets were used to obtain three-state left-to-right single-mixture monophone HMMs with diagonal covariance matrices using embedded Baum-Welch re-estimation. These monophone models were then cloned and re-estimated to obtain initial cross-word triphone models which were subsequently subjected to decision-tree state clustering. This was followed by five iterations of re-estimation. Finally, the number of Gaussian mixtures per state was gradually increased, each increase being followed by a further five iterations of re-estimation. This yielded diagonal-covariance cross-word tied-state triphone HMMs with three states per model and eight Gaussian mixtures per state.

As part of the research presented here, several different acoustic model sets were developed following this general training procedure. For each modelling approach a different variant of the decision-tree state clustering algorithm was applied. Since decision-tree state clustering is central to this study, the standard algorithm is described briefly in Section IV. Variants of the algorithm are subsequently described in Sections V and VI.

C. Language models

Comparison of recognition performance was based on phone recognition experiments. Using the SRILM toolkit [11], backoff bigram phone language models were trained for each accent individually from the corresponding training set phone transcriptions. Absolute discounting was used for the estimation of language model probabilities [12]. The development sets were used to optimise the word insertion penalty (WIP) and language model scaling factor (LMS) used during recognition. Because optimal WIP and LMS values showed almost no variation between accents, the same WIP and LMS settings were used for all experiments.

Since the presented work considers only the effect of the acoustic models, it was assumed that during testing the accent of each utterance was known. In order to isolate acoustic modelling effects, evaluation therefore involved presenting each test utterance only to a system employing an acoustic and language model matching the accent of that utterance.

IV. DECISION-TREE STATE CLUSTERING

The standard decision-tree state clustering algorithm that is used to construct tied-state triphone HMMs (Section III-B) is reviewed in this section. The content is based on [13] and [14].

A. Overview

The clustering process begins by pooling into a single cluster the data of corresponding states from all triphones with the same basephone. This is done for all triphones observed in the training set. A set of linguistically-motivated questions is then used to split these clusters. Such questions may, for example, ask whether the left context of a particular triphone is a vowel or whether the right context is a silence. There are, in general, many such questions and each potential question results in a split which subsequently results in an increase in training set likelihood. For each cluster the optimal question (leading to the largest likelihood increase) is determined. In this way clusters are subdivided repeatedly until either the increase in likelihood or the number of observation vectors associated with a resulting cluster (the cluster occupancy count) falls below a certain predefined threshold.

The result is a phonetically-motivated binary decision-tree where the leaf nodes represent clusters of triphone HMM states which are to be tied by pooling data. This ensures that model parameters are estimated on a sufficient amount of training data. Furthermore, each state of a triphone not seen in the training set can be associated with a leaf node in the decision-trees. This allows the synthesis of triphones that are required during recognition but are not present in the training set.

B. Details of decision-tree construction

Suppose question q splits the cluster with states $\mathbb{S}$ into two clusters with states $\mathbb{S}_1(q)$ and $\mathbb{S}_2(q)$, respectively. The increase in log likelihood resulting from the split can be calculated as

$$\Delta L_q = L(\mathbb{S}_1(q)) + L(\mathbb{S}_2(q)) - L(\mathbb{S}) \tag{1}$$

where $L(\mathbb{S})$ denotes the log likelihood of the training observation vectors assigned to the states in $\mathbb{S}$. The question q^* which maximises (1) is selected as the optimal question to split the cluster. In order to compute (1), however, the calculation of the likelihood of an arbitrary cluster of states must be tractable.

Let $\mathbb{S}$ denote an arbitrary set of HMM states and let $L(\mathbb{S})$ be the log likelihood of the training observation vectors assigned to the states in $\mathbb{S}$ under the assumption that all states in $\mathbb{S}$ share a common mean $\boldsymbol{\mu}(\mathbb{S})$ and covariance matrix $\boldsymbol{\Sigma}(\mathbb{S})$. We also assume that the transition probabilities have a negligible effect on the log likelihood and can therefore be ignored [14]. The

log likelihood that the observation vectors were generated by the states in $\mathbb{S}$ can then be calculated as

$$L(\mathbb{S}) = \log \prod_{f \in \mathbb{F}} p(\mathbf{o}_f|\mathbb{S}) = \sum_{f \in \mathbb{F}} \log \left[\mathcal{N}(\mathbf{o}_f|\boldsymbol{\mu}(\mathbb{S}), \boldsymbol{\Sigma}(\mathbb{S}))\right] \quad (2)$$

where $\mathbf{o}_f$ is the observation vector associated with frame f and $\mathbb{F}$ is the set of training frames for which the observation vectors are associated with the states in $\mathbb{S}$, i.e. $\mathbb{F} = \{f : \mathbf{o}_f$ is generated by states in $\mathbb{S}\}$. The observation probability density functions (PDFs) are single-mixture Gaussian PDFs.

The direct calculation of $L(\mathbb{S})$ using (2) requires direct recourse to the observation vectors $\mathbf{o}_f$. This is computationally intractable since datasets are large and the likelihood calculation will have to be repeated several times. Fortunately it can be shown (Appendix A) that [13]:

$$L(\mathbb{S}) = -\frac{1}{2}\left\{\log[(2\pi)^n|\boldsymbol{\Sigma}(\mathbb{S})|] + n\right\} \sum_{s \in \mathbb{S}} \sum_{f \in \mathbb{F}} \gamma_s(\mathbf{o}_f) \quad (3)$$

where n is the dimensionality of the observation vectors and $\gamma_s(\mathbf{o}_f)$ is the posterior probability that the observation vector $\mathbf{o}_f$ is generated by HMM state s. The log likelihood of a cluster of states is therefore only dependent on the shared covariance matrix $\boldsymbol{\Sigma}(\mathbb{S})$ and the total state occupancy of the cluster $\sum_{s \in \mathbb{S}} \sum_{f \in \mathbb{F}} \gamma_s(\mathbf{o}_f)$. It can be shown that the former can be calculated from the means and covariance matrices of the states in the cluster [13]. The state occupancy counts are determined during the Baum-Welch re-estimation procedure which precedes clustering. Thus, $L(\mathbb{S})$ can be calculated without recourse to the observation vectors and the decision-tree construction process becomes computationally tractable.

V. Traditional Modelling Approaches

The following gives an overview of acoustic modelling approaches considered in previous work [1] and summarises relevant results. These results are the baselines for Section VI.

A. Accent-specific and accent-independent acoustic modelling

As described in Section II, *accent-specific acoustic models* are obtained by not allowing any sharing of data between accents. By growing separate decision-trees for the different accents, triphone HMM states are clustered separately. Only questions relating to phonetic context are employed, resulting in completely distinct sets of acoustic models for each accent.

In contrast, *accent-independent models* are obtained by blindly pooling accent-specific data across accents for phones with the same IPA symbol, resulting in a single accent-independent model set. A single set of decision-trees is constructed across all accents and the clustering process employs only questions relating to phonetic context, resulting in a single accent-independent set of triphone HMMs for all accents.

These two approaches were applied to the training sets of the five accents of SAE described in Section III-A. For each accent, the decision-tree likelihood improvement threshold was optimised separately on its corresponding development set. This approach was followed for all experiments presented in this paper since the purpose here is to achieve best performance on a particular target accent and not to optimise average performance over all accents, as was the case in [1].

The first two entries in Table III show the phone recognition performance measured on the evaluation sets for the accent-specific and accent-independent modelling approaches. Accent-independent models perform better than the accent-specific models for all accents except BE. The average accuracy of the accent-independent models is also better by approximately 0.76% absolute. This improvement has been calculated to be statistically significant at the 99.9% level using bootstrap confidence interval estimation at the utterance level with 10^4 bootstrap replications over all five accents [15].

B. Multi-accent acoustic modelling

The third and final acoustic modelling approach considered in [1] is similar to accent-independent modelling. Again, the state clustering process begins by pooling corresponding states from all triphones with the same basephone. However, in this case the set of decision-tree questions take into account not only the phonetic character of the left and right contexts but also the accent of the basephone. The HMM states of two triphones with the same IPA symbol but from different accents can therefore be kept separate if there is a significant acoustic difference or can be tied if there is not. We refer to such models as *multi-accent acoustic models*. Figure 1 shows an example in which the centre state of the triphone [t]-[iy]+[ng] is tied across the AE and EE accents while the first and last states are modelled separately.

The third entry in Table III indicates the performance when using multi-accent acoustic models. For AE and IE, improved performance over the first two acoustic model sets is observed. For CE and EE, deterioration is seen relative to the accent-independent models. For BE, deterioration is seen relative to the accent-specific models. Nevertheless, the multi-accent models show a very small improvement in average accuracy over the accent-independent models. This improvement is statistically significant only at the 60% level.

To obtain some indication of what happens in the decision-tree clustering process, the type of questions most frequently asked during clustering can be considered. Figure 2 analyses the decision-trees of the multi-accent acoustic models giving optimal performance on the AE development set. The figure

TABLE III
Per-accent and average (avg.) phone recognition accuracies (%) measured on the evaluation set. The different acoustic model sets are described throughout the paper.

Acoustic model set	AE	BE	CE	EE	IE	Avg.
Accent-specific	64.80	56.77	64.59	72.97	64.27	64.68
Accent-independent	65.97	55.98	66.51	74.45	64.40	65.44
Multi-accent	66.20	56.56	66.31	73.94	64.60	65.50
Targeted multi-accent	64.60	55.17	64.11	72.65	64.44	64.21
Weighted targeted	66.74	56.56	66.13	73.94	64.96	65.65
Weight w_t used above	0.51	0.5	0.53	0.5	0.54	

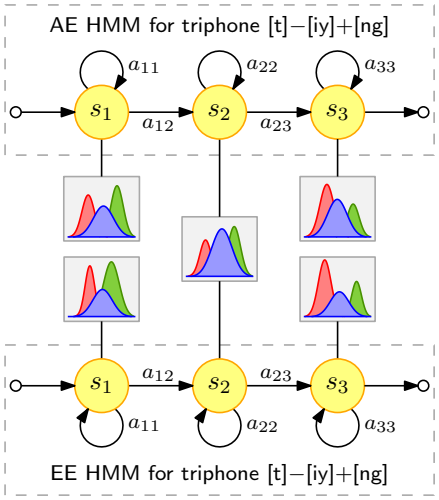


Fig. 1. Multi-accent HMMs for corresponding AE and EE triphones.

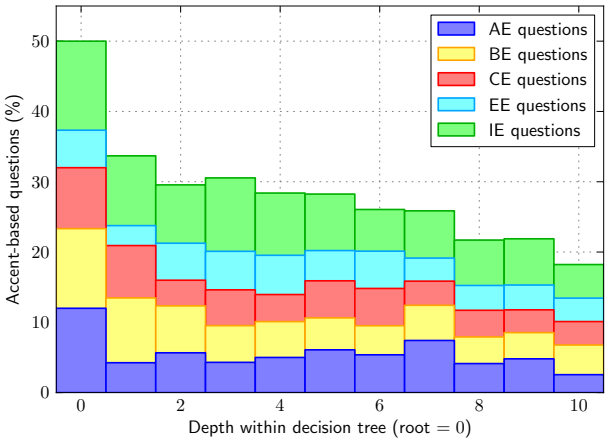


Fig. 2. The percentage of questions that relate to specific accents at various depths within the decision-trees for the multi-accent acoustic model set with optimal recognition performance on the AE development set.

shows that about 50% of all questions at the root nodes are accent-based and that this proportion drops to 34% and 30% for the roots' children and grandchildren respectively. Of the 12 970 resulting clusters (the leaf nodes) in the decision-trees, 13.2% are AE-only, 22.2% share AE with some other accent(s) and 64.7% are non-AE. These statistics and the analysis in Figure 2 are used for comparison in the next sections.

VI. Targeted Modelling Approaches

This section describes new extensions which we have made to the multi-accent acoustic modelling approach (Section V-B). We treat the results presented in Section V as baselines.

A. Motivation and overview

When clustering triphone states from several accents, the log likelihood $L(\mathbb{S})$ used as splitting criterion in the decision-tree clustering process is calculated over *all* accents. Although a particular cluster split guarantees an overall improvement in likelihood, improvements on a per-accent basis are not guaranteed. This raises the question whether the algorithm can be altered to optimise the likelihood on a particular target accent. In such an approach, a specific phonetic or accent-based question would be applied only when it is advantageous for the models of the selected target accent to do so.

B. Targeted multi-accent acoustic modelling

Suppose we have a cluster of states $\mathbb{S} = \mathbb{S}_x \cup \mathbb{S}_t$ with the states $\mathbb{S}_x$ generating observation vectors for frames $\mathbb{F}_x$ and $\mathbb{S}_t$ generating observation vectors for frames $\mathbb{F}_t$. Our aim is to optimise performance on the target states $\mathbb{S}_t$. In the traditional decision-tree state clustering procedure, the log likelihood of this cluster $\mathbb{S}$ generating the observation vectors for frames $\mathbb{F} = \mathbb{F}_x \cup \mathbb{F}_t$ would be calculated according to (3) and the optimisation criterion would be based upon this figure. We propose to determine instead the log likelihood of the target states $\mathbb{S}_t$ generating the observation vectors for frames $\mathbb{F}_t$. While all states in $\mathbb{S}$ still share a common mean $\boldsymbol{\mu}(\mathbb{S})$ and covariance matrix $\boldsymbol{\Sigma}(\mathbb{S})$, we base the cluster splitting criterion on this alternative log likelihood. By doing so, parameter estimation is still based on data from all frames $\mathbb{F} = \mathbb{F}_x \cup \mathbb{F}_t$ but the likelihood optimised is restricted to a set of target states $\mathbb{S}_t$ and no longer based on all the states $\mathbb{S}$.

The log likelihood of states $\mathbb{S}_t$ generating the associated observation vectors for frames $\mathbb{F}_t$ can be calculated as

$$\begin{aligned} L_t(\mathbb{S}) &= \log \prod_{f \in \mathbb{F}_t} p(\mathbf{o}_f|\mathbb{S}) \\ &= \sum_{f \in \mathbb{F}_t} \log \left[\mathcal{N}(\mathbf{o}_f|\boldsymbol{\mu}(\mathbb{S}), \boldsymbol{\Sigma}(\mathbb{S}))\right] \end{aligned} \tag{4}$$

This log likelihood is still dependent on all the states $\mathbb{S}$ since $\boldsymbol{\mu}(\mathbb{S})$ and $\boldsymbol{\Sigma}(\mathbb{S})$ are based on data from all the states.

As was the case in (2), the direct calculation of (4) is computationally intractable since it requires recourse to the observation vectors. However, we can again show (Appendix B) that this amended log likelihood can be calculated from the means, covariance matrices and state occupancy counts of the states in $\mathbb{S}$:

$$\begin{aligned} L_t(\mathbb{S}) = -\frac{1}{2} N_t \left\{\log[(2\pi)^n |\boldsymbol{\Sigma}(\mathbb{S})|]\right\} - \frac{1}{2} n (N_x + N_t) \\ + \frac{1}{2} \mathrm{tr}\{\boldsymbol{\Sigma}^{-1}(\mathbb{S}) N_x [\boldsymbol{\Sigma}(\mathbb{S}_x) \\ + (\boldsymbol{\mu}(\mathbb{S}_x) - \boldsymbol{\mu}(\mathbb{S}))(\boldsymbol{\mu}(\mathbb{S}_x) - \boldsymbol{\mu}(\mathbb{S}))^{\mathrm{T}}]\} \end{aligned} \tag{5}$$

with

$$N_t = \sum_{s \in \mathbb{S}_t} \sum_{f \in \mathbb{F}} \gamma_s(\mathbf{o}_f) \text{ and } N_x = \sum_{s \in \mathbb{S}_x} \sum_{f \in \mathbb{F}} \gamma_s(\mathbf{o}_f) \tag{6}$$

Since $\boldsymbol{\mu}(\mathbb{S}_x)$, $\boldsymbol{\mu}(\mathbb{S})$, $\boldsymbol{\Sigma}(\mathbb{S}_x)$ and $\boldsymbol{\Sigma}(\mathbb{S})$ are only the means and covariance matrices of the states in the corresponding clusters, the calculation of $L_t(\mathbb{S})$ as in (5) is computationally tractable.

C. Evaluation and analysis: targeted modelling

By considering each of the SAE accents in turn as the target accent, the *targeted multi-accent acoustic modelling* approach

was applied to the five training sets described in Section III-A. Phone recognition performance is shown in the fourth entry of Table III. The targeted multi-accent models are outperformed by all other models, yielding the lowest average accuracy of 64.21%. Worse performance is also achieved on a per-accent basis for all accents except for IE, for which a slight improvement over the accent-specific models is observed.

Figure 3 analyses the decision-trees of the targeted multi-accent acoustic models giving optimal performance on the AE development set. A striking feature is that the only accent-based question ever employed by the trees relate to the target accent AE. In fact, it is possible to show (Appendix C) that the target-accent-question will always be asked rather than a non-target-accent-question. Figure 3 shows that 53% of all questions at the root nodes relate to AE and that this proportion drops to 27% and 18% for the roots' children and grandchildren, respectively. Of the 5718 resulting clusters in the decision-trees, 84.7% are AE-only, 5.3% combine data from all five accents, and 10% combine data from all the accents apart from AE. This last group of clusters was consequently not used during recognition.

In comparison with the analysis of the multi-accent decision-trees in Figure 2, slightly more accent-based questions are asked at the root nodes and the proportion of accent-based questions tapers off much more quickly in the targeted case. This indicates that earlier separation of the AE accent occurs in the AE-targeted multi-accent decision-trees. Increased separation of AE is also observed when comparing the resulting cluster statistics in the targeted case to those of the non-targeted case (final paragraph, Section V-B); for the former, only 301 clusters (5.3% of 5718 clusters) share data from AE with data from any of the other accents while, for the latter, this figure is 2876 clusters (22.2% of 12 970 clusters).

Even though most clusters model AE separately, some sharing does occur in the targeted case. However, by comparing the results of the accent-specific and targeted multi-accent acoustic model sets in Table III, this small degree of sharing seems to lead to a deterioration compared to the case where accents are clustered separately from the outset.

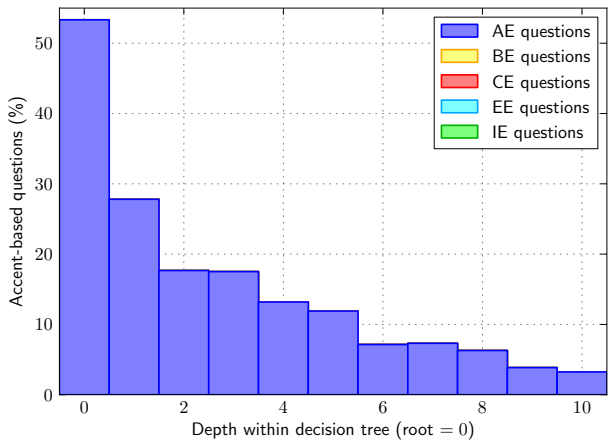


Fig. 3. The percentage of questions that relate to specific accents at various depths within the decision-trees for the targeted multi-accent acoustic model set with optimal recognition performance on the AE development set.

Although the comparative analysis presented in this section was described for the AE accent, the same trends were observed for the other four accents. Empirically we have therefore shown that the decision-trees constructed during targeted multi-accent acoustic modelling tend to model the target accent separately. However, this leads to deteriorated performance compared to simple accent-specific acoustic modelling.

D. Weighted targeted multi-accent acoustic modelling

The preceding section showed that targeted multi-accent decision-trees tend strongly towards the separation of the target accent. In this section we propose a further variant of the standard decision-tree state clustering algorithm (as applied in multi-accent modelling) in order to counteract this tendency.

Suppose again that we have a cluster of states $\mathbb{S} = \mathbb{S}_x \cup \mathbb{S}_t$ with the states $\mathbb{S}_x$ generating observation vectors for frames $\mathbb{F}_x$ and $\mathbb{S}_t$ generating observation vectors for frames $\mathbb{F}_t$. We propose that, instead of basing our cluster splitting criterion solely on the log likelihood $L_t(\mathbb{S})$ on the target states $\mathbb{S}_t$, we also assign some weight to the log likelihood $L_x(\mathbb{S})$ of the non-target states $\mathbb{S}_x$ generating the observation vectors $\mathbb{F}_x$. We calculate this alternative log likelihood as

$$L_w(\mathbb{S}) = w_t L_t(\mathbb{S}) + w_x L_x(\mathbb{S}) \tag{7}$$

with $w_t > 0$, $w_x > 0$ and $w_x = 1 - w_t$. The likelihood $L_t(\mathbb{S})$ is calculated according to (5) and, analogously, $L_x(\mathbb{S})$ is calculated as

$$\begin{aligned} L_x(\mathbb{S}) = -\frac{1}{2} N_x \{\log[(2\pi)^n |\boldsymbol{\Sigma}(\mathbb{S})|]\} - \frac{1}{2} n(N_t + N_x) \\ + \frac{1}{2} \mathrm{tr}\{\boldsymbol{\Sigma}^{-1}(\mathbb{S}) N_t [\boldsymbol{\Sigma}(\mathbb{S}_t) \\ + (\boldsymbol{\mu}(\mathbb{S}_t) - \boldsymbol{\mu}(\mathbb{S}))(\boldsymbol{\mu}(\mathbb{S}_t) - \boldsymbol{\mu}(\mathbb{S}))^{\mathrm{T}}]\} \end{aligned} \tag{8}$$

In this last equation the roles of the target and non-target states are simply reversed from the case presented in (5).

In Appendix D we show that when $w_t = w_x = 1/2$, this weighted targeted log likelihood reduces to $L_w(\mathbb{S}) = 1/2\, L(\mathbb{S})$ with $L(\mathbb{S})$ the overall log likelihood as in (2) and (3). Thus, when using equal weights, this new cluster splitting criterion is equivalent to that used for multi-accent acoustic modelling as described in Section V-B. When $w_t = 1$ and $w_x = 0$, we have $L_w(\mathbb{S}) = L_t(\mathbb{S})$, which is the unweighted targeted case presented in Section VI-B. Both multi-accent acoustic modelling and targeted multi-accent acoustic modelling are therefore special cases of this *weighted targeted multi-accent acoustic modelling* approach.

E. Evaluation and analysis: weighted targeted modelling

We again considered each of the SAE accents in turn as the target accent and applied the weighted targeted approach to the five training sets. Phone recognition performance is shown as the fifth entry in Table III. For each accent the target weight w_t was optimised on its development set. These weights are indicated in the final line of Table III.

The weighted targeted multi-accent model set achieves improved performance for AE and IE. Although multi-accent modelling is a special case of the weighted targeted approach, poorer performance might still occur since the weights are optimised on a development set. This is illustrated by the performance on CE, for instance, where accuracy deteriorates from 66.31% to 66.13%. For BE and EE, the target weight was determined to be 0.5 and the performance of the multi-accent models is therefore achieved: 56.56% and 73.94% respectively. The average performance of the weighted targeted approach is better than that achieved by any of the other approaches. The improvements in average accuracy of the weighted targeted multi-accent models (65.65%, Table III) over the accent-independent (65.44%) and multi-accent models (64.50%) are both statistically significant at the 80% level.

Figure 4 analyses the decision-trees of the weighted targeted multi-accent acoustic models giving optimal performance on the AE development set. Since the weight assigned to the target is small (0.51), the decision-trees are very similar to the non-targeted case shown in Figure 2. Of the 12 823 resulting clusters in the weighted targeted decision-trees, 13.6% are AE-only, 22.2% share AE with some other accent(s) and 64.2% are non-AE. The AE-only clusters are therefore slightly more here than in the trees analysed in Figure 2 where 13.2% of the 12 970 clusters were AE-only (final paragraph, Section V-B).

Although the improvements of the weighted targeted multi-accent acoustic modelling approach over the other approaches are relatively small, they do indicate that some gain can be obtained by targeting the decision-tree likelihood optimisation on a specific accent in this manner.

VII. Summary and Conclusions

We have described new techniques that extend the standard decision-tree state clustering algorithm used to construct tied-state hidden Markov models to allow explicit optimisation on a target accent. Using databases for the five accents of South African English, we compared these new techniques to the accent-specific, accent-independent and multi-accent acoustic modelling approaches developed in previous work.

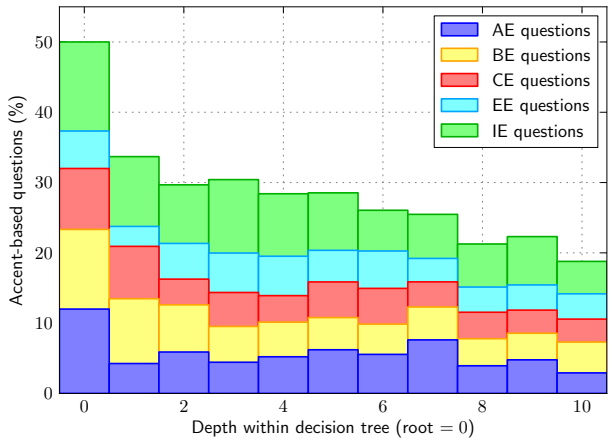


Fig. 4. The percentage of questions that relate to specific accents at various depths within the decision-trees for the weighted targeted multi-accent acoustic model set with optimal recognition performance on the AE development set ($w_t = 0.51$).

We showed that it is possible to derive expressions that allow the tractable implementation of the new clustering methods. In a first approach, the decision-tree state clustering process was altered so that the likelihood criterion used during decision-tree construction is calculated only on a target accent. Phonetic or accent-based questions are then asked only when it is advantageous for the target accent. However, both per-accent and overall average phone recognition performance indicated that this approach leads to poorer models compared to those obtained previously. Further analysis indicated that this is mostly due to the tendency of the targeted decision-trees to separate out the target accent into isolated clusters.

In order to alleviate this tendency towards separate modelling, we implemented a further extension to the algorithm in which the likelihood criterion also assigns some weight to the likelihood on non-target accents. By weighting the likelihoods on the target and non-target accents, the amount of separation could be controlled. Using this weighted targeted multi-accent modelling approach, very small average improvements ($\sim$0.2% absolute) were obtained over all other approaches.

In future work the proposed techniques should be compared to classical adaptation approaches. Clustering is also performed fairly early on in the complete acoustic model training process and is performed on the training set; changes in state-tying do not guarantee improvements for the final higher-mixture acoustic models. This warrants further investigation.

Appendix A
Log Likelihood of a Cluster of States

The log likelihood that the observation vectors were generated by the states in $\mathbb{S}$ can be calculated as

$$\begin{aligned} L(\mathbb{S}) &= \log \prod_{f \in \mathbb{F}} p(\mathbf{o}_f | \mathbb{S}) \\ &= \sum_{f \in \mathbb{F}} \log \left[\mathcal{N}(\mathbf{o}_f | \boldsymbol{\mu}(\mathbb{S}), \boldsymbol{\Sigma}(\mathbb{S})) \right] \end{aligned} \tag{A.9}$$

where the observation PDFs are assumed to be single-mixture Gaussian PDFs:

$$\mathcal{N}(\mathbf{o}_f | \boldsymbol{\mu}(\mathbb{S}), \boldsymbol{\Sigma}(\mathbb{S})) = \frac{1}{\sqrt{(2\pi)^n |\boldsymbol{\Sigma}(\mathbb{S})|}} e^{\left\{ -\frac{1}{2} (\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))^{\mathrm{T}} \boldsymbol{\Sigma}^{-1}(\mathbb{S}) (\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S})) \right\}} \tag{A.10}$$

From (A.10), equation (A.9) can then be written as

$$\begin{aligned} L(\mathbb{S}) = &-\frac{1}{2} \sum_{f \in \mathbb{F}} \log[(2\pi)^n |\boldsymbol{\Sigma}(\mathbb{S})|] \\ &- \frac{1}{2} \sum_{f \in \mathbb{F}} (\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))^{\mathrm{T}} \boldsymbol{\Sigma}^{-1}(\mathbb{S}) (\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S})) \end{aligned} \tag{A.11}$$

The covariance matrix of the cluster of states $\mathbb{S}$ can be calculated as

$$\boldsymbol{\Sigma}(\mathbb{S}) = \frac{1}{N} \sum_{f \in \mathbb{F}} (\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))(\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))^{\mathrm{T}} \tag{A.12}$$

where N is the number of frames in $\mathbb{F}$ and given by

$$N = \sum_{s\in\mathbb{S}} \sum_{f\in\mathbb{F}} \gamma_s(\mathbf{o}_f) \tag{A.13}$$

with $\gamma_s(\mathbf{o}_f)$ the posterior probability that the observation vector $\mathbf{o}_f$ is generated by HMM state s. By cross-multiplication, equation (A.12) becomes

$$N\,\mathbf{I} = \sum_{f\in\mathbb{F}} \boldsymbol{\Sigma}^{-1}(\mathbb{S})(\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))(\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))^{\mathrm{T}} \tag{A.14}$$

In [16, p. 62] the matrix identity

$$\mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{x} = \mathrm{tr}(\mathbf{A}\mathbf{x}\mathbf{x}^{\mathrm{T}}) \tag{A.15}$$

is given, where $\mathbf{x}$ is an $n\times 1$ vector, $\mathbf{A}$ is an $n\times n$ matrix and tr denotes the trace of a matrix. By taking the trace of both sides of (A.14) and then applying (A.15) we obtain

$$\begin{aligned} nN &= \mathrm{tr}\left[\sum_{f\in\mathbb{F}} \boldsymbol{\Sigma}^{-1}(\mathbb{S})(\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))(\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))^{\mathrm{T}}\right] \\ &= \sum_{f\in\mathbb{F}} \mathrm{tr}\left[\boldsymbol{\Sigma}^{-1}(\mathbb{S})(\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))(\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))^{\mathrm{T}}\right] \\ &= \sum_{f\in\mathbb{F}} (\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbb{S})(\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S})) \end{aligned} \tag{A.16}$$

where n is the dimensionality of the observation vectors.

By substituting (A.16) into (A.11) we obtain the result:

$$\begin{aligned} L(\mathbb{S}) &= -\frac{1}{2}\sum_{f\in\mathbb{F}} \log[(2\pi)^n|\boldsymbol{\Sigma}(\mathbb{S})|] - \frac{1}{2}nN \\ &= -\frac{1}{2}\left\{\log[(2\pi)^n|\boldsymbol{\Sigma}(\mathbb{S})|] + n\right\} N \\ &= -\frac{1}{2}\left\{\log[(2\pi)^n|\boldsymbol{\Sigma}(\mathbb{S})|] + n\right\} \sum_{s\in\mathbb{S}}\sum_{f\in\mathbb{F}} \gamma_s(\mathbf{o}_f) \end{aligned} \tag{A.17}$$

APPENDIX B
LOG LIKELIHOOD OF A TARGETED SUBSET OF STATES

The log likelihood of states $\mathbb{S}_t$ generating the associated observation vectors for frames $\mathbb{F}_t$ can be calculated as

$$\begin{aligned} L_t(\mathbb{S}) &= \sum_{f\in\mathbb{F}_t} \log\left[\mathcal{N}(\mathbf{o}_f|\boldsymbol{\mu}(\mathbb{S}), \boldsymbol{\Sigma}(\mathbb{S}))\right] \\ &= -\frac{1}{2}\sum_{f\in\mathbb{F}_t} \log[(2\pi)^n|\boldsymbol{\Sigma}(\mathbb{S})|] \\ &\quad - \frac{1}{2}\sum_{f\in\mathbb{F}_t} (\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbb{S})(\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S})) \\ &= -\frac{1}{2}N_t\left\{\log[(2\pi)^n|\boldsymbol{\Sigma}(\mathbb{S})|]\right\} \\ &\quad - \frac{1}{2}\sum_{f\in\mathbb{F}_t} (\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbb{S})(\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S})) \end{aligned} \tag{B.18}$$

where

$$N_t = \sum_{s\in\mathbb{S}_t}\sum_{f\in\mathbb{F}} \gamma_s(\mathbf{o}_f) \text{ and } N_x = \sum_{s\in\mathbb{S}_x}\sum_{f\in\mathbb{F}} \gamma_s(\mathbf{o}_f) \tag{B.19}$$

Calculation of the second term in (B.18) is slightly involved and we derive an expression for this term as follows.

The covariance matrix of the PDF of the cluster $\mathbb{S}$ is

$$\begin{aligned} \Sigma(\mathbb{S}) &= \frac{1}{N_x + N_t}\sum_{f\in\mathbb{F}} (\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))(\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))^{\mathrm{T}} \\ &= \frac{1}{N_x + N_t}\left[\sum_{f\in\mathbb{F}_x} (\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))(\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))^{\mathrm{T}}\right. \\ &\qquad \left. + \sum_{f\in\mathbb{F}_t} (\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))(\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))^{\mathrm{T}}\right] \end{aligned} \tag{B.20}$$

which leads to

$$\begin{aligned} \boldsymbol{\Sigma}(\mathbb{S})\,(N_x + N_t) &= \sum_{f\in\mathbb{F}_x} (\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))(\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))^{\mathrm{T}} \\ &\quad + \sum_{f\in\mathbb{F}_t} (\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))(\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))^{\mathrm{T}} \end{aligned} \tag{B.21}$$

An expression for the first term on the right hand side of (B.21) can be obtained as follows:

$$\begin{aligned} &\sum_{f\in\mathbb{F}_x} (\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))(\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))^{\mathrm{T}} \\ &= \sum_{f\in\mathbb{F}_x} ((\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}_x)) + (\boldsymbol{\mu}(\mathbb{S}_x) - \boldsymbol{\mu}(\mathbb{S})))\times \\ &\qquad ((\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}_x)) + (\boldsymbol{\mu}(\mathbb{S}_x) - \boldsymbol{\mu}(\mathbb{S})))^{\mathrm{T}} \\ &= \sum_{f\in\mathbb{F}_x} \left[(\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}_x))(\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}_x))^{\mathrm{T}}\right. \\ &\qquad + (\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}_x))(\boldsymbol{\mu}(\mathbb{S}_x) - \boldsymbol{\mu}(\mathbb{S}))^{\mathrm{T}} \\ &\qquad + (\boldsymbol{\mu}(\mathbb{S}_x) - \boldsymbol{\mu}(\mathbb{S}))(\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}_x))^{\mathrm{T}} \\ &\qquad \left. + (\boldsymbol{\mu}(\mathbb{S}_x) - \boldsymbol{\mu}(\mathbb{S}))(\boldsymbol{\mu}(\mathbb{S}_x) - \boldsymbol{\mu}(\mathbb{S}))^{\mathrm{T}}\right] \\ &= N_x\,\boldsymbol{\Sigma}(\mathbb{S}_x) + \sum_{f\in\mathbb{F}_x} (\boldsymbol{\mu}(\mathbb{S}_x) - \boldsymbol{\mu}(\mathbb{S}))(\boldsymbol{\mu}(\mathbb{S}_x) - \boldsymbol{\mu}(\mathbb{S}))^{\mathrm{T}} \\ &= N_x\left[\boldsymbol{\Sigma}(\mathbb{S}_x) + (\boldsymbol{\mu}(\mathbb{S}_x) - \boldsymbol{\mu}(\mathbb{S}))(\boldsymbol{\mu}(\mathbb{S}_x) - \boldsymbol{\mu}(\mathbb{S}))^{\mathrm{T}}\right] \end{aligned} \tag{B.22}$$

where, in the third step, we used the definitions:

$$\boldsymbol{\mu}(\mathbb{S}_x) = \frac{1}{N_x}\sum_{f\in\mathbb{F}_x} \mathbf{o}_f \tag{B.23}$$

and

$$\boldsymbol{\Sigma}(\mathbb{S}_x) = \frac{1}{N_x}\sum_{f\in\mathbb{F}_x} (\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}_x))(\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}_x))^{\mathrm{T}} \tag{B.24}$$

By substituting (B.22) into (B.21), it follows that

$$\begin{aligned} &\boldsymbol{\Sigma}(\mathbb{S})\,(N_x + N_t) = \\ &\quad N_x\left[\boldsymbol{\Sigma}(\mathbb{S}_x) + (\boldsymbol{\mu}(\mathbb{S}_x) - \boldsymbol{\mu}(\mathbb{S}))(\boldsymbol{\mu}(\mathbb{S}_x) - \boldsymbol{\mu}(\mathbb{S}))^{\mathrm{T}}\right] \\ &\quad + \sum_{f\in\mathbb{F}_t} (\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))(\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))^{\mathrm{T}} \end{aligned} \tag{B.25}$$

Multiply (B.25) with $\boldsymbol{\Sigma}^{-1}(\mathbb{S})$ and take the trace:

$$\begin{aligned} &n(N_x + N_t) = \\ &\mathrm{tr}\left\{\boldsymbol{\Sigma}^{-1}(\mathbb{S})N_x\left[\boldsymbol{\Sigma}(\mathbb{S}_x) + (\boldsymbol{\mu}(\mathbb{S}_x) - \boldsymbol{\mu}(\mathbb{S}))(\boldsymbol{\mu}(\mathbb{S}_x) - \boldsymbol{\mu}(\mathbb{S}))^{\mathrm{T}}\right]\right\} \\ &+ \sum_{f\in\mathbb{F}_t} \mathrm{tr}\left\{\boldsymbol{\Sigma}^{-1}(\mathbb{S})(\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))(\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))^{\mathrm{T}}\right\} \end{aligned} \tag{B.26}$$

and use the identity in (A.15):

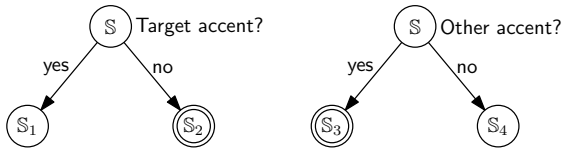
$$n(N_x + N_t) = \\ \text{tr}\left\{\boldsymbol{\Sigma}^{-1}(\mathbb{S})N_x\left[\boldsymbol{\Sigma}(\mathbb{S}_x) + (\boldsymbol{\mu}(\mathbb{S}_x) - \boldsymbol{\mu}(\mathbb{S}))(\boldsymbol{\mu}(\mathbb{S}_x) - \boldsymbol{\mu}(\mathbb{S}))^{\text{T}}\right]\right\} \\ + \sum_{f\in\mathbb{F}_t}(\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))^{\text{T}}\boldsymbol{\Sigma}^{-1}(\mathbb{S})(\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S})) \quad \text{(B.27)}$$

The last term on the right hand side of (B.27) is the required second term in (B.18). We thus obtain the result:

$$L_t(\mathbb{S}) = -\frac{1}{2}N_t\left\{\log[(2\pi)^n|\boldsymbol{\Sigma}(\mathbb{S})|]\right\} - \frac{1}{2}n(N_x + N_t) \\ + \frac{1}{2}\text{tr}\{\boldsymbol{\Sigma}^{-1}(\mathbb{S})N_x[\boldsymbol{\Sigma}(\mathbb{S}_x) \\ + (\boldsymbol{\mu}(\mathbb{S}_x) - \boldsymbol{\mu}(\mathbb{S}))(\boldsymbol{\mu}(\mathbb{S}_x) - \boldsymbol{\mu}(\mathbb{S}))^{\text{T}}]\} \quad \text{(B.28)}$$

APPENDIX C
ACCENT-BASED QUESTIONS IN TARGETED MULTI-ACCENT DECISION-TREES

Consider the two possible cluster splits illustrated in Figure 5. Assume we are using $L_t(\mathbb{S})$ as splitting criterion. In (a) the question relates to the target accent, e.g. "is the accent AE?" (assuming we optimise AE). In (b) the question relates to some non-target accent, e.g. "is the accent EE?". We show that case (a) will always occur rather than case (b).



(a) Question relates to the target accent.

(b) Question relates to some non-target accent.

Fig. 5. Two potential questions split the cluster $\mathbb{S}$. Mathematically it can be shown that (a) will always occur rather than (b).

$\mathbb{S}_2$ will contain no states from the target accent and $L_t(\mathbb{S}_2) = 0$; this cluster would therefore be a leaf node. Similarly, $\mathbb{S}_3$ will contain no states from the target accent and $L_t(\mathbb{S}_3) = 0$; again resulting in a leaf node. What distinguishes the likelihood improvement in the two cases is therefore $L_t(\mathbb{S}_1)$ and $L_t(\mathbb{S}_4)$. The former is given by

$$L_t(\mathbb{S}_1) = L_t(\mathbb{S}_t) = \sum_{f\in\mathbb{F}_t}\log\left[\mathcal{N}(\mathbf{o}_f|\boldsymbol{\mu}(\mathbb{S}_t), \boldsymbol{\Sigma}(\mathbb{S}_t))\right] \quad \text{(C.29)}$$

in accordance with (4). This log likelihood is the one maximised when performing maximum likelihood estimation of $\boldsymbol{\mu}(\mathbb{S}_t)$ and $\boldsymbol{\Sigma}(\mathbb{S}_t)$ on frames $\mathbb{F}_t$. For case (b) we have

$$L_t(\mathbb{S}_4) = \sum_{f\in\mathbb{F}_t}\log\left[\mathcal{N}(\mathbf{o}_f|\boldsymbol{\mu}(\mathbb{S}_4), \boldsymbol{\Sigma}(\mathbb{S}_4))\right] \quad \text{(C.30)}$$

with $\mathbb{S}_t \subset \mathbb{S}_4$. In this case $\boldsymbol{\mu}(\mathbb{S}_4)$ and $\boldsymbol{\Sigma}(\mathbb{S}_4)$ are obtained by maximising the log likelihood on all the frames $\mathbb{F}_4$ associated with $\mathbb{S}_4$, which is different to the calculation in (C.30) since $\mathbb{F}_t \subset \mathbb{F}_4$. It thus follows that $L_t(\mathbb{S}_4) < L_t(\mathbb{S}_1)$. The target-accent-question (a) will therefore always be asked in favour of a non-target-accent-question (b).

APPENDIX D
EQUAL WEIGHT TARGETED MODELLING

Using the form of (B.18) for both $L_t(\mathbb{S})$ and $L_x(\mathbb{S})$, we obtain the following result when $w_t = w_x = 1/2$:

$$\begin{aligned} L_w(\mathbb{S}) &= \frac{1}{2}L_t(\mathbb{S}) + \frac{1}{2}L_x(\mathbb{S}) \\ &= -\frac{1}{4}(N_x + N_t)\left\{\log[(2\pi)^n|\boldsymbol{\Sigma}(\mathbb{S})|]\right\} \\ &\quad - \frac{1}{4}\sum_{f\in\mathbb{F}}(\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S}))^{\text{T}}\boldsymbol{\Sigma}^{-1}(\mathbb{S})(\mathbf{o}_f - \boldsymbol{\mu}(\mathbb{S})) \\ &= -\frac{1}{4}(N_x + N_t)\left\{\log[(2\pi)^n|\boldsymbol{\Sigma}(\mathbb{S})|]\right\} - \frac{1}{4}n(N_x + N_t) \\ &= -\frac{1}{4}\left\{\log[(2\pi)^n|\boldsymbol{\Sigma}(\mathbb{S})|] + n\right\}N = \frac{1}{2}L(\mathbb{S}) \quad \text{(D.31)} \end{aligned}$$

where $N = N_x + N_t$ and we used (A.16) in the third line.

ACKNOWLEDGEMENTS

Parts of this work were executed using the High Performance Computer (HPC) facility at Stellenbosch University.

REFERENCES

[1] H. Kamper, F. J. Muamba Mukanya, and T. R. Niesler, "Multi-accent acoustic modelling of South African English," *Speech Communication*, vol. 54, no. 6, pp. 801–813, 2012.

[2] E. W. Schneider, K. Burridge, B. Kortmann, R. Mesthrie, and C. Upton, Eds., *A Handbook of Varieties of English*. Berlin, Germany: Mouton de Gruyter, 2004.

[3] V. Fischer, Y. Gao, and E. Janke, "Speaker-independent upfront dialect adaptation in a large vocabulary continuous speech recognizer," in *Proc. ICSLP*, Sydney, Australia, 1998, pp. 787–790.

[4] R. Chengalvarayan, "Accent-independent universal HMM-based speech recognizer for American, Australian and British English," in *Proc. Eurospeech*, Aalborg, Denmark, 2001, pp. 2733–2736.

[5] K. Kirchhoff and D. Vergyri, "Cross-dialectal data sharing for acoustic modeling in Arabic speech recognition," *Speech Commun.*, vol. 46, no. 1, pp. 37–51, 2005.

[6] J. Despres, P. Fousek, J. L. Gauvain, S. Gay, Y. Josse, L. Lamel, and A. Messaoudi, "Modeling Northern and Southern varieties of Dutch for STT," in *Proc. Interspeech*, Brighton, 2009, pp. 96–99.

[7] M. Caballero, A. Moreno, and A. Nogueiras, "Multidialectal Spanish acoustic modeling for speech recognition," *Speech Commun.*, vol. 51, pp. 217–229, 2009.

[8] T. Schultz and A. Waibel, "Language-independent and language-adaptive acoustic modeling for speech recognition," *Speech Commun.*, vol. 35, pp. 31–51, 2001.

[9] J. C. Roux, P. H. Louw, and T. R. Niesler, "The African Speech Technology project: An assessment," in *Proc. LREC*, Lisbon, Portugal, 2004, pp. 93–96.

[10] S. J. Young, G. Evermann, M. J. F. Gales, T. Hain, D. Kershaw, X. Liu, G. L. Moore, J. J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. C. Woodland, *The HTK Book (for HTK Version 3.4)*. Cambridge University Engineering Department, 2009.

[11] A. Stolcke, "SRILM – An extensible language modeling toolkit," in *Proc. ICSLP*, Denver, CO, 2002, pp. 901–904.

[12] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," *Comput. Speech Lang.*, vol. 13, pp. 359–394, 1999.

[13] S. J. Young, J. J. Odell, and P. C. Woodland, "Tree-based state tying for high accuracy acoustic modelling," in *Proc. Workshop Human Lang. Technol.*, Plainsboro, NJ, 1994, pp. 307–312.

[14] J. J. Odell, "The use of context in large vocabulary speech recognition," Ph.D. dissertation, University of Cambridge, 1995.

[15] M. Bisani and H. Ney, "Bootstrap estimates for confidence intervals in ASR performance evaluation," in *Proc. ICASSP*, Montreal, Quebec, Canada, 2004, pp. 409–412.

[16] H. A. Engelbrecht, "Automatic phoneme recognition of South African English," Master's thesis, Stellenbosch University, 2004.

Evaluating Individual mRNA Molecules Detection Techniques in Microscope Images

Rethabile Khutlang, Loretta Magagula, Musa Mhlanga
Gene Expression and Biophysics Group
Emerging Health Technologies
CSIR Biosciences
Pretoria, South Africa
rkhutlang@csir.co.za

***Abstract*—Single molecule fluorescence *in situ* hybridization followed by microscopic image analysis is one of the prominent methods used to study gene expression on a single cell level. There are various microscopic image analysis methods, leading to differing mRNA spots being detected in images for the same experiment. We present a technique to evaluate different mRNA spots detection algorithms. It is based on image annotation by expert biologists and the receiver operating characteristics. The detection methods can be compared using parameters that withstand imprecise and imbalanced environments. The proposed evaluation procedure highlighted the difference between two microscopic image analysis methods that are frequently used. It can be applied to any image analysis method that seeks to find mRNA spots on a single cell level.**

Keywords—sm-FISH; spot detection; receiver operating characteristics; F-measure

I. BACKGROUND

Gene expression is studied more and more on the single cell level [1]. One of the methods used to provide mRNA counts in individual cells is single molecule fluorescence *in situ* hybridization (sm-FSIH) followed by microscopic image analysis [2].

Single molecule FISH is a microscopy-based assay that allows for the visualization, detection and localization of specific nucleic acid sequences in their native environment. Since its origins, over 20 years ago [3], it has become a powerful molecular tool for the detection of cytogenetic and molecular genetic alterations. Applications of FISH have even extended to clinical diagnosis – chromosome analysis [4]. In a molecular setting, FISH has revealed insights in transcriptional dynamics [5, 6], mechanisms of RNA synthesis [2] and transport [7] and intracellular distribution [8,9].

The first application of fluorescent *in situ* detection involved the use of RNA probes directly labelled on the 3'end with a fluorophore to bind specific DNA sequences [10]. The labelling of probe sequences developed to use fluorophore-coupled amino-allyl modified bases [11] and the use of enzymatic incorporation of fluorophore-modified bases [12]. These advances in the technology allowed for the simple chemical production of an array of low-noise probes. Attempts to improve signal output of this assay came in the form of nick-translated, biotinylated probes, which were indirectly detected using fluorescently labelled streptavidin conjugates [13]. Currently, the standard FISH probe is produced by simple esterification chemistry to couple fluorophore to a 3'amine-modified base [14]. This method of probe preparation allows for precise and direct detection with high signal-to-noise ratios, improving the sensitivity of the assay.

Initially, RNA detection using FISH was constrained to use of large oligonucleotide probes. This was problematic as large probes could adhere to samples non-specifically resulting in false positives as well as lead to high levels in background noise. The use of reduced probe sizes lead to improved signal-to-noise-ratio and sensitivity, allowing for the single-copy detection of RNA entities and even parts of RNA [15, 16]. In this variation of the assay, 5 oligonucleotides, each about 50 oligonucleotides long, were labelled with fluorophore moieties. The hybridization of these probes to their mRNA targets yielded each target to be visualized as a diffraction-limited fluorescent "spot" [16]. However, the synthesis and purification of a small number of heavily labelled probes came with high difficulty and these probes tended to interact with each other altering hybridization characteristics which lead to severe quenching [17]. An improvement of the assay was made by using a tandem array (12-48) of reliably and singly labelled probes to accurately detect individual mRNA molecules at high spatial-temporal resolution. This advancement in the assay has lead to the simultaneous and accurate detection of multiple targets using spectrally distinct fluorophores within the same sample [18,19].

Post image acquisition, Femino et al. [16] used a constrained deconvolution algorithm to quantitatively restore out-of-focus light to its original points of origin. They could then calibrate for the fluorescent output per molecule of probe. In [19], calibration of fluorescent output per molecule of probe was not performed, however for 48 probes per mRNA they detected the same number of mRNA spots per image over a broad range of thresholds, validating the choice of a threshold parameter. Additionally, they avoided the difficulty in synthesizing and purifying heavily labelled probes.

Raj et al. [19] used the Laplacian of Gaussian filter to remove the non-uniform background and enhance particles. The resulting image conserves spatial resolution of spots, so

does the wavelet transform based filtering as used in [20]. The procedures are computationally less expensive than constrained deconvolution algorithms; so is the procedure proposed by Trcek et al. [21] – spatial band-pass filtering and local background subtraction to remove residual unevenness in the image.

There are different thresholding techniques that are applied to a filtered image to eventually find spots [16,19,20,21]. Raj et al. [19] chose a threshold from a range of thresholds for which the number of mRNAs detected varied the least. Trcek et al. [21] used Gaussian mask fitting to find the centre and intensity of each spot. In any case, the detected spots can be analysed on a per cell basis if the cell marker is used in an experiment.

We present an evaluation of individual mRNA molecules detection techniques in microscope images. The evaluation procedure is applied to two detection techniques. It is based on the use of expert biologists as the gold standard in marking spots in a microscope image. The evaluation procedure uses the receiver operating characteristics analysis (ROC) and performance evaluation metrics used in machine vision and learning.

The organisation of this paper is as follows. The next section outlines the method of evaluating detection techniques (methods used to prepare mRNAs are in supplementary data). Then detection techniques evaluation results are presented.

II. METHODS

A. Spots validation

Spots found in a z-stack image by an expert biologist constituted the gold standard used in evaluating the performance of a detection algorithm on that stack. Biologists circled all mRNA spots they could find using a custom made GUI. Hausdorff distance [22] was used to study intra- and inter-observer variability in marking spots and compare that to detection algorithms' found spots; the modified Williams index (MWI) [23] was obtained from the Hausdorff distances to further compare algorithms' spots boundaries to hand drawn ones. The index is the ratio between the average computer-observer agreement and the average observer-observer agreement. For N observations, MWI is calculated leaving one observation out at a time, for N-1 observations, resulting in N estimates.

B. Detection techniques evaluation

The posterior probability of a detected spot was calculated by finding the ratio of pixels found by both an algorithm and an expert to pixels found by an expert; minus fraction of pixels missed or over-segmented by an algorithm. Background pixels were regarded as non-target objects. The ROC curves were plotted using spots as the target class. The area under the ROC curve (AUC) is used as an evaluation value integrating the entire ROC. Sensitivity and specificity, typical two-class detection performance evaluation measures, could be established from the ROC curve at a chosen operation point.

Since the non-target class far exceeds the target class, the posfrac-recall ROC [24,25] was used to evaluate detection algorithms, as this is the imbalanced problem. The prior probability of the positive class is significantly less than that of the negative class, their ratio – skew, was used to study what fraction of non-target objects to include in the analysis. Typical imprecise environment detection evaluation measures can then be used to compare detection algorithms at one operating point: posfrac – fraction of positive detections (1),

$$posfrac = \frac{TP + FP}{N} \tag{1}$$

precision (2) – the fraction of positive detections that are actually correct and it is usually a meaningful parameter when detecting rare events because it effectively estimates an overall posterior probability [25],

$$precision = \frac{TP}{TP + FP} \tag{2}$$

recall and F-measure (3) – the geometric mean of precision and recall [25]. TP denotes the test objects labeled as target and are truly targets, while FP denotes false targets. TP_r - recall, and FP_r are calculated by normalizing TP and FP by the total number of positive and negative objects respectively, N is the sum of positive and negative objects. TP_r indicates sensitivity while 1 – FP_r denotes specificity.

$$F - measure = 2TP_r \frac{2TP_r}{TP_r + FP_r + 1} \tag{3}$$

III. RESULTS

A. Spot validation

Spot validation was studied using a set of 10 z-stack images. In each stack, the plane that showed spots the most clearly was chosen. The similarity of spots marked by the two expert biologists was studied on spots contours extracted using the custom made GUI. The comparisons in Table I were made using the Hausdorff distance. T11 and T12 represent the first expert marking spots the first and second times, more than a week apart, T2 represents the second expert. AL1 represents spots detected using the image analysis procedure outlined in [19], while AL2 represents spots found using wavelets-based detector [20].

The first expert had the highest intra-observer variability, 4.5518. There was the highest dissimilarity in the ellipses drawn around spots. The variability is further confirmed by the standard deviation of the Hausdorff distances between the first and second times the first expert marked the spots, it is the highest. The second expert still had high intra-observer variability, although it was not higher than inter-observer variability. The standard deviation of inter-observer variability is the second highest, elucidating the difference in marking spots between the two experts.

The mean Hausdorff distances between first round of spot marking by experts and automated detection procedures were lower than those between and among experts; prompting a

TABLE I. COMPARISON OF SPOTS MARKED BY TWO EXPERTS AND THOSE FOUND BY LOG PLUS THRESHOLDING AND WAVELETS BASED-METHODS

	T11& T12	T21& T22	T11& T21	T11& AL1	T21& AL1	T11& AL2	T21& AL2
Mean	4.5518	4.4353	4.5190	4.2816	4.1721	4.2768	4.1102
Std	1.5064	1.1813	1.4060	1.3628	1.0184	1.3247	1.1138

TABLE II. COMPARISON OF SPOTS MARKED BY THE TWO EXPECTS THE SECOND TIME AND THOSE FOUND BY LOG PLUS THRESHOLDING AND WAVELETS BASED- METHODS

	T12&AL1	T22&AL1	T12&AL2	T22&AL2
Mean	4.5932	4.1684	4.5353	4.1472
Std	1.0878	1.1322	1.1336	1.1209

suspicion than maybe experts marked spots differently the second time, a week later. The Hausdorff distances between both LoG-based and wavelets-based detections and experts the second time they marked spots were calculated, Table II.

Instead of experts marking spots differently the second time, Table II suggests that the first expert has higher variability in marking spots than the second expert. This is because variability between the second expert marking spots the second time and automated detections is stable when compared to that expert the first time and automated detections. This observation suggests that the first expert is the source of variability. The low Hausdorff distances between the first expert the first time and automated detections imply that though the first expect had the highest overall variability, the first expert had high variability the second time they marked spots.

Table II further shows that spot contours found by the wavelets-based method agree better with experts than those found using LoG-based method, as this was established in Table I. Fig. 1 shows typical spots marked by the first expert side by side with those detected by the two methods. If the first expert had the highest variability in marking the spots, yet visually that expert's spots marking look consistent then it can be concluded that the two experts marked spots similarly. Spots detected by automated detections visually have contours that differ from those of experts, however are acceptable as Hausdorff distances for 10 stacks are comparable to those of inter-expert.

The set of expert markings comprised four observations per object; two experts marked spots twice. The value of the MWI for the LoG based method was 1.0094; its 95% confidence interval, assuming the standard normal distribution, was (1.0070, 1.0118). The value of the MWI for the wavelets based method was 1.0172; its 95% confidence interval was (1.0148, 1.0196). The upper limit of the confidence interval for both methods is greater than one, indicating that the methods agree with the experts at least as well as the experts agree with each other.

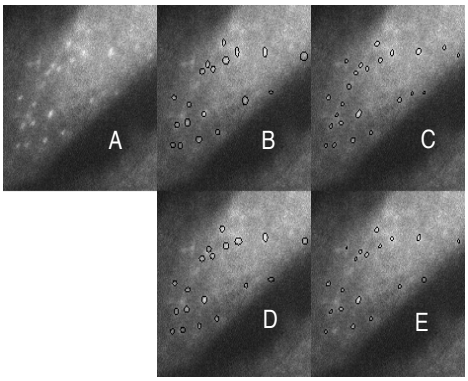


Figure 1. A shows original image, B is spots marked by the first expert the first time, C highlights those detected by the LoG-based method, D shows spots marked by the expert the second time and lastly E is spots found using wavelets-based method.

B. Detection techniques evaluation

Fig. 2 shows ROC plot for both methods using objects on a z-stack level deemed the most in focus visually. Spots marked by an expert constituted the gold standard. The AUC for the LoG-based method was 0.7751, while that of the wavelets-based method was 0.6070. The LoG-based method had a higher AUC value; over a range of posterior probabilities cut-offs it had better performance than the wavelets-based method.

For each method, at the operating point corresponding to posterior probability threshold set at 0.5, Table III shows the performance evaluated using parameters deemed suitable for imprecise environment. Sensitivity versus specificity was considered not informative enough, as the two classes were imbalanced.

Even though the LoG-based method had the highest AUC value, it is less precise than the wavelets-based method at the operating point chosen. Precision, what fraction of detected spots are actually spots should be an important measure in evaluating detection algorithms as noise frequently increases the false positive detections. The gain in precision came at the loss in sensitivity – recall. Sensitivity fell by 10% for an increase in precision of 20.50%. The wavelets-based method picks up a lot less non-spots, a quarter of those by LoG-based method, objects at the expense of missing a few true positive spots. This leads to the implication that maybe the normal ROC is not suitable for this problem; the posfrac-recall ROC could offer better performance evaluation.

TABLE III. PERFORMANCE EVALUATION OF THE LOG-BASED AND WAVELETS-BASED METHODS THE IN THE IMPRECISE ENVIRONMENT

	Precision	Recall	F-measure	Posfrac
LoG-based	0.6906	0.9600	0.6575	0.9720
Wavelets-based	0.8958	0.8600	0.6324	0.8727

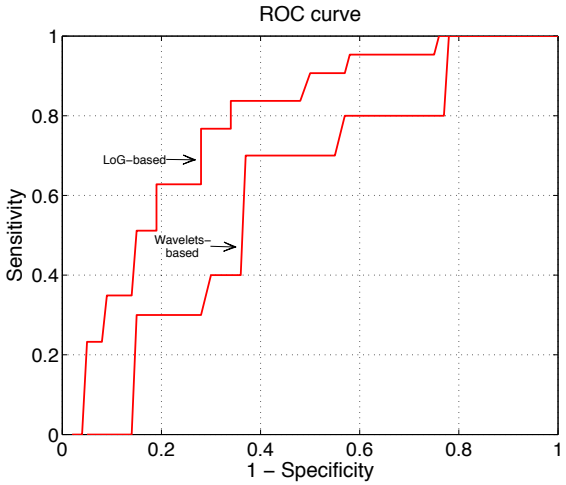


Figure 2. Example The ROC curves for the LoG-based and wavelets-based methods.

Figures 3 and 4 show posfrac-recall ROC curves for the LoG-based and wavelets-based methods respectively, for the target prior probabilities π_t= 0.5, 0.1 and 0.01. The prior probability of the non-target class was varied by varying the fraction of background pixels from the gold standard image considered as the non-target objects.

The posfrac-recall curves indicate that the two methods have similar performance with varying skew values. The choice of skew, fraction of non-target objects to include in evaluating a method, depends on the percentage of posfrac deemed acceptable in detecting spots in an application. The posfrac of both methods lowers with increasing skew for a set sensitivity. However, precision is fixed as skew varies.

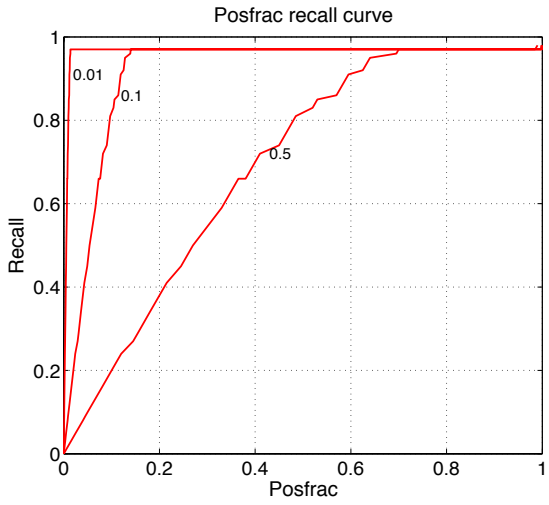


Figure 3. The posfrac-recall ROC curves for the LoG-based method.

TABLE IV. POSFRAC OF THE LOG- AND WAVELETS-BASED METHODS FOR VARYING VALUES AT 80% SENSITIVITY AND THEIR AUC VALUES

π_t	LoG		Wavelets	
	AUC	Posfrac	AUC	Posfrac
0.5	0.6844	0.4800	0.6524	0.4350
0.1	0.9129	0.0960	0.8425	0.0870
0.01	0.9643	0.0096	0.8852	0.0087

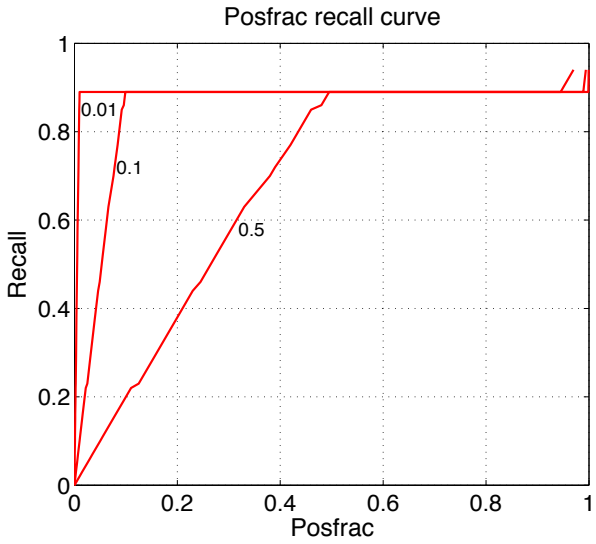


Figure 4. The posfrac-recall ROC curves for the wavelets-based method.

According to posfrac-recall curves, the LoG-based method has better overall sensitivity; but since it is less precise than the wavelets-based method its posfrac is high due to high false positives. Table 4 shows the posfrac of the two methods at 80% sensitivity with varying skew; it also shows their AUC. At 80% sensitivity, the wavelets-based method has lower posfrac for all skew values.

However, above maximum sensitivity of the wavelets-based method, its posfrac significantly surpasses that of the LoG-based method. That is confirmed by the AUC values – LoG-based method values are consistently higher than those of the wavelets-based method. The choice of the skew value and sensitivity at which to operate depends on the problem being investigated. If a method that finds all the spots, even at an expense of including background noise is desired, the high posfrac value can be ignored.

When the spot detection algorithms performance evaluation is treated as an imbalanced case problem, the posfrac-recall curves can be used to help decide at what skew and sensitivity different methods can be compared. This is appropriate because the distribution of spots to be detected is not known a priori. The methods are evaluated on a per stack basis, but the evaluations can be conducted on a batch of stacks of images. Spots can be detected in 3D or maximum projections of stacks, the evaluation metrics proposed would still hold. The

evaluation metrics can be applied to other spot detection algorithms, not just the two tested here.

IV. CONCLUSIONS

We have proposed a procedure to evaluate performance of spot detection algorithms in microscope images. The procedure depends on the marking of spots in images by an expert biologist. The marked spots form a gold standard in determining accuracy of an algorithm in imprecise and imbalanced environment. This methodology was demonstrated on two spot detection algorithms, the LoG-based and wavelets-based methods. It was able to highlight the differences in performance between the two methods. It can be applied on other spot detection algorithms, provided that they seek to find the entire diffraction-limited spot.

REFERENCES

[1] R.D. Larson, H.R. Singer, D. Zenklusen, "A single molecule view of gene expression." Imaging Cell Biology, vol. 19(11), pp. 630–637, 2009.

[2] A. Raj, S.C. Peskin, D Tranchina, D.Y. Vargas, S. Tyagi, "Stochastic mRNA Synthesis in Mammalian Cells." PLoS Biology, vol. 4(10), pp. 1-13, 2006

[3] E.F. DeLong, G.S. Wickham, N.R. Pace, "Phylogenetic stains: ribosomal RNA-based probes for the identification of single cells." Science, vol. 243(4896), pp. 1360-1363, 1989

[4] M. Guttenbach, W. Engel, M. Schmid, "Analysis of structural and numerical chromosome abnormalities in sperm of normal men and carriers of constitutional chromosome aberrations. A review." Human Genetics, vol. 100(10), pp. 1-21, 1997

[5] D.Y. Vargas, K. Shah, M. Batish, M Levandoski, S. Sinha, S.A. Marras, I.P. Schedl, S. Tyagi, "Single-molecule imaging of transcriptionally coupled and uncoupled splicing." Cell, vol. 147(5), pp. 1054-1065, 2011

[6] R.P. Jansen, M Kiebler, "Intracellular RNA sorting, transport and localization." Nat. Struc. & Mol. Bio., vol. 12, pp. 826-829,2005

[7] Y.D. Vargas, A. Raj, S.A.E. Marras, F.R. Kramer and S.Tyagi, "Mechanism of mRNA transport in the nucleus." PNAS, vol. 102(47), pp. 17008-17013, 2005

[8] F.J. Oborra, D.A. Jackson, P.R. Cook, "The path of transcripts from extra-nucleolar synthetic sites to nuclear pors: transcripts in transit are concentrated in discrete structures containing SR proteins." Journal of Cell Science, vol. 115(15), pp. 2269-2282, 1998

[9] C.S. Osborne, L. Chakalova, K.E. Brown, D. Carter, A. Horton, "Active genes dynamically colocalize to shared sites of ongoing transcription." Nature Genetics, vol. 36, pp. 1065-1071, 2004

[10] J.G Bauman, J. Wiegent, P. Borst and P. Duijn, "A new method for fluorescence microscopical localization of specific DNA sequences by in situ hybridization of fluorochrome-labelled RNA." Expe. Cell Research, vol. 128(2), pp. 485-490, 1980

[11] P.R. Langer, A.A. Waldrop and D.C. Ward, "Enzymatic synthesis of biotin-labeled polynucleotides: novel nucleic acid affinity probes." PNAS, vol. 78(11), pp. 6633-6637, 1981

[12] J. Wiegant, T. Ried, P.M. Nederlof, M. van der Ploeg, H.J. Tanke and A.K. Raap, "In situ hybridization with fluoresceinated DNA." Nucleic Acids Research, vol. 19(12), pp. 3237-3241, 1991

[13] R.C. Singer and D.C. Ward, "Actin gene expression visualized in chicken muscle tissue culture by using in situ hybridization with a biotinated nucleotide analog." PNAS, vol. 79(23), pp. 7331-7335, 1982

[14] M. Batish, A. Raj and S. Tyagi, "Single Molecule Imaging of RNA In Situ. Jeffrey E. Gerst (ed.)," RNA Detection and Visualization: Methods and Protocols. Methods in Molecular Biology, vol. 714(1), pp. 3-13,2011

[15] J.B. Lawrence, R.H. Singer, C.A. Villnaye, J.L. Stein, and G.S. Stein, "Intracellular distribution of histone mRNAs in human fibroblasts studied by in situ hybridization." PNAS vol. 85(2), pp. 463-467,1988

[16] A.M. Femino, F.S. Fay, K Fogarty and R.H Singer, "Visualization of single RNA transcripts in situ." Science, vol. 285(5363), pp. 585-590, 1998

[17] A.M. Femino, K.Fogarty, L.M. Lifshitz, W.Carrington, R.H. Singer, "Visualization of single molecule of mRNA in situ." Methods in Enzymology, vol. 361, pp.245-304, 2003

[18] J.M. Levsky, S.M. Shenoy, R.C. Pezo and R.H. Singer, "Single cell gene expression profiling." Science, vol. 297, pp. 836-840, 2002

[19] A. Raj, P. van den Bogaard, S.A. Rifkin, A. van Oudenaarden, S. Tyagi, "Imaging individual mRNA molecules using multiple singly labelled probes." Nature Methods, vol. 5, pp. 877-879, 2008

[20] J.C.Olivo-Marin, "Extraction of spots in biological images using multiscale products." Pattern Recognition, vol. 35, pp. 1989-1996, 2002

[21] T. Trek, A.J. Chao, R.D. Larson, H.Y. Park, D. Zenklusen, M.S. Shenoy, R.H. Singer, "Single-mRNA counting using fluorescent in situ hybridization in budding yeast." Nature Protocols, vol. 7, pp. 408-419, 2012

[22] D. Huttenlocher, G. Klanderman and W. Rucklidge, "Comparing images using the Hausdorff distances." IEEE Trans. Pattern Anal. Machine Intell., vol.15(9), pp. 850-863, 1993

[23] V. Chalana and Y. Kim, "A methodology for evaluation of boundary detection algorithms on medical images." IEEE Trans. Med. Imag., vol. 16(5), pp. 642-652, 1997

[24] T. Landrebe, P. Paclik, D.M.J. Tax, S. Verzakov and R.P.W Duin, "Cost-based classifier evaluation for imbalanced problems." Lecture Notes in Computer Science, vol. 3138, pp. 762-770, 2004

[25] T. Landrebe, P. Paclik, R.P.W. Duin, A.P. Bradley, "Precision-recall operating characteristic (P-ROC) curves in imprecise environments." 18th Inter. Conf. on Pattern Recognition, pp. 123-127, 2006

V. SUPPLIMENTARY DATA

Methods and Materials

The eGFP gene sequence was found on PubMed and inserted in 5'-3' direction into the probe designer algorithm on www.singlemoleculefish.com. The parameters set on the algorithm were as follows:

Number of probes	48
Probe length	20 nucleotides
GC content	45%

No of GFP probes

Lyophilized probes (Biosearch Technologies) were resuspended in 100 µl of TE (10mM Tris, 1 mM EDTA, Sigma) buffer (pH 8) to a final concentration of 100 mM each and stored at -20°C. Equal volumes of thawed probes were aliquoted (10 mM each) and pooled together for each gene to a final concentration of 480 mM for genes with 48 probes. Initially, precipitation was carried out with 10% volume of 3M Sodium Acetate (pH 5.2, Sigma) and 2.5X volume 100% cold Ethanol (Minema) according to smFISH protocol by Batish *et*

al. (2011) Probes were precipitated overnight by incubation at -20°C. Probes were then spun at 14 500 Xg, 4°C for 20 min. The pellet was then resuspended in 200 µl 0.1 M Sodium Bicarbonate (Sigma) or Sodium Tetraborate (Sigma). Approximately 0.3 mg of ATTO-565 NHS-ester dye (ATTO-TEC, Germany) was dissolved in 10 µl dimethyl sulphoxide (DMSO, Sigma). Dissolved dye solution was added to 190 µl of 0.1 M Sodium Bicarbonate (Sigma). The dye solution was added to the probe solution and incubated overnight in the dark at 37°C. Following conjugation reaction, the probes were reprecipitated at -20°C overnight as previously described. Probes were then spun at 14 500 Xg, 4°C for 20 min. Supernatant which consisted of unconjugated dye was discarded and conjugated probe pellet was rinsed twice with 70% Ethanol at 14 500 Xg, 4°C for 5 min. Supernatant was discarded and pellet was allowed to air dry. Pellet was resuspended in 200 µl of Buffer A (0.1 M Triethyl ammonium (TEA, Sigma)). Conjugated probes were separated and purified to enrich for dye-conjugated probes by reverse phase HPLC on a C18 column. Buffer A is the aqueous phase column which allows sample molecules to adhere to column and Buffer B (Triethyl ammonium and 70% (v/v) acetonitrile (Labscan) contains organic solvents in which oligonucleotides are preferentially soluble. An optimized programme of 2 to 98% Buffer B over 20 min was used to purify probes. Conjugated probes were detected at two wavelengths, 260 nm for nucleic acid and corresponding wavelength for dye used either 565 nm for ATTO-565. The appropriate fractions, containing conjugated were collected and dried in a Centri-Vac. Dried probes will were then re-precipitated overnight as previously described. Probes were then spun down with the same parameters as previously described. Probes were allowed to air dry and were re-suspended in a small volume of TE buffer (pH 8, Sigma). DNA concentrations were then determined using a Nanodrop. Probes were then diluted to a final concentration of 50ng and stored at -20°C until hybridization steps.

Cell Culture

Transfections

HeLa cells were grown in DMEM (Dulbecco's Modified Eagles's Medium, Gibco) with 10% FBS (Fetal Bovine Serum, Gibco), 2 mM L-glutamine (Sigma Aldrich) and G418. Cells were transfected with 1 µg JOMU WT and LIpofectamine 2000 (Invitrogen) complexes and 1ml Opti-MEM I Reduced Serum Medium (Gibco). Media was changed to DMEM after 4 hours and cells were incubated at 37°C and 5% CO_2 for 24 hr. Cells were passaged at 1:10 into fresh growth medium containing kanamycin sulphate (Roche). After cells had reached 90% confluency, cells were seeded in 12 well plates, each well containing an ethanol cleaned 15mm coverslip. Approximately 1 X 10^5 cells were seeded in each well in 1 ml of media. Cells were grown in a 37°C incubator with 5% CO_2 overnight. Cells were stimulated with 20ng/ml TNF-α (Tumor Necrosis Factor Alpha, Sigma Aldrich) and fixed after the following time points: 2hr, 2hr 30min and 3hr.

Cell Fixation

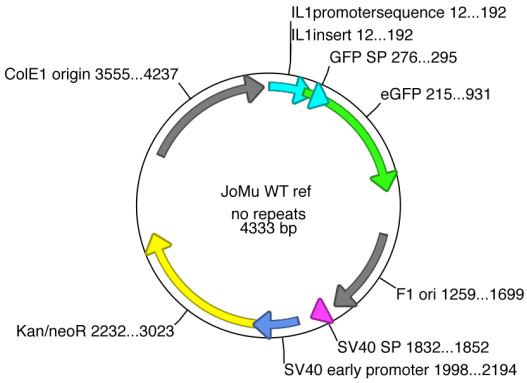
For fixation, culture medium was aspirated off wells and cells were gently washed 2X with phosphate buffered saline (PBS, Lonza). 1ml of paraformaldehyde (PFA, Sigma Adrich) was added to cells and incubated in PFA for at least 10min. PFA was aspirated off and cells were gently washed 2X with PBS. Cells were then stored in 70% Ethanol (Minema) at 4°C in parafilm sealed plates until hybridization experiments.

Probe hydrization and Imaging

Prior to hybridization, cells are gently washed 2X with PBS. A volume of 50ng of a specific conjugated probe is then added to hybridization buffer (50% (v/v) deionised formamide (CalBiochem), 10% (w/v) dextran sulphate (Sigma), 300 mM NaCl (Sigma), 20 mM NaH_2PO_4 (Sigma), 2 mM EDTA (Sigma), 10 µl vanadyl ribonucleoside complex (Sigma), 250 ug/ml E. coli tRNA (Sigma). For each coverslip, 7 µl of hybridization buffer containing 50ng of probe is used. Coverslips are then inverted, cell side down, onto 7 µl of hybridization buffer on parafilm coated glass. Hybridization was then carried out in 37°C water bath in the dark overnight. Coverslips were transferred into a 12 and 2X SSC (300 mM

NaCl, 0.3 M tri-sodium citrate, Ambion) at 37°C for 30min. Wash step was repeated three times in fresh wash buffer. Then 0.125 µg DAPI (Invitrogen) was added 20 min into the final wash step and incubated under the same conditions for 10 min. Coverslips were then gently washed 2X in PBS and incubated with equilibration buffer for 2-5min. Coverslips were then mounted onto ethanol cleaned coverslips, using glox buffer containing 3.7 X 10-3 mg/µl glucose oxidase (Sigma) and 164.38U/µl catalase (Sigma) as a mounting buffer. Cells were imaged on a Nikon widefield TIRF microscope using a 100X oil immersion objective under lamp illumination. Imaging was done using mercury lamp illumination through the appropriate filter sets at low camera gain in each of the fluorescent channels using an Andor iXion897 camera. The DAPI nuclear stain was visualized in the 405 channel at 10ms exposure time. GFP was imaged in the 488 channel with 100ms exposure time. eGFP mRNA ("spots") were imaged in the 561nm channel after 200ms exposure (imaging software, µManager).

JOMU WT Plasmid Map



Identifying suitable mathematical translation candidates from the logs of Dr. Math

Bertram Haskins
School of Information and Communication Technology
Nelson Mandela Metropolitan University
Port Elizabeth, South Africa, 6001
Email: Bertram.Haskins@nmmu.ac.za

Reinhardt A Botha
School of Information and Communication Technology
Nelson Mandela Metropolitan University
Port Elizabeth, South Africa, 6001
Email: ReinhardtA.Botha@nmmu.ac.za

***Abstract*—Dr. Math is a service, which connects high school students with math problems to volunteer human tutors. Some of the tutors on the Dr. Math service have difficulty in servicing queries received in Mxit lingo. Identifying which of these queries contain valid mathematical questions steals time which could be better spent on the actual tutoring process. This paper develops and tests filtering algorithms based on numbers, symbols and tag words, in order to identify queries containing suitable mathematical translation candidates. A combination of numeric and symbolic filtering yields the most accurate results, whereas filtering using numbers, symbols and tag words returns the highest number of results. On average, the algorithms return their filtered results in under a millisecond.**

***Index Terms*—Text analysis**

I. Introduction

i wnt u 2 hlp me wth maths abt mxid frections

Does the statement above have you scratching your head? It's an example of how student queries generally start in Dr. Math. If the initial question is so ill-phrased, imagine having to translate statements written in this shorthand to usable mathematical equations.

simp : sq rt 27 . sq rt 18 . sq rt 32 ova sq rt 12 . sq rt 8

With enough time and experience, a tutor should be able to decipher statements such as the one above. But what if they didn't have to?

A. Mxit and Dr. Math

Mxit is an on-line chat service, mainly used on mobile telephones. It provides similar functionality to other services such as Google Talk, but with added features, such as chat rooms, games and apps. Mxit has been broadly adopted by South African school learners, because of it's relatively low usage cost and it's availability on handsets from most mobile telephone manufacturers.

Mxit lingo is a general term for the non-standardized, shorthand language used by teenagers when communicating on the social platform Mxit. One of the services available on Mxit is a a math tutoring service, called Dr. Math, which has been created to take advantage of the large user base that Mxit has under South African school learners. Dr. Math currently allows over 30 000 learners to query volunteer human tutors with mathematical queries [1].

B. Problem statement and paper objective

Some tutors on the Dr. Math service have difficulty in reading the Mxit lingo statements received during tutoring sessions. Attempting to decipher the messages wastes time which could be more productively spent in tutoring other school learners. Implementing a system to automatically render qualifying queries as well-formatted mathematical equations may support the tutors, by allowing them to focus on relevant queries without first attempting to perform a translation.

This paper addresses the problem of identifying queries which are candidates for translation to mathematical equations. Thus, the objective of this paper is to devise a method with which to sift through Mxit lingo queries to determine which statements may be valid candidates for translation to a mathematical equation.

II. Methodology

To meet the paper's objective, the design science research methodology [2] has been followed. The activities of the methodology are shown in Figure 1.

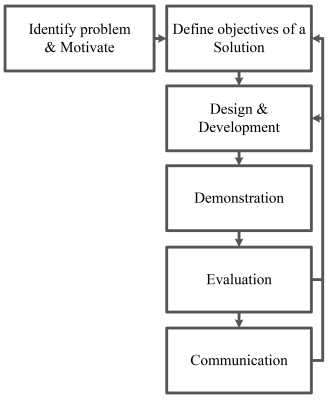


Figure 1. The design science research methodology activities [2].

In design science an artifact may be a construct, model, method or instantiation that clarifies or solves problems in implementing successful information systems [3]. This study will deliver algorithms to filter out suitable mathematical translation candidates, from the logs of Dr. Math, as its artifact.

The first two activities *identify problem* and *motivate and define objectives of a solution*, have been completed by defining the initial problem statement and research objective.

The *design and development* activity will involve the development of the algorithms (the artifact) necessary for providing the final solution to the problem. The fourth activity, *demonstration*, requires that the artifact be used to solve an instance of the problem. This will be done by means of using the artifact on a test data set, to gather experimental results. The artifact will be *evaluated* by comparing the results of testing the artifact with the initial objective of the solution

This paper forms part of the *Communication* activity, which states that the problem, its importance and its artifact must be shared with with a relevant and applicable audience. The design science research methodology requires iteration through previous activities to ensure that the artifact sufficiently solves the research problem. Previous activities will be revisited after performing initial tests using training data. The results from these tests will be used to modify the artifact to more accurately function within the problem domain, before retesting the artifact on the training set. The artifact will then be further tested on a test data set.

The Council for Scientific and Industrial Research (CSIR) Meraka Institute are responsible for the development of Dr. Math. All the processing and testing of the developed artifact will be done on the historic system logs of the Dr. Math service. The Dr. Math logs were obtained, with permission, from the CSIR.

The following sections will detail the steps followed in the development and testing of the artifact, by discussing the various filtering algorithms in turn. An overview of the steps involved in this process is shown in Figure 2.

III. Log filtering

In order for a log filtering process to be feasible, the process needs to be completely automated. Tutors deal with multiple students concurrently during tutoring sessions and may receive multiple queries from each of these learners.

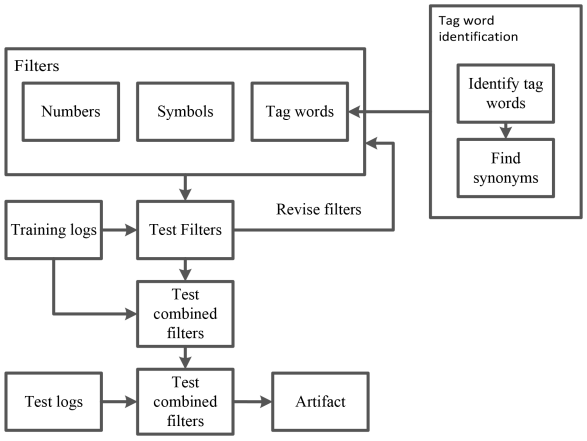


Figure 2. Steps followed in the development and testing of the artifact

Table I
TAG DESCRIPTORS

Tag	Description
N	The entry's topic is not mathematical in nature.
Y	The entry's topic is mathematical in nature, but not a suitable translation candidate.
X	The entry's topic is mathematical in nature and is a suitable translation candidate.

One of the problems with the tutoring process is finding willing, suitable tutors. Therefore, appointing extra personnel to perform filtering on logs, before passing them along to the tutors, is not feasible.

Since the automated process has to emulate the judgement of a human tutor in order select suitable translation candidates, it was necessary to create tagged versions of the logs. The Dr. Math logs represent the anonymous conversations between tutors and learners. For the purposes of this paper, the flow of a conversation is not tracked, instead the contents of individual messages, forming part of a conversation, are analyzed. To facilitate the tagging process, the logs were split into groupings of 500 entries (messages) each. Five of these groupings were randomly selected to act as a training data set and a further five were selected to act as the final testing set.

The tagging process was simplified by developing an application, which displays all 500 entries for a given group, in sequence. Each entry was then individually scrutinized (without regard for preceding or following entries) and tagged as shown in Table I.

Scrutiny of the Dr. Math logs reveal that learners phrase their mathematical queries using either numbers, symbols (non-alphanumeric characters), words or a combination of all three concepts. Therefore, the decision was made to attempt to create filters to detect suitable translation candidates, based on these three concepts. The following sections describe the processes and motivations involved in performing log filtering by numbers, symbols or tag words. The rules created and discussed in the following sections were informed by studying a sampling of South African high school mathematics text books [4], [5], [6], [7], [8].

IV. Numeric filtering

In order to filter on numbers, every log entry was checked to see whether it contained the numeric characters *0* to *9*. This was thought to be an easy indicator as to whether or not a log entry pertained to a mathematical topic and whether it would be possible to translate the query to a mathematical equation. However, upon scrutiny, the contents of the log entries prohibited this from being a viable solution for three reasons:

1) A log entry may contain the on-line pseudonym of a learner. These names sometimes contain a number, which may or may not be separated form the alphabetic portion of the name by means of a space or non-alphanumeric character.

2) Some learners discuss events in their personal lives with the tutors. These topic include their placement in an athletic event or competitions. This results in entries such as *1st* or *4th*, which could be identified as being an indicator of a mathematical topic when filtering on numbers.
3) Learners use numbers to provide typing shortcuts for certain words, e.g. *2* instead of *to* and *l8r* instead of *later*.

To decrease the number of false positives, the following set of rules was identified for the selection of translatable content by means of number filtering:

1) An entry must contain at least two instances of numeric character strings. This decision was made to ensure that there are at least 2 separate number sequences being applied to one another by means of an operator. This, however, has the detrimental effect of eliminating entries such as *5z* or *50 + a* from being viable translation candidates.
2) A numeric string may not be preceded or followed by more than 3 non-alphanumeric characters. Our study of equations in South African high school mathematics textbooks show very few entries for equations with more than 3 concurrent non-alphanumeric symbols.
3) A numeric string may not be directly preceded by an alphabetic character. Most equations in South African high school textbooks are written in the format of a constant followed by a variable, i.e. *3a* and rarely *a3*.
4) A numeric string may not be directly followed by more than 3 alphabetic characters. Our study of equations in South African high school mathematics textbooks indicate that formulas rarely contain more than three variables per grouping, i.e. *3abc* and very rarely *3abcd*.

Table II displays the results of applying the rules to the training data set. The addition of preprocessing and the introduction of other rule sets yielded no improvement on results. As such, the initial rule set was kept unchanged.

The *Manual* column indicates how many entries were identified as being suitable for translation under human scrutiny. The *Filtered* column indicates how many translation candidates the filtering process discovered, without regard for whether they are actual candidates or not. The *% Found* indicates how large a percentage of the manually identified candidates were identified by the filtering process. The last column *Accuracy %* specifies the percentage of how many of the candidates identified by the filter are valid candidates.

Table II
TRAINING RESULTS FOR NUMERIC FILTERING

Set	Manual	Filtered	% Found	Accuracy %
Training set 1	81	3	2.47	66.67
Training set 2	53	3	5.66	100.00
Training set 3	33	0	0.00	100.00
Training set 4	45	3	2.22	33.34
Training set 5	21	2	0.00	0.00
Averages:	47	2	2.07	60.00

Table III
TRAINING RESULTS FOR SYMBOL-BASED FILTERING

Set	Manual	Filtered	% Found	Accuracy %
Training set 1	81	57	61.73	87.72
Training set 2	53	26	35.85	73.08
Training set 3	33	14	21.21	50.00
Training set 4	45	42	57.78	61.90
Training set 5	21	53	57.14	22.64
Averages:	47	38	46.74	59.07

V. FILTERING ON SYMBOLS

For the purpose of filtering on symbols, a set of non-alphanumeric characters were identified. In order to qualify for selection, these characters had to fulfill certain prerequisites:

1) They had to represent a specific mathematical operator or construct as found in South African high school mathematics textbooks.
2) A school learner must be able to type the character using a mobile phone keypad.

A set of rules was then created in order to identify entries which may be candidates for translation, based on the symbols they contain. The rules are as follows:

1) An entry has to contain at least one instance of one of the identified characters.
2) An entry is disqualified if it contains more than 3 consecutive instances of these identified characters. Some of these symbolic characters may be used by students in their on-line pseudonyms. Mathematical equations in South African high school textbooks rarely contain more than three consecutive symbols.
3) An entry is disqualified if it contains more than 3 consecutive alphabetic preceding or following characters. A study of equations in a sampling of South African high school mathematics textbooks indicate that formulas rarely contain more than three variables per grouping, i.e. *+abc* or *abc+* might occur, but not *+abcd* or *abcd+*.

After scrutinizing the results of performing symbol-based filtering on the training set (Table III), it became clear that the following factors were influencing the accuracy of the results:

1) There are multiple instances in the logs where learners attempt to gather automatic responses. Auto-responses provide functionality such as an encyclopedia look-up. The auto-response commands are always in the form of a period followed by an alphabetic character or two. In some cases a period may be used to indicate multiplication and the alphabetic characters following may be interpreted as variables. The decision was made to remove all these automatic response commands, by means of preprocessing.
2) Emoticons are collections of symbols, such as *:)*, used to convey emotions. These emoticons may consist of valid symbolic characters such as ")" or "(". Even though we were aware of their existence beforehand, we did

Table IV
REPROCESSED TRAINING RESULTS FOR SYMBOL-BASED FILTERING

Set	Manual	Filtered	% Found	Accuracy %
Test set 1	81	75	81.48	88.0
Test set 2	53	46	66.04	76.09
Test set 3	33	30	66.67	73.33
Test set 4	45	47	82.22	78.72
Test set 5	21	64	80.95	26.56
Averages:	47	38	75.47	68.54

not simply want to filter all of them out, because they may not have an effect on the identification process. The decision was made to remove any emoticons, containing symbols used in mathematical equations and identified in the training set, by means of preprocessing.

3) Most of the equations identified in the training set contained the operators plus, minus, divide, multiply and the equal sign. A rule was added to give extra weighting to any entry which contained these specific symbols.

Table IV demonstrates that the *% Found* shows an increase of 28.73% and the *Accuracy %* an increase of 9.47% after the introduction of preprocessing and the additional rule.

VI. SELECTING TAG WORDS

Filtering the queries for tag words depend on two basic principles:

1) Some learners type out equations using words, instead of symbols. This may be because of preference or lack of knowledge as to which symbol, on a mobile telephone keypad, to use for concepts such as exponents or fractions. In cases such as these there may be certain words, such as plus or minus, which could be directly construed as indicators for the presence of mathematical equations.
2) In other cases however, there may be words which are not used by learners to type a mathematical equation, but they may form part of a question or word sum which may still be translated to a mathematical equation.

In order to satisfy the needs of both these principles, a study was done to identify which English words are prevalent in the South African high school mathematics curriculum.

The first phase of the study involved manually scrutinizing South African high school mathematics textbooks [4], [5], [6], [7], [8] and typing any statements which form part of either a question or an explanation of an answer into a custom application. This application enabled the identification of distinct words from these textbooks as well as a counter for how often they occur.

The second phase of the study was performed by the creation of software to identify and count instances of individual words from an input file. The South African curriculum statements for mathematics [9] and mathematics literacy [10] were used as input to this software. The curriculum statements do not contain only text, but examples of equations as well. The software captured these equations as individual words, which skewed the results.

The results from the first and second phase were combined to form a single list of words and their associated rates of occurrence.

Because this list still contained some equations, identified in phase two, a free on-line English dictionary [11] was sourced and converted to a compatible format. The words in the combined list were compared to the entries in the dictionary and any illegal entries, such as equations and incorrectly spelled words were filtered out. After the filtering process, 244 words remained of which some were plural forms. After removing these plural forms a final tally of 233 individual words remained.

If this paper focused on the selection of translation candidates from well-formed English statements to mathematical equations, these identified words may have been sufficient, but because Mxit lingo contains a non-standardized form of English, a last phase was necessary in order to identify possible synonyms for these words from the historic logs of Dr. Math. Performing this process manually would be prohibitively time-consuming, so a decision was made to automate the initial selection of synonyms and then perform a final manual selection from the results of the automated selection.

For the purpose of automatically selecting synonyms for the 233 words, custom software was written to analyze an input set of Dr. Math logs. Several techniques were then used to process these logs to attempt to identify synonyms. The following sections discuss the various techniques used.

A. Containment

The simplest method of detecting word similarity is to test wether one word contains the other. If the one word is already in its root form, then the comparison yields a low-cost and efficient means of detecting word similarity.

B. Weighting

By supplementing the containment process discussed in the previous section, with weighting, the process becomes a bit more accurate. The weighting process works on the principle that whenever one word contains another, there might be some letters left over. The less letters left, the greater the chance that the words are related. If there are no letters left over, the match would be 100%. If one word is contained by another, the initial weighting is 40%. The value of 40 % was chosen by means of trial and error. Table V shows which percentage is added to the match for certain length differences between words.

C. Stemming

There are a variety of stemming algorithms (stemmers), which are used to group words based on semantic similarity [12]. Stemmers change words by either removing pre- and suffixes or by substituting them, e.g. *engineering* is changed to *engineer*. A stemming algorithm is a computational procedure which reduces all words with the same root (or, if prefixes are

Table V
WEIGHTING AS APPLIED TO WORD LENGTH DIFFERENCE

Difference (in letters)	Added %	Total Match %
1	50	90
2	30	70
3	20	60
4	10	50
5	5	45
6	1	41

left untouched, the same stem) to a common form, usually by stripping each word of its derivational and inflectional suffixes [13].

The premise behind stemming is to get a word as close to its root English form as possible. This lessens the amount of words necessary in the normalized text base, which in turn lessens the amount of comparisons necessary to facilitate an accurate translation.

Stemming algorithms generally try to match the longest possible affix to one stored in a list. Once this is complete the algorithm will try to handle any spelling differences between root forms. One of the earliest stemming algorithms was proposed by Lovins [13]. This stemmer contains 294 endings, 29 conditions and 35 transformation rules. Various implementations of the stemmer modify these settings or add their own depending on the target language.

The stemmer chosen for this study was developed by Porter [14]. The original Porter stemmer consisted of 5 steps, each consisting of various rules, which are tested in turn. These rules may be adjusted to fit certain applications or languages, but their intention remains to determine whether two words,*W1* and *W2*, may be reduced to a common stem *S*, while retaining the meaning of their parent sentences. Many applications have taken to using the default set of rules provided by the stemming algorithm, without optimization. To this end, Porter has provided a free software implementation of his algorithm. This study makes use of a Visual C# variant of the algorithm.

The approach followed by Porter does however share some common ground with Lovins, in that they both describe a general algorithm for stemming and that they provide a specific collection of rules under which the algorithm may be applied [15].

D. Partial stemming

Partial stemming is the process of removing known prefixes or suffixes, but not in conjunction. In order for this process to work, a list of predefined prefixes and suffixes was created. A list of 38 prefixes and 37 widely used English suffixes, ranging in length from 1 to 5 letters, were compiled. Word comparisons were then done by attempting to remove these affixes from a target word, in turn, and then attempting to match the modified word to the input word.

E. Adjacency

Using adjacency to determine word similarity works on the premise that words serving a similar purpose should routinely be surrounded by the same words. An adjacency lists is generated for a word by determining how many times a word appears in conjunction with a given word in relation to all of the other words in the corpus of a given text-base or language. If an adjacency list is generated for the word *plus*, it may reflect adjacencies to words such as *five*, *ten* and *twenty*. If an adjacency list is generated for the word *add*, it is to be expected that it may occur adjacent to these same words at some stage. By comparing how many words out of the total corpus of words in the language both *plus* and *add* are adjacent to, we arrive at a matchable percentage. The higher this percentage, the greater the chance that a word may be considered a synonym for another.

F. Word dilution

Word dilution refers to the process of replacing all double occurrences of a letter in a word with a single occurrence of the letter, i.e. *address* gets converted to *adres*. This is a lightweight means of identifying words which might have been misspelled in a given sentence. In this study, when a word has been converted in this way, it is said to be in its base form. This should not be confused with the root form of a word, which is the result of applying a stemming algorithm to a word.

The author has not been able to find any precedence for this procedure, by means of literature study, but has been using these rules for synonym / substitute matching for a few years. This process is incorporated into a proprietary piece of software, called EasyMark [16], which is used for the automatic marking of student scripts.

Depending on how misspelled the word is, the dilution may need to be taken a step further, by removing the vowels from the interior of the word, as well. In such an example *address* would get converted to *adrs*. Vowels at the start or end of the word are usually not removed, as they tend to be placed correctly. This results from people typing words as they say and hear them. The beginnings and endings of words generally tend to be pronounced very distinctively, so most spelling errors are usually made in the middle of words. Medial letters of a word have more neighbours than letters at the periphery of a word, so they are more prone to being misspelled [17].

G. N-grams

Any word can be divided into smaller chunks. The smallest possible chunks being single letters. Inherently humans divide words into syllables, but a computing algorithm would not need to divide a word using these same principles. N-grams are a means of dividing a word into smaller overlapping chunks. These individual chunks could then be compared to the chunks of another word to determine similarity.

The letter *n* in the word n-gram refers to the variability in the length of the individual word chunks. Different applications may use different lengths of n-grams to different effects. Some approaches even combine several different lengths simultaneously or append blanks to the beginning and ending of a word. This helps with matching beginning-of-word and ending-of-word situations [18].

Table VI
TRAINING RESULTS FOR TAG WORD-BASED FILTERING

Set	Manual	Filtered	% Found	Accuracy %
Training set 1	81	9	7.41	66.67
Training set 2	53	4	5.66	75.00
Training set 3	33	19	27.27	47.37
Training set 4	45	18	11.11	27.78
Training set 5	21	6	14.29	50.00
Averages:	47	2	13.15	53.36

According to [19], character n-gram tokenization is an attractive alternative to stemming. Some of the n-grams derived from a word will span only portions of the word which do not show any differentiation from the word's root form. This means that many of the benefits of stemming can be achieved without any knowledge of the target language. This study employs uni-, bi- and trigrams.

VII. FILTERING ON TAG WORDS

The techniques discussed in the previous section were all employed on every one of the 233 identified tag words. An average percentage match was generated for each word with regards to each of the words contained in the input logs. The top 50 automated synonym matches for each word was then stored in a separate file. These 233 files (each containing 50 entries) were then manually scrutinized to determine which of the identified words may be seen as synonyms if encountered by a human.

These synonyms included plurals, words sharing a common root, misspelled words and completely different words. Finally, these words were compiled into a single list and filtered for duplicates, resulting in a final total of 1775 tag words.

The following set of rules were compiled in order to facilitate the process of selecting translation candidates by filtering on tag words.

1) A query must contain at least two of the tag words in order to be considered a candidate for translation. Some of the tag words may appear in normal conversation, but in a different context. Thus the decision was made to have at least two of these words in a query, to attempt to rule out general conversational usage.
2) Extra weight is added to a query if it contains predefined word pairs, which may be indicators of alternate means of specifying mathematical operations. A list of these words were sourced from [20].

Table VI show the results of performing filtering on the training set using tag word filtering. The addition of preprocessing and testing various other rule sets did not yield any improvements to the technique. As such, the initial rule set was kept unchanged.

VIII. COMPARING AND COMBINING

Having selected and implemented the three different filtering techniques, we decided to compare which of the techniques

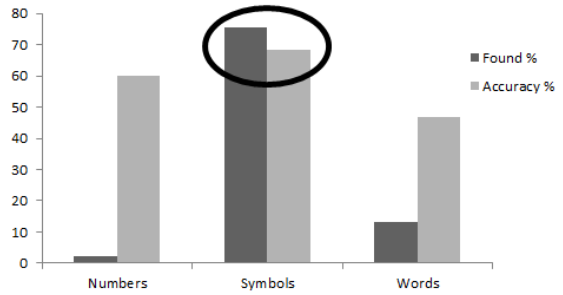


Figure 3. Comparison of the three filtering techniques

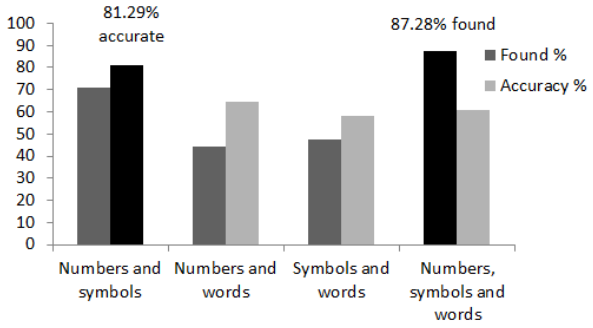


Figure 4. Results obtained from combining the filtering techniques

yielded the best results. This comparison is shown in Figure 3.

The results show a clear difference in both the number of results found and accuracy of the three filtering techniques. Filtering on symbols yield the most results, as well as the most accurate results. A problem with comparing the three techniques directly are that they should find completely different candidates, except if by chance a query contains elements to trigger for more than one of the techniques.

Most queries from learners contain some combination of numbers, symbols and words. In order to test whether combining the filters yielded different results, the training set was reprocessed using various combinations of the three filters. The combination tests also measured the average length of time processing takes for each query, to ensure that the processes are feasible in real-time. Even combining all three filters only yielded an average processing time of 0.84 milliseconds. This processing time will vary, depending on which hardware platform the processing is performed on, but serves to indicate that the filtering algorithms provide results in real-time.

As Figure 4 illustrates, a combination of filtering on numbers, symbols and tag words, returns the highest number of translation candidates, but does not return the least false positives. The most accurate of the processes are a combination of filtering on numbers and symbols.

Figure 5 demonstrates that the filters are not only applicable to the training set, by applying the three combined filters on the test set. The results prove to be similar, with a slight variation

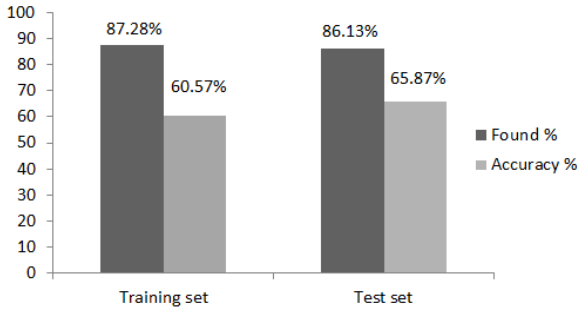


Figure 5. Comparing the results of all filters between the training and test sets

in both the number of results found and the accuracy of the results.

IX. Conclusion and Future Work

This paper set out to create a method for the selection of suitable mathematical translation candidates from queries received in Mxit lingo. Three techniques have been identified for this purpose and have been tested in combination to ensure the selection of the best method. It was determined that combining all three of the developed filters yields the most translatable results, it is however not the most accurate. If accuracy is a higher priority, then combining the number and symbol filters would be a better solution.

All of these techniques are light-weight, even using the combination of all three filters yield results in real-time. Future development could cross-reference the multi-filter processes to find a middle-ground between the number of results found and accuracy. However, this step might not be necessary as the speed at which the process takes place makes the time lost in processing false positives negligible.

As this study follows the design science research methodology [2], we have decided to use the design science guidelines [3] to evaluate whether the study's goals have been met. To address the first guideline, *design as an artifact*, the study has been structured to provide an implemented solution to detect translation candidates as it's final artifact.

The second guideline, *problem relevance*, is met by the fact that the study focuses on a real-world problem as demonstrated in the logs of Dr. Math. As it stands, the filters have been evaluated by comparing the results generated by the system itself. This does not take into account that the developed algorithms and testing procedures may be prejudiced or inaccurate. The third guideline, *design evaluation*, may be made more rigourous by using multiple coders and testing correspondence between coding through Krippendorff's alpha [21].

The study meets the fourth guideline, *research contributions*, by providing a means by which to enhance a real-world tutoring system applicable to over 30 000 school learners [1]. *Research rigour*, the fifth guideline, has been applied to the research by employing various methods on a training data set, learning from the first round of results and then applying changes before testing the methods on the training set again. Further rigour has been applied by testing the consistency of the methods and results on an alternate test data set.

The sixth guideline specifies that the *design should be a search process*. This search process was facilitated by employing three different filter types, a variety of natural language text processing techniques and finally modifying the techniques upon the receipt of initial data.

This paper serves to satisfy the last guideline, *communication of research*, which specifies that our research should be presented to an appropriate audience for verification.

The focus of this paper was on identifying techniques for the selection of suitable translation candidates. A future study may involve the development of rules for the translation of the selected candidates to mathematical equations.

References

[1] L. Butgereit and R. Botha, "A model to identify mathematics topics in MXit lingo to provide tutors quick access to supporting documentation." *Pythagoras*, vol. 32, no. 2, pp. 79–85, 2011.

[2] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A design science research methodology for information systems research." *Journal of Management Information Systems*, vol. 24, no. 3, pp. 45–77, Winter 2008.

[3] A. R. Hevner, S. T. March, and J. Park, "Design science in information systems research." *MIS Quarterly*, vol. 28, no. 1, pp. 75–105, March 2004.

[4] B. Goba, H. Morgan, K. Press, C. Smuts, and M. Van der Walt, *Study and Master Mathematics Grade 8*, 9th ed. Cambridge University Press, 2011.

[5] P. Carter, L. Dunne, H. Morgan, and C. Smuts, *Study and Master Mathematics Grade 9*, 12nd ed. Cambridge University Press, 2010.

[6] B. Goba and D. Van der Lith, *Study and Master Mathematics Grade 10*, 2nd ed. Cambridge University Press, 2008.

[7] D. Van der Lith, *Study and Master Mathematics Grade 11*, 5th ed. Cambridge University Press, 2008.

[8] ——, *Study and Master Mathematics Grade 12*, 9th ed. Cambridge University Press, 2010.

[9] South African Department of Education, "National Curriculum Statement Grades 10-12 (General) Mathematics," 2003.

[10] ——, "National Curriculum Statement Grades 10-12 (General) Mathematical Literacy," 2003.

[11] Various, "Webster's Unabridged Dictionary," Web, 2009, retrieved: 3 August 2012, http://www.gutenberg.org/ebooks/29765.

[12] W. Frakes and C. Fox, "Strength and similarity of affix removal stemming algorithms." *ACM SIGIR Forum*, pp. 26–30, 2003.

[13] J. Lovins, "Development of a Stemming Algorithm," *Mechanical Translation and Computational Linguistics*, vol. 11, no. 1 an 2, March and June 1968.

[14] M. F. Porter, *An algorithm for suffix stripping*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1997, pp. 313–316.

[15] C. D. Paice, "Another stemmer," *SIGIR Forum*, vol. 24, pp. 56–61, November 1990.

[16] Riptide Technology, "Easymark 2008 and EasyMark Worskpace 2008," Web, 2010, retrieved: 2 September 2012, http://www.riptidecc.com/softwaredevelopment.aspx.

[17] N. Schiller, J. Greenhall, J. Shelton, and A. Caramazza, "Serial order effects in spelling errors: Evidence from two dysgraphic patients," *Neurocase*, vol. 7, no. 1, pp. 1–14, June 2001.

[18] W. B. Cavnar and J. M. Trenkle, "N-Gram-Based Text Categorization," in *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, 1994, pp. 161–175.

[19] J. Mayfield and P. McNamee, "Single n-gram stemming," in *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, ser. SIGIR '03. New York, NY, USA: ACM, 2003, pp. 415–416.

[20] Stapel, E, "Translating Word Problems: Keywords," Web, retrieved: 3 August 2012, http://www.purplemath.com/modules/translat.htm.

[21] A. F. Hayes and K. Krippendorff, “Answering the call for a standard reliability measure for coding data,” *Communication Methods and Measures*, vol. 1, no. 1, pp. 77–89, 2007.

FastSLAM with Stereo Vision

Wikus Brink
Electronic Systems Lab
Electrical and Electronic Engineering
Stellenbosch University
Email: wikusbrink@ieee.org

Corné E. van Daalen
Electronic Systems Lab
Electrical and Electronic Engineering
Stellenbosch University
Email: cvdaalen@sun.ac.za

Willie Brink
Applied Mathematics
Department of Mathematical Sciences
Stellenbosch University
Email: wbrink@sun.ac.za

***Abstract*—We consider the problem of performing simultaneous localization and mapping (SLAM) with a stereo vision sensor, where image features are matched and triangulated for use as landmarks. We explain how we obtain landmark measurements from image features, and describe them with a Gaussian noise model for use with a Rao-Blackwellized particle filter-based SLAM algorithm called FastSLAM. This algorithm uses particles to describe uncertainty in robot pose, and Gaussian distributions to describe landmark position estimates. Simulation and experimental results indicate that FastSLAM is well suited for vision-based SLAM, because of an inherent robustness to landmark mismatches, and we achieve accuracies that are comparable to other state-of-the-art systems.**

I. Introduction

Simultaneous localization and mapping (SLAM) is a rapidly growing part of the autonomous navigation field. SLAM attempts to solve the problem of estimating a mobile robot's position in an unknown environment while building a map of the environment at the same time. This is a challenging problem since an accurate map is necessary for localization and accurate localization is necessary for mapping.

Most SLAM algorithms use a probabilistic landmark-based map rather than a dense map. If landmarks in the map can be measured, relative to the robot, and tracked over time the pose of the robot and the locations of the landmarks can be estimated in an optimal manner.

Initial implementations made use of the extended Kalman filter (EKF), but displayed several shortcomings such as quadratic complexity and sensitivity to incorrect feature tracking [1] [2]. The particle filter can be used to overcome these limitations. However, because of the high dimensionality of the problem the particle filter cannot be used directly. Instead, the Rao-Blackwellized particle filter [3] is used. This filter estimates some states with particles and others with EKFs. In the case of SLAM particles are used for the pose of the robot and an EKF for each landmark. This method is called FastSLAM and has shown promising results in the literature [4] [5].

Stereo vision is an attractive sensor to use with SLAM as it can provide a large amount of 3D information at every time step. Extracting that information reliably can, however, be challenging. Powerful algorithms such as SIFT [6] or SURF [7] have been used to solve this problem by extracting salient features from images. These algorithms can be employed to track features over multiple images so that landmarks for SLAM can be identified.

In this paper we attempt to solve the 2D SLAM problem by using FastSLAM and image features (the 3D extension is conceptually the same). We begin with a brief description of how we obtain measurements of landmarks with a Gaussian noise model. A detailed description of the FastSLAM algorithm is given, followed by some simulations where we compare FastSLAM with the popular EKF SLAM algorithm [2]. We provide experimental results from our system on an outdoor dataset and measure accuracy against differential GPS ground truth.

II. Image features and stereo geometry

In this section we discuss a method of finding features in images, triangulating these features for use as landmarks and approximating the noise associated with each measurement of a landmark. This characterization of the stereo vision sensor is important for accurate optimal estimation. Since this section is similar to previous work, the explanation will be brief. For a more in depth discussion refer to [8] and [9].

A. Feature detection and matching

In order to identify landmarks we opt for one of two popular feature detection algorithms: the scale-invariant feature transform (SIFT) [6] or speeded-up robust features (SURF) [7]. Note that since we perform SLAM in 2D we discard the vertical coordinates of image features.

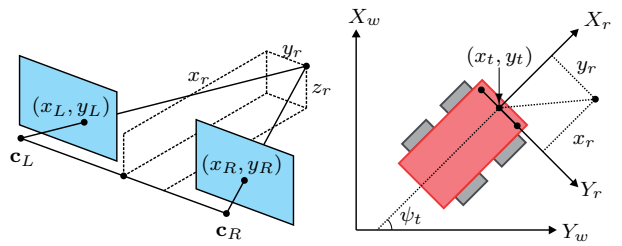
At every time step we search for feature matches in a synchronized pair of rectified stereo images. We model each match as a measurement with Gaussian noise:

$$\mathbf{z}_{\text{im}} = \begin{bmatrix} x_L \\ x_R \end{bmatrix} + \mathcal{N}(\mathbf{0}, \mathbf{N}_t), \tag{1}$$

where x_L and x_R are the image coordinates of the feature in the left and right images. By $\mathcal{N}(\mathbf{0}, \mathbf{N}_t)$ we mean a sample drawn from the normal distribution with zero mean and covariance matrix $\mathbf{N}_t$ (the same notation is used throughout the rest of this paper). We describe the noise covariance in Equation 1 by

$$\mathbf{N}_t = \begin{bmatrix} \sigma_{x_L}^2 & 0 \\ 0 & \sigma_{x_R}^2 \end{bmatrix}, \tag{2}$$

with σ_{x_L} and σ_{x_R} the standard deviations in pixels of the match measurement, which we obtain through testing.



(a) camera geometry (b) robot geometry

Fig. 1. The geometry of our system.

We can then match the descriptors of a new measurement with the descriptors of features already found at previous time steps, to arrive at putative landmark correspondences.

B. Stereo geometry of calibrated images

Now that we have stereo image features that can be tracked over time, we convert them into 2D landmarks.

Figure 1(a) depicts the geometry of a pair of stereo cameras with camera centres at $\mathbf{c}_L$ and $\mathbf{c}_R$, where the image planes have been rectified, and a landmark $\begin{bmatrix} x_r & y_r & z_r \end{bmatrix}^T$ observed at image coordinates (x_L, y_L) in the left image and (x_R, y_R) in the right image. As mentioned we are working in 2D, so the features are effectively projected onto the $X_r - Y_r$ plane.

With the geometry of the stereo camera pair, the landmark location in metres can be calculated in robot coordinates as

$$\begin{bmatrix} x_r \\ y_r \end{bmatrix} = \begin{bmatrix} \frac{fb}{x_L - x_R} \\ \frac{(x_L - p_x)b}{x_L - x_R} - \frac{b}{2} \end{bmatrix} + \mathcal{N}(\mathbf{0}, \mathbf{Q}_t), \tag{3}$$

where b is the baseline (distance between $\mathbf{c}_L$ and $\mathbf{c}_R$), f the focal length and p_x and p_y the x- and y-offset of the principal point, all obtained from an offline calibration process. $\mathbf{Q}_t$ is the noise covariance matrix of the measurement.

Note that we differentiate between robot coordinates (subscript r) and world coordinates (subscript w) as indicated in Figure 1(b), where x_t, y_t and ψ_t are the robot's position and orientation in world coordinates at time t.

We know that a transformation from $\mathbf{N}_t$ to $\mathbf{Q}_t$ is possible if we have a linear system and, since Equation 3 is not linear, we use a first order Taylor approximation to find the transformation matrix

$$\mathbf{W}_t = \begin{bmatrix} \frac{\partial x_r}{\partial x_L} & \frac{\partial x_r}{\partial x_R} \\ \frac{\partial y_r}{\partial x_L} & \frac{\partial y_r}{\partial x_R} \end{bmatrix}. \tag{4}$$

It then follows that $\mathbf{Q}_t$ can be approximated as

$$\mathbf{Q}_t = \mathbf{W}_t \mathbf{N}_t \mathbf{W}_t^T. \tag{5}$$

This approximation is performed to maintain a Gaussian noise model, which is necessary for FastSLAM. We use this noise model and the triangulated locations of landmarks to find outliers in putative correspondences between new measurements and those already in the map, according to the RANSAC-based probabilistic method discussed in [9].

From Figure 1(b) we see that the robot pose can be described with the state vector

$$\mathbf{x}_t = \begin{bmatrix} x_t \\ y_t \\ \psi_t \end{bmatrix}, \tag{6}$$

with x_t and y_t the location of the robot and ψ_t its orientation. We define the rotation matrix

$$\mathbf{R}_t = \begin{bmatrix} \cos(\psi_t) & -\sin(\psi_t) \\ \sin(\psi_t) & \cos(\psi_t) \end{bmatrix}. \tag{7}$$

In order to perform SLAM we need to establish a relationship between robot and world coordinates. We denote the location of a landmark i in the map corresponding with measurement j at time t as

$$\mathbf{m}_{i,t} = \begin{bmatrix} x_w \\ y_w \end{bmatrix} \quad \text{and} \quad \mathbf{z}_{j,t} = \begin{bmatrix} x_r \\ y_r \end{bmatrix}. \tag{8}$$

The measurement $\mathbf{z}_{j,t}$ will always be as the robot observes the landmark in robot coordinates, and the landmark's location $\mathbf{m}_{i,t}$ will always be given in world coordinates. The transformation between robot and world coordinates is given by the measurement equation

$$\mathbf{z}_{j,t} = \mathbf{h}(\mathbf{x}_t, \mathbf{m}_{i,t}) = \mathbf{R}_t^T \begin{bmatrix} x_w - x_t \\ y_w - y_t \end{bmatrix}, \tag{9}$$

or inversely,

$$\mathbf{m}_{i,t} = \mathbf{h}^{-1}(\mathbf{x}_t, \mathbf{z}_{j,t}) = \mathbf{R}_t \begin{bmatrix} x_r \\ y_r \end{bmatrix} + \begin{bmatrix} x_t \\ y_t \end{bmatrix}. \tag{10}$$

Exactly which measurement corresponds to which landmark in the map, as matched with the feature descriptors and confirmed with the outlier detection scheme, is stored in a correspondence vector $\mathbf{c}_t$.

III. Motion Model

Now that we have established a measurement equation, we need to derive a motion model for our robot so that we can perform SLAM. We use the velocity motion model. At every time step the controller of the robot will give it a forward and angular velocity,

$$\mathbf{u}_t = \begin{bmatrix} v \\ \dot{\psi} \end{bmatrix} + \mathcal{N}(\mathbf{0}, \mathbf{M}_t), \tag{11}$$

with v the forward translational speed and $\dot{\psi}$ the angular velocity. To characterize the uncertainty we add zero mean Gaussian noise with covariance matrix

$$\mathbf{M}_t = \begin{bmatrix} \alpha_1 v^2 + \alpha_2 \dot{\psi}^2 & 0 \\ 0 & \alpha_3 v^2 + \alpha_4 \dot{\psi}^2 \end{bmatrix}, \tag{12}$$

as is common practice [1]. The α parameters are robot and environment specific, and have to be estimated with practical testing and some degree of guesswork.

To update the robot states with the control input we define the motion equation as

$$\mathbf{x}_t = \mathbf{g}(\mathbf{x}_{t-1}, \mathbf{u}_t) = \begin{bmatrix} x_{t-1} \\ y_{t-1} \\ \psi_{t-1} \end{bmatrix} + \begin{bmatrix} \mathbf{R}_{t-1} \begin{bmatrix} vT\cos(\dot{\psi}T) \\ vT\sin(\dot{\psi}T) \end{bmatrix} \\ \dot{\psi}T \end{bmatrix}, \tag{13}$$

with T the sample period of the system. Although this is an approximation, the accuracy lost due to the approximation is far smaller than the effect of expected noise in the control input $\mathbf{u}_t$.

IV. SLAM WITH THE RAO-BLACKWELLIZED PARTICLE FILTER

The particle filter can be used to approximate any distribution, and it is often utilized to accurately estimate non-Gaussian systems. A major drawback of the particle filter, however, is that with high dimensional problems a large number of particles is needed to describe the distribution sufficiently. The Rao-Blackwellized particle filter has been developed to overcome this problem [3]. This filter uses particles to describe some states and Gaussian distributions to represent all other states. In order to utilize it we need to factorize the SLAM problem as

$$p(\mathbf{x}_t, \mathbf{m} \,|\, \mathbf{z}_{1:t}, \mathbf{u}_{1:t}) = p(\mathbf{x}_t \,|\, \mathbf{z}_{1:t}, \mathbf{u}_{1:t}) \prod_{i=1}^{n} p(\mathbf{m}_i \,|\, \mathbf{z}_{1:t}, \mathbf{u}_{1:t}). \tag{14}$$

With this factorization we describe the required posterior as a product of $n+1$ probabilities. If we suppose that the exact location of the robot is known, it is reasonable to assume that the landmark positions are independent from one another and can therefore be estimated independently. Naturally, we do not know the robot's location, but this independence can be utilized when we use particles to estimate the robot position. It can even be shown that the above factorization is exact and not an approximation [4].

FastSLAM uses a particle filter to compute the posterior over robot states, $p(\mathbf{x}_t \,|\, \mathbf{z}_{1:t}, \mathbf{u_{1:t}})$, and a separate EKF for every landmark in the map to obtain $p(\mathbf{m}_i \,|\, \mathbf{z}_{1:t}, \mathbf{u_{1:t}})$. What this means is that, instead of only one filter, we factor the problem into $1 + nm$ filters, where m is the number of particles. The large number of filters may seem excessive, but because of the low dimensionality of each individual filter the algorithm is remarkably efficient.

We define every particle to have a state vector for the robot states, and a mean vector and covariance matrix for every landmark, as

$$Y_t^{[k]} = \left\langle \mathbf{x}_t^{[k]}, \langle \mathbf{m}_{1,t}^{[k]}, \mathbf{\Sigma}_{1,t}^{[k]} \rangle, \ldots, \langle \mathbf{m}_{n,t}^{[k]}, \mathbf{\Sigma}_{n,t}^{[k]} \rangle \right\rangle, \tag{15}$$

with $\mathbf{x}_t^{[k]}$ the robot location and orientation for particle k, and $\langle \mathbf{m}_{i,t}^{[k]}, \mathbf{\Sigma}_{i,t}^{[k]} \rangle$ the i-th landmark's Gaussian mean and covariance. The FastSLAM algorithm, as it is executed at every time step, is given below in Algorithm 1. We proceed with a step by step explanation.

Algorithm 1 FastSLAM($Y_{t-1}, \mathbf{u}_t, \mathbf{z}_t, \mathbf{c}_t$)

1: **for** all particles $k \in \{1, 2, \ldots, m\}$ **do**
2: $\quad \mathbf{x}_t^{[k]} \sim p(\mathbf{x}_t \,|\, \mathbf{x}_{t-1}^{[k]}, \mathbf{u}_t)$
3: $\quad$ **for** all observed landmarks $\mathbf{z}_{i,t}$ **do**
4: $\quad\quad j = c_{i,t}$
5: $\quad\quad$ **if** landmark j has never been seen **then**
6: $\quad\quad\quad \mathbf{m}_{j,t}^{[k]} = \mathbf{h}^{-1}(\mathbf{x}_t^{[k]}, \mathbf{z}_{i,t})$
7: $\quad\quad\quad \mathbf{H}_j = \mathbf{J_h}(\mathbf{m}_{j,t}^{[k]})$
8: $\quad\quad\quad \mathbf{\Sigma}_{j,t}^{[k]} = (\mathbf{H}_j^{-1})\mathbf{Q}_i(\mathbf{H}_j^{-1})^T$
9: $\quad\quad$ **else**
10: $\quad\quad\quad \hat{\mathbf{z}} = \mathbf{h}(\mathbf{x}_t^{[k]}, \mathbf{m}_{j,t}^{[k]})$
11: $\quad\quad\quad \mathbf{H}_j = \mathbf{J_h}(\mathbf{m}_{j,t}^{[k]})$
12: $\quad\quad\quad \mathbf{Q} = \mathbf{H}\mathbf{\Sigma}_{j,t-1}^{[k]}\mathbf{H}^T + \mathbf{Q}_i$
13: $\quad\quad\quad \mathbf{K} = \mathbf{\Sigma}_{j,t-1}^{[k]}\mathbf{H}_j^T\mathbf{Q}^{-1}$
14: $\quad\quad\quad \mathbf{m}_{j,t}^{[k]} = \mathbf{m}_{j,t-1}^{[k]} + \mathbf{K}(\mathbf{z}_{i,t} - \hat{\mathbf{z}})$
15: $\quad\quad\quad \mathbf{\Sigma}_{j,t}^{[k]} = (\mathbf{I} - \mathbf{K}\mathbf{H_j})\mathbf{\Sigma}_{j,t-1}^{[k]}$
16: $\quad\quad\quad w^{[k]} = w^{[k]} f(\mathbf{Q}, \mathbf{z}_{i,t}, \hat{\mathbf{z}})$
17: $\quad\quad$ **end if**
18: $\quad$ **end for**
19: $\quad$ **for** all other landmarks $j' \notin \mathbf{c}_t$ **do**
20: $\quad\quad \mathbf{m}_{j',t}^{[k]} = \mathbf{m}_{j',t-1}^{[k]}$
21: $\quad\quad \mathbf{\Sigma}_{j',t}^{[k]} = \mathbf{\Sigma}_{j',t-1}^{[k]}$
22: $\quad$ **end for**
23: **end for**
24: **for** all $k \in \{1, 2, \ldots, m\}$ **do**
25: $\quad$ draw random particle k with probability $\propto w^{[k]}$
26: $\quad$ include $\left\langle \mathbf{x}_t^{[k]}, \langle \mathbf{m}_{1,t}^{[k]}, \mathbf{\Sigma}_{1,t}^{[k]} \rangle, \ldots, \langle \mathbf{m}_{n,t}^{[k]}, \mathbf{\Sigma}_{n,t}^{[k]} \rangle \right\rangle$ in Y_t
27: **end for**
28: **return** Y_t

- **Lines 1 and 2:** As with a normal particle filter, the FastSLAM algorithm begins by entering a loop over all the particles. The control input is used to sample a new robot pose for every particle according to the uncertainty in the motion model. We add random noise drawn from a zero mean Gaussian distribution with a covariance of $\mathbf{M}_t$, given in Equation 12, to the control input and use the motion equation $\mathbf{g}$, given in Equation 13, to find the new location and orientation of each particle.
- **Lines 3 and 4:** For every particle we enter a loop over all the measured landmarks. For every iteration the algorithm can do one of two things: add a new landmark, or update

an old landmark. The index of an old landmark in the map is given by the correspondence vector.

- **Lines 5 to 8:** A new landmark is added to the map using the measurement equation $\mathbf{h}$, given in Equation 9, to calculate its location in world coordinates. Since we want to use an EKF to estimate each landmark we have to linearize the measurement model by using a first order Taylor approximation with the Jacobian

$$\mathbf{J_h}(\mathbf{x}_t, \mathbf{m}_{j,t}) = \begin{bmatrix} \frac{\partial x_r}{\partial x_w} & \frac{\partial x_r}{\partial y_w} & \frac{\partial x_r}{\partial z_w} \\ \frac{\partial y_r}{\partial x_w} & \frac{\partial y_r}{\partial y_w} & \frac{\partial y_r}{\partial z_w} \\ \frac{\partial z_r}{\partial x_w} & \frac{\partial z_r}{\partial y_w} & \frac{\partial z_r}{\partial z_w} \end{bmatrix}. \quad (16)$$

  With this Jacobian we transform the uncertainty in measurement to an uncertainty in world coordinates.
- **Lines 9 to 15:** If a landmark has been observed before, we use the normal EKF equations to update its state vector and covariance. The state estimate is calculated by using the measurement model. The measurement model is then linearized with a Jacobian similar to the one used for new landmarks.
- **Line 16:** Once the landmark has been updated by using the measurement we have to calculate its effect on the weighting of the particle in question. As with a normal particle filter the importance weight is given by

$$w^{[k]} = \frac{\text{target distribution}}{\text{proposal distribution}}. \quad (17)$$

  The weighting function used in the algorithm can be shown [4] to be

$$f(\mathbf{Q}, \mathbf{z}_{i,t}, \hat{\mathbf{z}}) = |\mathbf{Q}|^{-\frac{1}{2}} e^{-\frac{1}{2}(\mathbf{z}_{i,t}-\hat{\mathbf{z}})^T \mathbf{Q}^{-1}(\mathbf{z}_{i,t}-\hat{\mathbf{z}})}. \quad (18)$$

  It is not necessary to update the weight for new landmarks as they will be the same for all particles, and therefore have no overall effect.
- **Lines 19 to 22:** If a previously observed feature has not been observed at the current time step its state vector and uncertainty will remain unchanged. All unobserved landmarks are therefore essentially ignored. This property of the algorithm is especially useful when a large map is maintained, as the number of unseen landmarks in the map does not impact the execution time.
- **Lines 24 to 27:** Resampling is done by drawing particles with a probability proportional to their normalized weights. Particles with low weights will be more likely to perish while particles with high weights will be copied and used at the next time step.
- **Line 28:** Finally the updated and resampled particles are returned to be used at the next time step.

A powerful possibility emerging from the use of particles is that of multiple hypothesis tracking. What it entails is that, since particles represent possible paths that the robot could have taken, we can calculate landmark correspondences for each particle separately. Because of the expensive nature of calculating feature matches we decide against this procedure and, instead, calculate one correspondence vector for all the particles. It is, however, important to note that the algorithm creates this possibility and future extensions can explore this feature.

V. Simulation

In order to test our SLAM systems we created a simulation environment that provides a realistic representation of the real world while facilitating a quantitative evaluation of the performance of the system.

A. Simulation environment

We created the environment with the aim of simulating the real world without it being unnecessarily complicated. We opted for a route through a corridor-like environment with landmarks on the walls. Although these landmarks are more structured than they typically would be in a real world situation, the structure should not influence the result significantly and should have the benefit of being easy to evaluate visually.

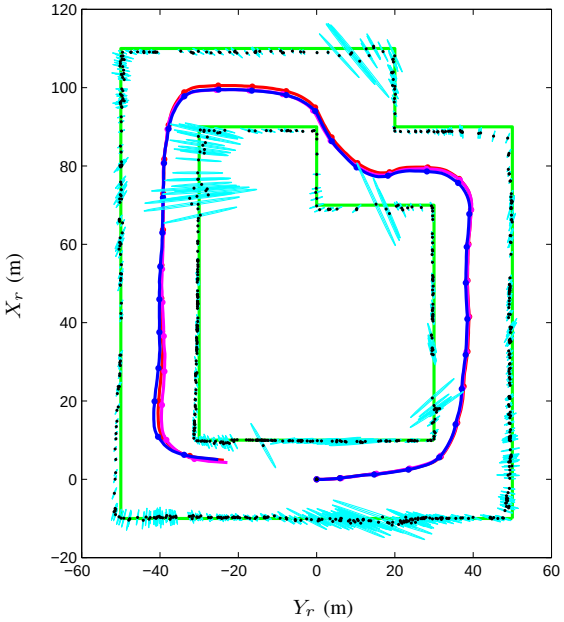
In order to create a control input we supply waypoints for the simulated robot to follow. At each time step a simple gain controller generates an input command that steers it towards the next waypoint. This control input is stored for use in the SLAM simulations but, before the robot executes the command, we add some Gaussian noise to simulate the uncertainty that we know exists in this process (in other words, we add process noise to the control input). The robot's actual motion from the noisy control is used as a ground truth trajectory and to generate the measurements.

As the robot moves through the environment, landmarks in the robot's field of view are included in the measurement at every time step. Because feature detectors will sometimes see a landmark at one time step and not at the next, even if it is in the field of view, we add a probability that a landmark will be seen. We project the landmarks onto the image planes of two cameras fixed on the robot and then add Gaussian noise to the pixel coordinates. Each landmark is assigned a unique scalar to be used as a descriptor. By changing or mixing these descriptors in a measurement we can simulate feature mismatches and investigate their effect on the accuracy of the SLAM system.

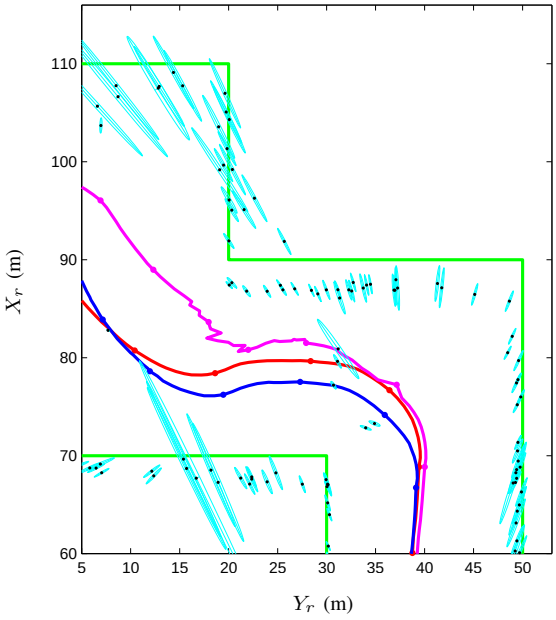
B. Simulation results

The simulation environment and the route and map as estimated by FastSLAM, using 250 particles, is depicted in Figure 2. At every time step each landmark has a 40% chance of being observed, but if it is observed, matching is done without error. When we display the route estimated by FastSLAM, we use a weighted average of the particles at every time step. In order to evaluate the accuracy we compare it to results obtained from another popular SLAM algorithm, namely EKF SLAM [8]. Results of the two algorithms are consistently similar in this simulation, even with varied noise parameters.

The experiment described above shows that it is possible to achieve accurate results using 250 particles with FastSLAM. To further investigate the relationship between the number



(a) simulation without landmark mismatches



(b) simulation with landmark mismatches

Fig. 2. The route and map from a simulation of FastSLAM, compared to EKF SLAM and ground truth (top). The bottom panel depicts an enlarged section of a simulation with landmark mismatches. The routes calculated with EKF SLAM are shown in magenta, the ground truth route in red and the environment walls in green. The estimated routes from FastSLAM are depicted in blue and the estimated landmark positions as black dots with corresponding confidence ellipses in cyan. Trajectories are shown with markers on every tenth time step.

of particles and accuracy we ran several simulations, each with a different number of particles. For every such number we ran the test 20 times in an attempt to remove the effect

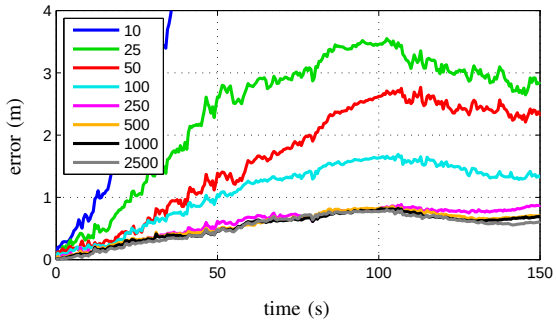


Fig. 3. The effect of different numbers of particles on the Euclidean error of the route estimated by FastSLAM.

of randomness introduced by the pose sampling step of the algorithm. Results of these experiments are shown in Figure 3.

We see that with FastSLAM in 2D, 250 particles is a good number to use as we do not lose much accuracy in comparison to using a larger number of particles.

In order to test the effect of landmark mismatches on the accuracy of FastSLAM we performed a simulation with such mismatches. The EKF SLAM algorithm is notorious for its inability to handle this kind of error [1] [8] and our simulation confirms this. With only six landmark mismatches over three time steps the EKF becomes unstable. With the same mismatches FastSLAM remains stable and introduces only a small degree of drift. This is a major practical advantage of the algorithm. These results are also depicted in Figure 2.

With these simulations we can establish, in a controlled environment, that FastSLAM achieves accuracy similar to EKF SLAM and is robust to landmarks mismatches. The following section describes our practical tests and results.

VI. Experimental results

The final step in our investigation and development of a FastSLAM system that uses stereo vision as a sensor is to test

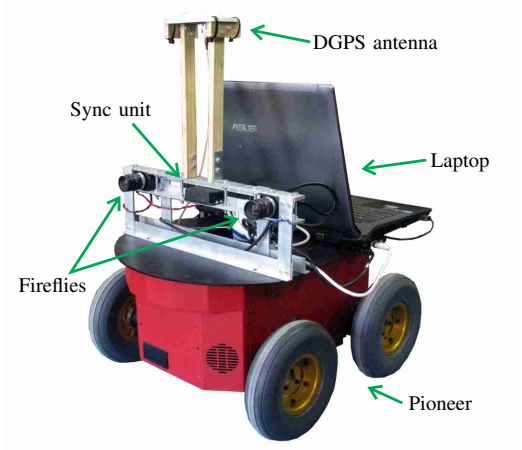


Fig. 4. Test platform.

Fig. 5. Sample frames (captured by the left camera) of the datasets used in our experiments.

the complete system with real world datasets.

A. Experimental setup and datasets

A real world dataset should ideally consist of a set of images captured by two synchronized and calibrated cameras, a control input and independently obtained ground truth location information that can be used to evaluate the performance of algorithms.

In order to capture such datasets we mounted a stereo camera set on a Pioneer 3-AT from Mobile Robots. We programmed the robot to execute a command given to it by a human using a joystick controller. At every time step we store the forward and rotational velocities so that they can be used as control input by the SLAM algorithms.

Our stereo camera rig consists of two Point Grey Firefly MV cameras with a synchronization unit we developed.

Ground truth data is recorded with a DGPS (accurate to about 5 cm) mounted on the robot. Note that this ground truth data is not used in our SLAM system, and is employed merely for evaluating results.

Figure 4 shows a picture of our test platform, indicating the various components.

When we work in a real world scenario we should expect problems such as bad lighting, uncluttered scenes (that give very few features), and a fair amount of shaking. We tried to capture realistic datasets that included these problems to a degree.

Two datasets were captured on the roof of the Electrical and Electronic Engineering building in Stellenbosch. The roof is a suitable environment to test 2D SLAM algorithms, since it is more or less flat. Apart from background trees moving in the wind it is also completely static.

The first of the two roof datasets includes a fair amount of maneuvering around two obstacles over a distance of about 45 metres. The second dataset comprises of a slow turn, a fairly long straight section, a three point turn with some reversing, and another straight section. The robot covered about 70 metres. Note that turning increases the process noise substantially because of wheel slippage.

A few frames of the datasets captured by one of the cameras are shown in Figure 5.

B. Experimental results

We show the results obtained from two experiments. The first was done using SURF features on the first dataset, and the second using SIFT features on the second dataset. These results are depicted in Figure 7 with corresponding location errors in Figure 6. We see that the Euclidean error from the first experiment grows over time. Drift is something that will be present with any localization system that does not employ absolute measurements (like GPS). In our work we attempt to limit this drift as much as possible.

We see that both SIFT and SURF can be used to obtain accurate results. Although we have no way of measuring the accuracies of the estimated maps, we can observe some structure and large quantities of landmarks located on the obstacles around which the robot moved.

VII. Conclusions

In this paper we investigated the use of the FastSLAM algorithm with landmarks originating from stereo image features. We explained how image features can be used as landmarks, with associated uncertainties in the form of Gaussian distributions. A measurement function converts features relative to the robot to landmarks in world coordinates and these landmarks are then matched over time, and outliers are identified and rejected. The FastSLAM algorithm then uses a particle filter to maintain the robot states, and for each particle a set of separate EKFs to estimate landmark locations.

We tested the system in a controlled simulation environment, and found that FastSLAM can be as accurate as EKF SLAM (when landmark matches are uncontaminated) but has the advantage of being largely unaffected by landmark

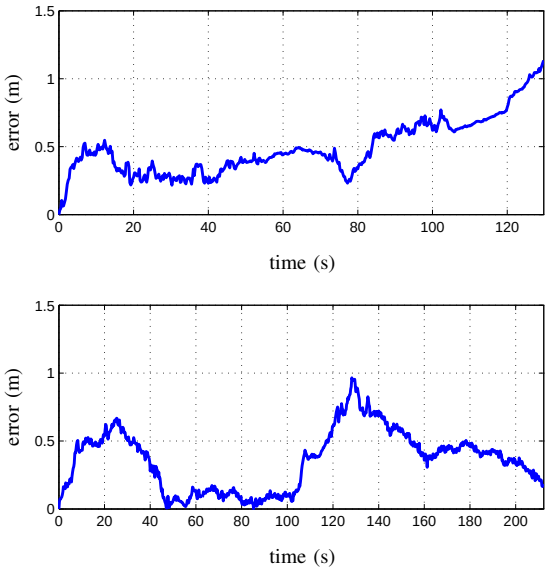


Fig. 6. The Euclidean error over time, as measured against DGPS, of the FastSLAM system using SURF features on the first dataset (top) and SIFT features on the second dataset (bottom).

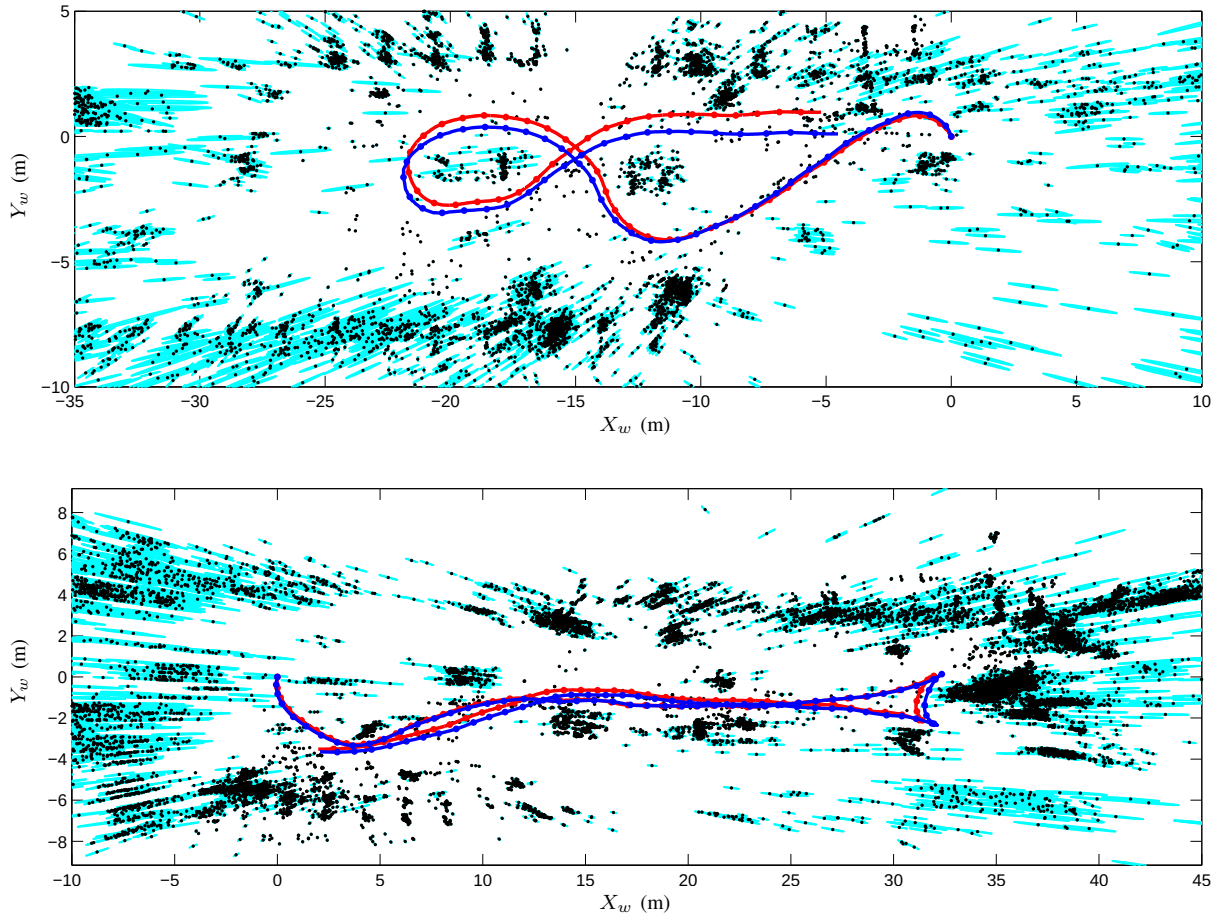


Fig. 7. Estimated routes (in blue starting at the origin) and maps from the FastSLAM algorithm using SURF features on the first outdoor roof datasets (top) and SIFT features on the second (bottom) with the DGPS ground truth in red. Markers are placed at every tenth time step of the routes. The landmarks that we show, as black dots with cyan confidence ellipses, are those that were observed on multiple time steps, i.e. those that contributed to the accuracy of the route estimation.

mismatches. This advantage of FastSLAM is significant, particularly when stereo features are used as landmarks, due to the unavoidable possibility of mismatches occurring. The problem of mismatches is inherent to image features, that often exhibit ambiguous characteristics, and we must therefore be able to rely on the SLAM system to remain stable in spite of such errors.

We also tested our complete FastSLAM system on data captured by a real robot. The accuracies achieved with either SIFT or SURF features are comparable to other state-of-the-art systems [10] [11].

We conclude that, because of its accuracy and robustness, FastSLAM can be a very effective algorithm to use with measurements from a stereo vision sensor.

REFERENCES

[1] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. MIT Press, 2006.

[2] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping (SLAM): Part I," *IEEE Robotics and Automation Magazine*, vol. 13, no. 2, pp. 99–110, 2006.

[3] A. Doucet, J. de Freitas, K. Murphy, and S. Russel, "Rao-Blackwellized particle filtering for dynamic Bayesian networks," *Conference on Uncertainty in Artificial Intelligence*, pp. 176–183, 2000.

[4] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM: A factored solution to the simultaneous localization and mapping problem," *Proceedings of the AAAI National Conference on Artificial Intelligence*, 2002.

[5] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with Rao-Blackwellized particle filters," *IEEE Transactions on Robotics*, vol. 23, no. 1, pp. 34–46, 2007.

[6] D. Lowe, "Object recognition from local scale invariant features," *IEEE International Conference on Computer Vision*, pp. 1150–1157, 1999.

[7] H. Bay, A. Ess, T. Tuytelaars, and L. van Gool, "Speeded-up robust features (SURF)," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[8] W. Brink, C. van Daalen, and W. Brink, "Stereo vision as a sensor for EKF SLAM," *22nd Annual Symposium of the Pattern Recognition Association of South Africa*, pp. 19–24, 2011.

[9] ——, "Probabilistic outlier removal for robust landmark identification in stereo vision based SLAM," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2822–2827, 2012.

[10] G. Dubbelman, W. van der Mark, and F. Groen, "Accurate and robust ego-motion estimation using expectation maximization," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3914–3920, 2008.

[11] J. Civera, O. Grasa, A. Davison, and J. Montiel, "1-point RANSAC for EKF-based structure from motion," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3498–3504, 2009.

Automation of Region Specific Scanning for Real Time Medical Systems

Denis Wong
University of Cape Town
Cape Town, South African
Email: denis.wong@uct.ac.za

Fred Nicolls
University of Cape Town
Cape Town, South Africa
Email: fred.nicolls@uct.ac.za

Abstract— **X-rays have played a vital role in both the medical and security sectors. However, there is a limit to the amount of radiation a body can receive before it becomes a health risk. Modern low dose x-ray devices operate using a c-arm which moves across the entire human body. This paper shows how radiation can be reduced on a human body by isolating the region that requires exposure. This work is based on a medical scanner that is still under development and therefore a prototype of the scanner is developed for running simulations. A camera is attached onto the prototype and used to point out the regions that are required to be scanned. This is both faster and more accurate than the traditional method of manually specifying the areas, as it also accommodates minor movements from the patient. An analysis is performed on the automation process as there are many variables such as speed, accuracy and searching thresholds that need to be catered for in the experiment. It is found that the correct region of interest can be located with the use of reliable feature points and that certain regions of the body are easier to locate than others. Currently, partial scans are done manually and this is a step forward towards automating the process completely.**

I. INTRODUCTION

Digital image processing has become a field of growing interest, especially in industries such as the medical sector. Digital image processing is achieved by a set of computer algorithms to perform image processing on digital images [1] to aid in the analysis of the human body. Image processing can be in the form of analysis or manipulation on digital images.

Lodox Systems designed and developed its medical x-ray scanners which originated from the Scannex [2], an x-ray security scanner developed by De Beers, to prevent diamond theft within the mining industry [3]. The Lodox scanners are unique in that they can produce a full body x-ray image within minutes after a single 13 second scan. Lodox Systems currently has a medical scanner in the market, called the Statscan, and is in the process of developing their latest medical device, the Versascan. The Versascan is designed to be a multi-purpose, self-contained and transportable digital radiography system for general and orthopedic radiography. It is a vertically-orientated scanner, so the patient is also vertically positioned. The apparatus used to capture data needs to mimic that of the Versascan as closely as possible, for experimental purposes. A simple garage motor track, vertically-orientated, is used to capture video data of subjects, with the aid of an attached camera. For the remainder of the paper, the garage motor track is referred to as the c-arm unless otherwise specified.

The Versascan can perform partial scans of the human body, but these have to be done manually. Performing partial scans manually leads to human errors when having to specify the starting and stopping points, using the laser, which is built in the c-arm, as a guide and requires that the patient stand very still during the scan. This paper aims to provide the information needed in order to perform a partial scan of the human body. This proposed setup uses the camera to capture a full body image of the patient during a pre-scan where x-rays are disabled. The operator uses the full body image to mark the region that requires scanning. Using a camera to locate the region of interest, in real-time, provides an allowance for the patient to move slightly. Therefore the main aim of this paper is to find whether it is possible to perform a partial scan, on the Versascan, with the aid of a camera. It is proposed that the camera be attached to the c-arm, above the x-ray source.

There are two important practical aspects, namely the workflow for the radiographer and the processes that occur to locate the marked region. Figure 1 is a flowchart containing both aspects for performing a partial scan automatically. The elliptical elements show the steps taken by the radiographer and the rectangular elements indicate what processes are performed, at each step, in order to locate the region of interest automatically. The workflow for the radiographer is important if the proposed modification is to be accepted for the Versascan design. The reason is because the radiographer needs to know what the procedures are in order to perform a partial scan and locate the region of interest automatically.

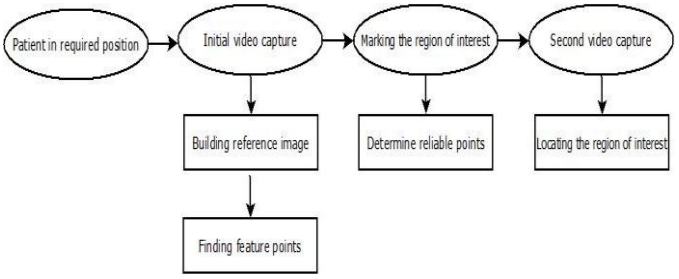


Figure 1: Flowchart of performing a partial scan automatically.

This approach requires that there are pairs of video data in order to perform the relevant tests, one for the reference image and one for the scanned image to search for the region of interest. The pairs of video data are of patients in standard poses with slight adjustments to their stances so as to mimic minor movements in a realistic situation.

Section II looks at relevant academic research from various papers, articles, websites and books in the image processing field. A discussion of the entire proposed workflow for the radiographer and the processes to perform a partial scan automatically is given in section III. All assumptions and experiments performed are evaluated in section IV. Section V presents a summary of the entire paper with its findings, followed by a discussion of ways in which this work could be taken further.

II. RELATED WORK

A. Template Matching

Template matching is commonly used for object recognition and stereo-matching. Template matching techniques compare portions of images against one another. Correlation values are calculated in the various positions that indicate how well the template matches the image. Correlation is a measure of the degree to which two variables agree, not necessarily in actual value but in general behaviour.

A common way to calculate the position (u, v) of the object in the search window is to evaluate the cross correlation coefficient value, *c*, at each point (u, v) for function *f* and the template *t*. The positions (u, v) represent the shift of *u* in the x-direction and by *v* in the y-direction. Cross correlation is motivated by the Euclidean distance which is a measure of similarity and is shown as

$$d(u,v) = \sqrt{\sum_{x,y}[f(x,y) - t(x-u, y-v)]^2}. \quad (1)$$

Euclidean distance is only appropriate for data measured on the same scale as no adjustments are made for differences in scale. Equation 2 is made by expanding *d* and is shown as,

$$d^2(u,v) = \sum_{x,y}[f^2(x,y) - 2f(x,y)t(x-u, y-v) + t^2(x-u, y-v)]. \quad (2)$$

In equation 2, if $f(x,y)$ and $t(x-u, y-v)$ are standardized, the sums are both equal to a constant value n. Therefore, $\sum f(x,y)t(x-u, y-v)$ is the only non-constant term just as it is in the reduced formula for the correlation coefficient:

$$c(u,v) = \frac{n\sum_{x,y} f(x,y)t(x-u,y-v) - (\sum_{x,y} f(x,y))(\sum_{x,y} t(x-u,y-v))}{\sqrt{(n\sum_{x,y} f^2(x,y) - (f(x,y))^2)(n\sum_{x,y} t^2(x-u,y-v) - (t(x-u,y-v))^2)}}. \quad (3)$$

Lewis states that there are a few disadvantages to using the cross correlation coefficient for a measure of similarity [4]. Some of the disadvantages mentioned are that the range of the correlation coefficient value is dependent on the size of the feature and that it is not invariant to changes in scale and lighting conditions. Lewis states that the difficulties with the cross correlation can be overcome by normalizing the image to unit length [4]. The normalized cross correlation is shown as

$$\gamma(u,v) = \frac{\sum_{x,y}[f(x,y) - \bar{f}_{u,v}][t(x-u,y-v) - \bar{t}]}{\sqrt{(\sum_{x,y}(f(x,y) - \bar{f}_{u,v})^2)(\sum_{x,y} t(x-u,y-v) - \bar{t})^2)}} \quad (4)$$

where $\bar{t}$ is the mean of the feature and $\bar{f}_{u,v}$ is the mean of $f\ (x,y)$. Normalized cross correlation is a popular measure of similarity as its easy hardware implementation makes it useful for real-time applications. Work has been done on increasing the performance of normalized cross correlation with the use of basis functions. Briechle and Hanebeck proposed using rectangular basis functions where the number of calculations depend linearly on the number of basis functions used [5]. The specific example used in [5] has an outcome that results in a computational reduction of 47 times using basis functions.

There have been some image matching methods performed based on normalized cross correlation [6, 7, 8]. However, these methods do not perform well as normalized cross correlation is not invariant to rotation. Zhao, et al. propose a hybrid method, consisting of both feature points and templates, to improve the results of normalized cross correlation [9]. The hybrid method consists of using feature points on the two images to determine the rotation and scale changes according to the characteristic scale and dominant direction of the points. The invariant normalized cross correlation is then applied at the corresponding feature points.

The main difference between the works of [9] and [5] is that the one potentially eliminates the measure's variance and the other increases its performance by reducing its computation time.

Another application for template matching is not only to classify an object but also to track it. Object tracking is usually categorized into two classes. One is where tracking takes place while the camera is stationary and the other is when the camera is moving. The most important characteristic is that to make a real-time system, the image captured by the camera must be processed before the next frame is digitized.

Pal and Biswas propose an automated correlation based tracking approach using edge strength and Hausdroff Distance Transform (HDT) technique for tracking moving targets [10]. The approach produces a complete real-time video tracking system for both detecting and tracking moving targets from optical image sequences.

Other methods used for detecting objects can be seen in [11] and [12] with differences being that the object's shape is known. Cole et al. describe how a 2D model can be used [11]

and Gupta et al. show how detecting objects can be done with a 3D model [12].

B. Feature Matching

The feature-based matching approach is the easiest method for finding image displacements. The method finds features in an image, such as edges and corners, and calculates the change in distances of the position of the feature points from the original image to another.

When video data is considered, the displacement is calculated from frame to frame. This is basically a two-step approach. Firstly, feature extraction is performed on two or more consecutive frames, to both reduce the amount of information to be processed and to obtain a higher level of understanding of the image scene. Secondly, these feature points are matched between frames to find any change in the positions of the points. Generally, changes in feature point positions between frames usually mean a movement of some object or background.

Feature-based matching is usually preferable when an image has strong features, such as sharp corners, in it. A feature-based approach is generally faster than a template-based approach because it does not consider the entire image but rather only the feature points found.

Edges indicate boundaries in an image, which makes them important for image processing. Edges in an image usually appear as intensity changes in pixels situated next to each other. There are many different methods of edge detection but they can be grouped into two categories, gradient-based and Laplacian-based. Mlsna and Rodriguez show that the difference between the two categories is that the gradient methods consider maximum values in the first derivative of an image and Laplacian methods look for zero crossings in the second derivative of an image [13].

It is shown in [14] that, in practice, a zero crossing filter is created by performing Gaussian smoothing followed by Laplacian filtering. The Laplacian-of-Gaussian (LoG), which is the convolution mask of the zero crossing operator, can be obtained from using various orders of linear filters and the rotational symmetry of Gaussian filter.

Few examples of other edge detectors are Sobel, the Canny, the Local Threshold and Boolean Function Based edge detectors [15] and color edge detection using euclidean distance and vector angle [16]. Nadernejad et al. have performed a greater analysis of various edge detectors, including the ones previously mentioned [17].

Corners are the intersections of two edges of sufficiently different orientations. Therefore corners contain two dimensional features and can potentially represent object shapes. The ability to represent object shapes play important roles in matching and pattern recognition.

There are many different corner detectors that exist such as the Principal Curvature-Based Region (PCBR) detector [18] and the Harris operator [19]. Corner detectors have many applications in motion tracking, stereo matching and image database retrieval. Mokhtarian and Suomela modify the corner detector to make it more robust, based on the curvature scale-space (CSS) representation [20]. The quality of a corner detector is determined by its ability to detect the same corner in multiple images of the same scene but under different conditions, like lighting, translation, rotation and other transformations.

The Harris corner detector is a good method to use to detect corners as it provides good quality corners under varying rotation and illumination and may detect interest points other than corners. For the purpose of this paper the Harris corner detector is considered due to its strong invariance to rotation, scale, illumination variation and image noise [21].

Harris and Stephens propose combining the corner and edge detector based on the local auto-correlation function to obtain feature points for tracking algorithms [22]. Weijer, et al. propose combining the two detectors by photometric quasi-invariants [23] and Ando by gradient covariance [24]. Parks and Gravel provide a detailed comparison of over 10 various corner detectors including ones mentioned previously [25].

Lowe developed and published the algorithm called Scale Invariant Feature Transform (SIFT) to detect and describe local features in images [26]. The University of British Columbia has patented this algorithm but it is available to the public for research purposes only and there are papers available by Lowe that give a better understanding of the SIFT keypoint detector method [27, 28, 26]. The SIFT algorithm is robust because, as the name suggests, it is able to handle image transformations like scale, rotation and deformation. There are four steps that SIFT goes through to transform image data into scale invariant coordinates relative to local features [26]. Aly has shown that SIFT can be used to find feature points in a face to identify a person for surveillance and access control [29]. There are various other applications that use SIFT such as image stitching [30], video tracking [31] and 3D modeling [32, 33].

The advantages and disadvantages of the two matching methods are mentioned in this section in order to get a better understanding of them. A good understanding of the matching methods is necessary in order to create an accurate and reliable online search to locate a region of the body.

III. WORKFLOW PROCESSES

To achieve a better understanding of the entire proposed system, the methods used are broken up into separate processes. Figure 1 shows the radiographer's workflow where the respective processes are performed at each stage.

A. *Reference Image*

The reference image is the first image displayed on the workstation and this is where the radiographer marks the region of interest. The reference image is obtained during the c-arm's first pass by stitching the initial video captured from the camera at 60 *fps*. Figure 2 shows examples of stitched reference images. Routine views are generally in the anteroposterior and lateral positions. Figures 2a and 2b show examples of the anteroposterior position and figures 2c and 2d show the lateral position.

(a) (b) (c) (d)

Figure 2: Examples of reference images. (a) and (b) Anteroposterior position. (c) and (d) Lateral position.

The video captured during the first pass consists of 780 frames. The reference images shown in this paper are constructed from video data consisting of 1080 frames. The reason for the additional 300 frames, or 5s, is due to the delay between operating the camera and the garage opener. It was found that the time taken for the workstation to configure the camera and begin capturing was inconsistent. To overcome this, a delay was implemented, such that after 3s of capturing a signal was sent to start the garage opener. The remaining 2s is used to cater for the approximate time taken for the slide to reach the other end.

Image stitching is the process in which multiple images are aligned by various registration algorithms and blended together in a seamless manner [34]. The video datasets were captured by a camera mounted on a vertically-orientated garage motor track. The reference image was created by taking a number of rows, r, at each frame captured. The rows that are used for stitching the reference image are at the centre of every frame. The method of stitching performed uses a number of rows at each frame and stacks them underneath or above each other, depending on the scan direction, i.e. pass 1 or pass 2. The result of taking into account the centre rows of the video data produces the reference image.

By using the centre rows of every frame, it is effectively providing the information that is directly in front of the camera. However, for the second pass, the camera would have to 'look-ahead' to identify the area before the c-arm reaches it. Using the proposed camera configuration, a maximum look-ahead distance of 179.13mm and 435.13mm is achieved if the c-arm travels downwards and upwards respectively for its second pass. Therefore, a recommendation is made that the c-arm's first pass start from the top moving downwards and from the bottom moving upwards for its second pass.

The method of image stitching mentioned causes some concern for data loss as it simply takes a number of rows at the centre in each frame and constructs the reference image from that. It is seen that by considering two rows at a time, a loss of only 0.16% is obtained which is seen to be minimal.

B. *Finding Feature Points*

Feature points are important as they are used to locate the region of interest on the scanned image. However, one problem was found with this approach: when there is ambiguity or patterns present around the region of interest, corners are sometimes found at other locations that are visually similar. To remove corners that are either ambiguous or found in patterns, a need for more reliable feature points is necessary.

C. *Determining Reliable Points*

The process of determining reliable feature points occurs after the radiographer has marked the region of interest, as only the feature points which fall within the marked region are considered and the rest are ignored. Only feature points within the marked region are considered as these are the ones used to locate the region of interest on the scanned image.

The approach for determining whether a feature point is reliable or ambiguous consists of looking at the neighbourhood of each point. A small window of size 15 × 15, centred at the detected corner, is considered. Template matching is then performed over the surrounding area of size 75 × 75 to see whether there are other locations which have similar appearances. Normalized cross correlation is selected to measure how similar the feature point is to the background. Various thresholds have an impact on the resulting correlation value which determines how reliable each corner is and this is discussed in more detail in the next section.

D. *Locating the Region of Interest*

After the region of interest has been marked, the radiographer controls the c-arm to perform the second pass. The second set of video data is not only being stitched together, but also being used to locate the region of interest.

In order to find the region of interest and provide the location to the x-ray source before it passes it, a search is required to take place ahead of the c-arm. Searching ahead of the c-arm is done by adjusting the stitching method during the second pass. Instead of using two rows at the centre of every frame, higher rows are considered. The look-ahead distance is not set to a constant value but is varied depending on the height of the marked region. In the case where the height of the marked region is greater than the maximum look-ahead distance, the maximum look-ahead distance is then considered.

An online searching method is necessary in order to locate the region of interest in the scanned image efficiently and

accurately. The time taken to locate the region of interest is important as it is necessary to identify the marked area before the c-arm reaches it.

The time available for the online search is catered for with the varying look-ahead distance. In order to achieve accuracy within 2% source to image detector distance (SID), the actual region scanned as a result of the search needs to fall within a distance of 20mm from the region marked by the radiographer.

The approach is to use reliable points found within the region of interest, on the reference image, in order to identify the respective area in the scanned image. A template of size 15 × 15 pixels centred on each reliable point is used for the search in the scanned image. To cater for minor movements, a search window of size 51 × 51 pixels is used to provide an allowance of minor movements of 25 pixels in any direction. One factor that determines the size of the search window is the accuracy as a distance error of more than 20 pixels is greater than 5%, which is regarded as a failed test.

The search for the matches for the reliable points on the scanned image yields normalized correlation coefficient values at each point within the search window. The point with the highest match value is regarded as the best match. If the highest match value is greater than some search threshold, then that point is considered a reliable match. If it is below the threshold then the corresponding match is determined to have not been found and the match pair is thus ignored. Once a specified number of corresponding reliable points are found, an estimate of the marked region on the scanned image can be calculated.

The estimated location of the marked region on the scanned image is calculated with the use of the pixel coordinates of both the original and the corresponding match points found. First, the pixel distances are measured, both horizontally and vertically, between each reliable point and the marked region on the reference image. These distances are then transferred to the corresponding match points and are used to calculate the location of the marked region on the scanned image. In principle the marked region can be found by using the pixel distances measured on any single reliable match as they all indicate the location of the marked area.

Figure 3 shows the reference image, on the left, and the scanned image, on the right, as the c-arm moves downwards and performs an online search. In this case, the number of corresponding reliable points is 2. The yellow line indicates the position of the c-arm and the green line indicates the camera's viewpoint, which is also the last row that has been stitched. A closer look at the scanned image is needed to see where the estimated location of the marked region is. Figure 4 shows the same scanned image from figure 3 with the addition of the red box which indicates the estimated location of the marked region.

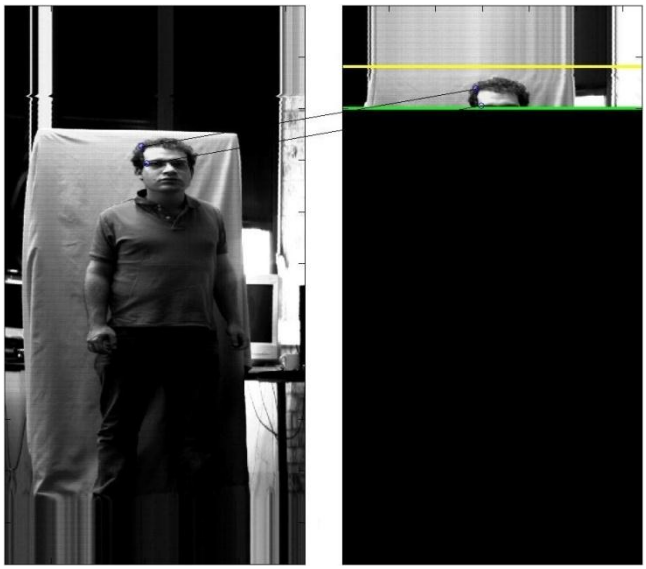


Figure 3: Result of online search after finding two corresponding reliable points. The yellow and green lines indicate the position of the c-arm and the camera's viewpoint respectively.

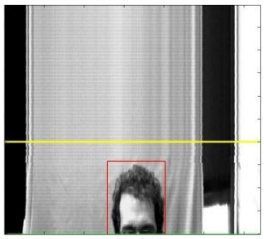


Figure 4: Result of online search indicating the location of the marked region on scanned image. The yellow and green lines indicate the position of the c-arm and the camera's viewpoint respectively. The red box is the estimated location of the marked region.

IV. Experiments and Results

This section provides a detailed analysis of the experiments performed on the datasets acquired for this paper.

A. Ground Truth

A measure of accuracy needs to be defined for the estimated location of the region of interest on the scanned image. A maximum error of 2% of the SID is allowed in order for the proposed modification of attaching a camera onto the c-arm to be accepted for the Versascan. The ground truth is only used as a measure of accuracy to see how well the search method performs.

Visual inspection can be used to see whether the estimated marked area has captured the required body region, but this doesn't provide a quantified accuracy measure. Therefore, once the second video has been captured, another search is performed. However, in this case instead of doing a progressive search, all of the reliable points on the reference image are used. The region found using all the reliable points, referred to as the ground truth, is then compared against the estimated marked area for an accuracy measure. The ground truth is assumed to be the closest location to the original marked region. In addition to the ground truth, a visual test is

also made to determine whether the correct marked region is found on the scanned image. The distance between the ground truth and the estimated region is the distance error used to determine accuracy in pixels.

Using the previous example where figure 4 shows the estimated location of the marked region on the scanned image, the ground truth is determined and shown in figure 5. Figure 5 shows the entire scanned image where the red and green boxes represent the estimated locations of the marked region and the ground truth respectively.

Using figure 5 as an example, the error is found to be 7 pixels horizontally and 4 pixels vertically which is equivalent to 17.92mm and 10.24mm. Therefore, the example illustrates a successful test as it resulted in locating the region of interest correctly within 2% accuracy. The results of each test in the experiment are analyzed using the ground truth to determine whether the correct region has been found and to what accuracy it is.

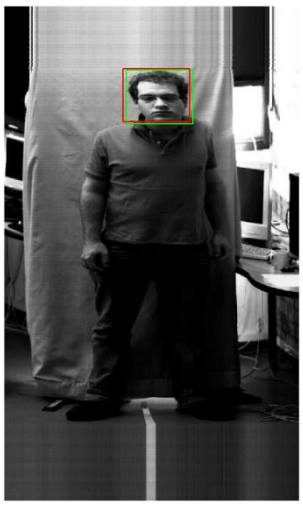


Figure 5: Entire scanned image with an estimated location of the region of interest and ground truth indicated by red and green respectively.

B. Thresholds

Various thresholds were mentioned in the workflow processes, all of which influence the results in some way. An evaluation is made on the different thresholds and a range of optimum values is identified that give a suitable result. The three thresholds evaluated are the number of reliable points found on the scanned image, the correlation coefficient value when searching for the reliable points, and the reliability measure of the feature points used.

The experiment consists of two hundred and sixty tests where different regions of the body were marked and searched for. The results in this section have been captured by repeating the experiment and changing the various thresholds accordingly. For experimental purposes, tests which have achieved an error within 2%, 3% and 5% are recorded as passed tests as the marked area identified on the scanned image contains the body region.

Each threshold is varied and a recommendation is made based on two results, the percentage of tests passed and the percentage of those passed tests that are within 2% and 3% error. The outcome of combining the two results is a percentage of tests passed within a certain accuracy. Therefore, a recommendation is made for each threshold based on the combination of the two results. For all the accuracy illustrations, the red and blue points show passed tests within 2% and 3% accuracy respectively.

The number of matches required is an important parameter in the matching process. If this parameter is set to be one, the estimated location of the region of interest would be obtained from the relative position of a match of one point. This makes the process of finding the estimated region fast but potentially inaccurate. On the other hand, if the parameter is set too high then the estimated region might not be found because the number of matches required within the marked region might never be obtained

Therefore, the varying number of matches required used for the experiment are 2, 5, 10, 20 and 50. This particular experiment used the search and point reliability threshold of 0.9 and 10 respectively. It is found that the greater the number of matches required, the more tests that fail. Figure 6a shows the results of the range of the number of required matches considered. The results suggest that one should use a low number of matches for the search. However, figure 6d shows an increase in obtaining more accurate results as the number of required matches increases to approximately 20. The product of combining the two results are, in order: 0.225, 0.273, 0.168, 0.092 and 0.004. Therefore a recommendation is made to set the number of matches required to 5 to cater for both correct and accurate results.

The correlation coefficient values from the search must be greater than the search threshold for a match to be declared. The highest correlation coefficient value around the search window is then used as the best match location to where the matching point is. The search threshold is therefore an indication of how good a match has to be for it to be considered reliable. The thresholds used for this experiment are 0.8, 0.85, 0.9, 0.95 and 0.98. This experiment used a required number of matches and a point reliability threshold of 5 and 10 respectively.

Figure 6b shows the results of the experiment where an increase in the search threshold results in a decrease in the percentage of tests passed. The accuracy of the tests passed is not drastically affected by the varying search threshold, as shown in figure 6e. The product of combining both results are, in order: 0.257, 0.271, 0.273, 0.210 and 0.136. Therefore, a recommendation is made to use a search threshold of 0.9 to cater for both correct and accurate results.

A test is performed on each feature point individually to specify whether it is reliable or ambiguous. This test uses

template matching of size 15 × 15 and a search window of size 75 × 75 with each feature point as its centre, to determine whether there is a similar point nearby. Normalized cross correlation is used as the template matches around the search window.

All the correlation coefficient values are evaluated and accumulated as a weighting to how reliable the point is. If the correlation coefficient is 1, this is generally the case where the template is in its original position and therefore ignored. If the correlation coefficient is greater than 0.95, it is assumed that there is a similar template in the search window and 5 is added to the accumulated weighting. If it is greater than 0.9, then only 1 is added as it is not strongly similar. If the highest correlation coefficient value in the search window is less than 0.9, it is assumed that there are no points similar and ignored.

The accumulated weighting value is then compared to the point reliability threshold. If the weighting is smaller than the point reliability threshold then it is identified as a reliable point. If the weighting is greater than the point reliability threshold it is identified as an ambiguous point and is ignored when performing a search for the marked region. In the experiment, the point reliability threshold values considered are 10, 20 and 50.

To see the effects of the point reliability threshold in the experiment, the required number of matches and search threshold has been set to 5 and 0.9 respectively. It is shown in figure 6c that a higher point reliability threshold results in an increase towards the percentage of tests passed. However, reliable points achieve better accuracy than non-reliable points, as shown in figure 6f. The product of combining the two results are, in order: 0.290, 0.250 and 0.226. Therefore, a recommendation is made to use a point reliability threshold of 10 to cater for both correct and accurate results.

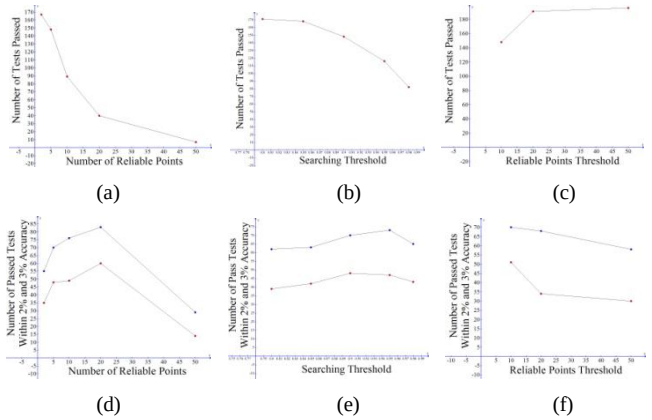


Figure 6: Results of experiment varying various thresholds. (a), (b) and (c) shows tests passed with varying various thresholds. (d), (e) and (f) show the corresponding tests passed within 2% and 3% accuracy.

C. *Performance of Different Body Regions*

An evaluation is performed on each body region individually to see if some regions are found more easily than others. Recommended values for the different thresholds, mentioned previously, are used in determining the performance of different body regions.

An analysis is performed on each body region to see whether some regions perform better than others. Figure 7 shows the percentage of tests passed for each region.

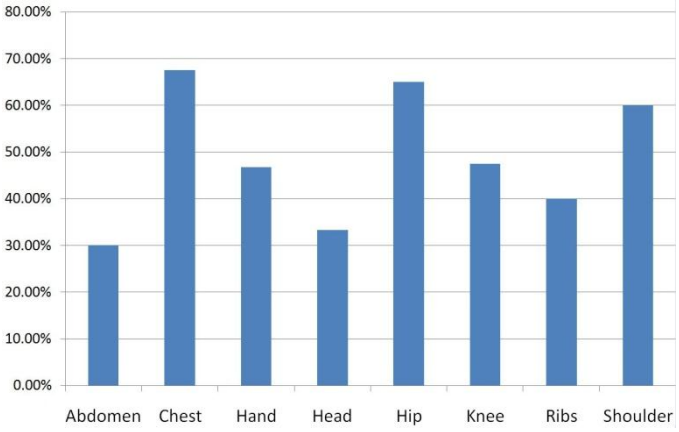


Figure 7: Results of tests passed for each body region.

The poorer performing regions, the abdomen, head and ribs, have been excluded to observe how it affects the overall performance. Figure 8 shows the performance using all the body regions and the other excluding the poorer performing regions. The blue, red and green indicates the tests passed and accuracies within 2% and 3% respectively. Removing the poorer performing regions results in an increase in the number of tests passed without having an impact on accuracy. This shows that certain regions of the body are easier to locate than others using the proposed online search.

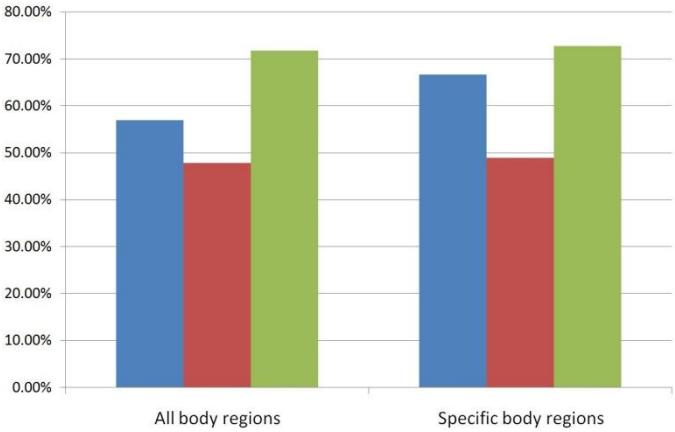


Figure 8: Results of the experiment using all and only specific regions.

V. CONCLUSIONS AND FUTURE WORK

The most significant result is that it is possible to automate the search for a region of interest on a real-time medical scanner. After performing an experiment consisting of 260 tests, it has been found that it is possible to locate a region on the body, marked by the radiographer, with the aid of a camera attached to a c-arm.

The main factors that influence the search were the thresholds placed on the number of matches required, the search threshold, and the reliability of the feature points. An evaluation on the various thresholds, which consisted of varying the threshold values, was performed in order to see the impact on the results and a recommended value was provided for each threshold. Taking the recommended threshold values into consideration, the results are found to have an overall performance of 57% of which 48% and 72% were within 2% and 3% accuracy.

It was also found that certain regions of the body were easier to locate than others. When ignoring the regions that were harder to locate, the abdomen, chest and head, the overall performance had increased to 67% of which 49% and 73% were within 2% and 3% accuracy respectively. This shows that certain regions of the body are easier to locate than others.

Actual data could not be acquired as the Lodox Versascan is still under development. This was overcome by mimicking the Versascan environment as closely as possible and obtaining datasets accordingly. Once the Versascan is operational, it would be of interest to acquire datasets from the actual device and compare them to the results found in this paper. As this experiment has shown, the method used to perform an online search to locate the region of interest is moderately successful.

REFERENCES

[1] I. Pitas. *Digital Image Processing Algorithms and Applications.* John Wiley and Sons, Inc., 2000.

[2] De Beers. SCANNEX X-ray Body Scanner, 2011.URL *http://www.debeersgroup.com/ImageVault/Imagesid_1893/scope_0/ImageVaultHandler.aspx*

[3] HERCA Working Group 2. Facts and Figures Concerning the use of Full Body Scanners using X-rays for Security Reason. Oslo *HERCA Plenary Meeting*, June 2010.

[4] J.P. Lewis. Fast Normalized Cross-Correlation. Visual Interface, Canadian Image Processing and Pattern Recognition Society, pages 120-123, 1995

[5] K. Briechle and U.D. Hanebeck. Template Matching using Fast Normalized Cross Correlation. In *Proc. SPIE 4387*, pages 95-102, 2001

[6] W. Forstner. A Feature-Based Correspondence Algorithm for Image Matching. *International Archives of Photogrammetry and Remote Sensing*, 26(3):150-166, 1986.

[7] Z. Zhane, R. Deriche, O. Faugeras and Q.T. Luong. A Robust Technique for Matching Two Uncalibrated Images Through the Recovery of the Unknown Epipolar Geometry. *Artificial Intelligence Journal*, 78(1), 1995.

[8] M. Pilu. A Direct Method for Stereo Correspondence Based on Singular Value Decomposition. *Computer Vision and Pattern Recognition*, pages 261-266, 1997

[9] F. Zhao, Q. Huang and W. Gao. Image Matching by Normalized Cross-Correlation. In Proc. IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP2006, 2:14-19, 2006

[10] S. Pal and P.K. Biswas. Modified Hausdorff Distance Transform Technique for Video Tracking. *Indian Conference on Computer Vision, Graphics and Image Processing*, 2000.

[11] L. Cole, D. Austin and L. Cole. Visual Object Recognition using Template Matching. *Australian Conference on Robotics and Automation*, 2004.

[12] N. Gupta, R. Gupta, A. Singh and M Wytock. Object Recognition using Template Matching, 2008. URL *http://www.stanford.edu/class/cs229/proj2008*.

[13] P.A. Mlsna and J.J. Rodriguez. Gradient and Laplacian Edge Detection. *The Essential Guide to Image Processing (Second Edition)*, pages 495-524, 2009.

[14] D. Csetverikov. Basic Algorithms for Digital Image Analysis. *Institute of Informatics, Budapest*, 2009.

[15] M.B. Ahmad and T.S. Choi. Local Threshold and Boolean Function Based Edge Detection. *IEEE Transactions on Consumer Electronics*, 45(3):674-679, August 1999.

[16] S. Wesolkowski and E. Jernigan. Color Edge Detection in RGB Using Jointly Euclidean Distance and Vector Angle. *In Proc. Of the IAPR Vision Interface Conference*, pages 19-21, May 1999.

[17] E. Nadernejad, S. Sharifzadeh and H. Hassanpour. Edge Detection Techniques: Evaluations and Comparisons. *Applied Mathematical Sciences*, 2(31):1507–1520, 2008.

[18] H. Deng, W. Zhang, E. Mortensen, T. Dietterich and L. Shapiro. Principal Curvature-Based Region Detector for Object Recognition. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1-8, July 2007.

[19] K.G. Derpanis. The Harris Corner Detector. Technical report, York University, October 2004.

[20] F. Mokhtarian and R. Suomela. Robust Image Corner Detection Through Curvature Scale Space. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12):1376–1381, December 1998.

[21] C. Schmid, R. Mohr and C. Bauckhage. Evaluation of Interest Point Detectors. *International Journal of Computer Vision*, 37(2):151–172, June 2000.

[22] C. Harris and M. Stephens. A Combined Corner and Edge Detector. *In Alvey Vision Conference*, pages 147-151, 1988.

[23] J. van der Weijer, T. Gevers and J.M. Geusebroek. Edge and Corner Detection by Photometric Quasi-Invariants. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(4): 625–630, April 2005.

[24] S. Ando. Image Field Categorization and Edge/Corner Detection from Gradient Covariance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(2):179–190, February 2000.

[25] D. Parks and J.P. Gravel. Corner Detection. URL *http://www.cim.mcgill.ca/dparks/CornerDetector/harris.ht*.

[26] D.G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, January 2004.

[27] D.G. Lowe. Object Recognition from Local Scale-Invariant Features. *International Journal of Computer Vision*, 2:1150-1157, September 1999.

[28] D.G. Lowe. Local Feature View Clustering for 3D Object Recognition. *IEEE Conference on Computer Vision and Pattern Recognition*, 1:682-688, December 2001.

[29] M. Aly. Face Recognition using SIFT Features. Computer Science Department, California Institute of Technology, 2006.

[30] Z. Hua, Y. Li and J. Li. Image Stitch Algorithm Based on SIFT and MVSC. *Seventh International Conference on Fuzzy Systems and Knowledge Discovery*, pages 2628-2632, September 2010.

[31] Z. Chaoyang. Video Object Tracking using SIFT and Mean Shift. Master's thesis, Chalmers University of Technology, Sweden, 2011.

[32] L. Lei. Three Dimensional Shape Retrieval using Scale Invariant Feature Transform and Spatial Restrictions. Technical report, National Institute of Standards and Technology, August 2009.

[33] S. Se and P. Jasiobedzki. Stereo-Vision Based 3D Modeling and Localization for Unmanned Vehicles. *International Journal of Intelligent Control and Systems*, 13(1):46–57, March 2008.

[34] I. Pitas. Digital Image Processing Algorithms and Applications. John Wiley and Sons, Inc., 2000.

Automatic segmentation of TIMIT by dynamic programming

Van Zyl van Vuuren*, Louis ten Bosch† and Thomas Niesler*

*Department of Electrical and Electronic Engineering
University of Stellenbosch, South Africa
Email: {15446204,trn}@sun.ac.za

†Department of Linguistics
Radboud University Nijmegen, The Netherlands
Email: l.tenbosch@let.ru.nl

***Abstract*—We propose an algorithm based on the principle of dynamic programming for the automatic segmentation of continuous speech into phoneme-like units. A measure of local dissimilarity among consecutive feature vectors is combined with a knowledge of the expected statistical distribution of the segment lengths within a dynamic programming framework to obtain an optimal placement of segment boundaries. We compare the performance of our algorithm with the performance of two recently-proposed alternatives by measuring how closely the hypothesised boundaries match the TIMIT phone boundaries. The results showed that we are able to improve on the performance of the two contrasting approaches. Furthermore, we show that a hybrid approach which combines aspects of all three algorithms leads to even better results.**

I. Introduction

The task of accurately segmenting a speech signal into phoneme-like units plays an important role in the speech processing field. Although accurate manual segmentation can be achieved by trained phoneticians, the task is tedious, expensive and intrinsically subjective. In HMM-based ASR systems, time-aligned phonetic transcriptions are often needed for the development of the pronunciation dictionary and acoustic models. This is not always feasible, and is a particular obstacle for the development of ASR systems for under-resourced languages, for which no, or very little, transcribed phonetic material is available. In these situations, automatic segmentation algorithms can accelerate the task of developing a pronunciation dictionary and obtaining suitable bootstrapping acoustic training data, thereby substantially reducing the time it would take to develop the ASR system. The availability of reliable automatic segmentation algorithms is also useful in technologies outside ASR, such as the study of pronunciation variation and the development of coherent large-scale dictionaries.

Several approaches to the automatic segmentation of speech have been proposed over the years. Some require prior training, relying for example on HMM forced alignments [1]. Others make use of previously stored speech segments for template matching by using the phonetic transcription [2]. A third and more prevalent class of algorithms rely solely on the acoustic information to detect transient events in the speech signal [3]–[7]. When considering a under-resourced setting in which speech corpora are unavailable or very small, model training may not be feasible. Under these circumstances this latter class of algorithms represents the most viable option.

In this paper we propose a new algorithm for the acoustic segmentation of speech based on the principle of dynamic programming (DP). DP-based segmentation has been proposed in [3], in which a distortion metric within segments is minimised by using prior knowledge of the number of phones in a sequence. The algorithm we propose requires no information regarding the number of phones and maximises the probability of a specific segment boundary sequence.

A well known class of speech segments are phonemes, the identification of which is the goal of most published segmentation algorithms. By using the annotated phoneme boundaries given in TIMIT, the acoustic characteristics in the vicinity of the phoneme boundaries as well as the lengths of the phonemes can be inspected. The proposed algorithm then uses this prior information to infer the probability of a boundary occurring at every specific point in time in a speech signal. Dynamic programming principles are then applied to detect the most probable sequence of boundary positions.

Section II gives a brief overview of the class of segmentation algorithms based on transient events in acoustic information, and includes a discussion on a few recent algorithms. Section III provides a detailed description of the proposed DP-based segmentation algorithm, and Section IV discusses the quality measures used to assess segmentations. The experimental setup is specified in Section V, and experimental results are given in Section VI. Finally concluding remarks are presented in Section VII.

II. Background

Many segmentation algorithms are based on the assumption that there are regions in speech, termed speech segments, where the acoustic features stay relatively constant, and that there are clear transitions between such regions. To detect these transitions, the algorithms employ some estimate of the local acoustic change in the signal. 'Local' in this context refers to temporal acoustic changes taking place at a specific time independent of any previous or future acoustic changes within the signal. A function that quantifies these local acoustic changes will be referred to as the **local score** function in the remainder of this text. The local score function is central to all acoustic segmentation methods, and therefore different types of local score functions and their application in the recent literature will briefly be reviewed.

A. Algorithms based on maximum local acoustic change

The most common approach used in speech segmentation is to hypothesise segment boundaries at the times at which local acoustic change is at a maximum. These local maxima are found by searching for the peaks in the local score. However, the local score may contain many small peaks, which are the result of small acoustic changes that do not necessarily indicate segment boundaries. These additional peaks can lead to *over-segmentation*, where more than one segment boundary is hypothesised while only one is truly present. Over-segmentation can be reduced by including a threshold below which peaks are ignored. A selection of segmentation algorithms falling into this category are reviewed in the following. They were specifically chosen to illustrate a diversity of local score functions, of which a

selection will later be compared experimentally. The local score will henceforth be denoted as LS in equations.

1) Räsänen et al. [4]: The local score function used in this algorithm is the cross correlation between two FFT magnitude vectors. This is shown in Equation 1, where f and g represent the FFT magnitude vectors for the frames to the left and to the right respectively of the investigated frame, F_j.

$$LS(F_j) = \frac{f.g}{\parallel f \parallel \parallel g \parallel} \tag{1}$$

Feature vectors that are similar will give a score close to 1, and dissimilar vectors will give a score closer to 0. The algorithm applies a non-linear filter to the cross-correlation sequence in order to quantify the degree of uniformity in the region preceding and following the point of interest. In a similar way, the dissimilarity between these regions is also determined. The difference between the dissimilarity and uniformity values leads to a signal of which the valleys corresponds to probable segment boundaries. However, this signal is very noisy, and there are many small valleys. The number of these smaller valleys is reduced by application of a 'minmax' filter, which searches a fixed region (n_{mm}) around the point of interest to find the local maximum and minimum values. The difference between this maximum and minimum serves as the output of the filter at the position of the minimum. This filter is applied throughout the signal in non-overlapping regions. The filter output is a signal of which the peaks represents possible boundaries. Given that the 'minmax' filter region is usually very small and applied in non-overlapping intervals, many closely spaced peaks may still remain. Temporal peak masking is therefore applied in a subsequent step. Two peaks falling within a determined interval (t_d) of each other and which are above a chosen threshold (p_{min}) are identified, and the highest peak retained. The location of the highest peak is also shifted a small distance toward the eliminated smaller peak in proportion to their amplitudes.

2) Ten Bosch et al. [5]: This work uses the angle between the smoothed feature vectors just before and just after the point of interest to quantify the degree of local change. This is given by Equation 2, where f and g are the averages of the two feature vectors before and after the frame of interest F_j respectively.

$$LS(F_j) = arccos\frac{f.g}{(\parallel f \parallel \parallel g \parallel)^{\frac{1}{2}}} \tag{2}$$

12 MFCC and log energy together with their first and second derivatives are used as a 39-dimensional feature vector. All local maxima above a threshold (δ) are hypothesised as boundaries.

3) Estevan et al. [7]: This algorithm employs maximum margin clustering to detect points of change in a feature vector consisting of 12 MFCC coefficients, log energy and their first and second derivatives. A sliding window, N frames wide and centered about the frame of interest, sweeps through the signal. MMC clustering (using a RBF kernel) is applied to the frames within this window. The width of the RBF kernel, W, is estimated from a development set. The MMC clustering results in a cluster label for each frame within the window, and changes in these labels indicate possible segment boundaries. It was found that the best way to detect these changes is by using the Euclidean distance, as given by Equation 3, between the cluster labels and the cluster means. Let f be the cluster label of each frame within the sliding window, and g be the mean of the cluster throughout the signal. Peaks in the Euclidean distance will then indicate the segment boundaries.

$$LS(F_j) = [\sum_{l=1}^{T}(f_l - g_l)^2]^{\frac{1}{2}} \tag{3}$$

4) Sarkar et al. [6]: This method differs from the previous three by operating in the time domain rather than the frequency domain. The local score function used in this case is the average level crossing rate. The level crossing rate is closely related to the zero crossing rate, but with multiple additional levels other than $y = 0$, and among which the average crossing rate is taken. The levels can be distributed uniformly or non-uniformly. For this choice of local score, a boundary corresponds to a valley rather than a peak. As for some of the preceding algorithms, a threshold is required to prune out shallow valleys which lead to over-segmentation.

B. Algorithms based on minimising a distortion metric

Another approach to speech segmentation is to increase the uniformity within segments, i.e. to minimise some distortion metric within segments. In the work by Sharma et al. [3], the local score is the Euclidean distance applied to MFCC features. The distortion within a segment is calculated by Equation 5. This calculation employs the local score at frame j, given by Equation 3, and the mean of the local score from frame i to n, given by Equation 4. The segment stretching from frame i to frame n is denoted by $S_{i,n}$.

$$M_{i,n} = \frac{1}{n-i+1}\sum_{j=i}^{n} LS_j \tag{4}$$

$$distortion\_metric(S_{i,n}) = \sum_{j=i}^{n}(LS_j - M_{i,n})^2 \tag{5}$$

The overall distortion of the speech signal is a cumulative sum of the distortions of all the segments. The overall distortion can be minimised by applying a level-based DP algorithm to search for the optimal segmentation, assuming that the number of levels (segments) in the signal is known.

C. A proposed local score

For our formulation of the segmentation problem it is convenient if the local score lies between the values of 0 and 1. We propose the use of a normalised city block distance as shown in Equation 6,

$$LS(F_j) = \frac{\sum_{l=1}^{T}|f_l - g_l|}{\sum_{l=1}^{T}|f_l| + \sum_{l=1}^{T}|g_l|} \tag{6}$$

where f and g are the feature vectors before and after the frame of interest Fj. This proposed formulation of the local score will be compared with other candidates in the experimental evaluation. Note that parameterisations for f and g are not specified, allowing different feature vectors to be used during experimentation.

III. A DP-BASED SEGMENTATION ALGORITHM

Most segmentation algorithms based on maximum local-acoustic changes are prone to over-segmentation because they hypothesise more than one segment boundary at a point of acoustic change. This occurs due to the presence of multiple local maxima in the local score. To counteract this, the algorithms include various types of thresholds to eliminate such very short segments. Several examples

of such measures were given in Section II. These remedies are ad-hoc, however, and introduce additional parameters into the algorithm that require optimisation.

The algorithm we propose includes an explicit probabilistic model for the length of a segment. Segments that are either very short or very long are penalised by their associated low probability. The probability distribution of phoneme lengths for TIMIT can be estimated from the phonetic annotations, as illustrated in Figure 1. For illustrative purposes, the distribution is normalised with respect to its maximum probability.

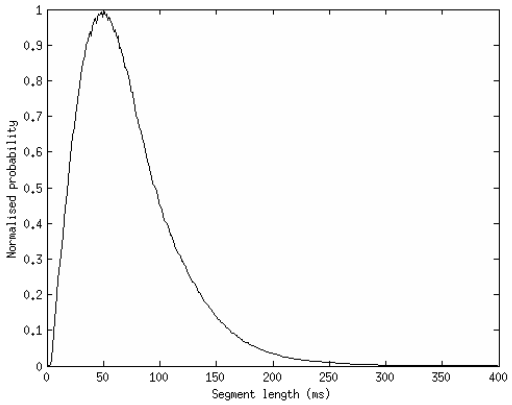


Fig. 1. Probability distribution of phoneme lengths in the TIMIT training set [8].

A. Segment probability

To gain some insight into the behaviour of local scores near segment boundaries, the local score in the close vicinity of phoneme boundaries, as given by the TIMIT annotations, is calculated and used to estimate a local score probability distribution given a boundary. A similar distribution is determined for the local score values taken far from boundaries, i.e. a local score probability distribution given that there is no boundary. Figure 2 shows these distributions, each normalised with respect to its maximum probability, for the local score calculated with Equation 6 when using FFT magnitudes as the feature vector. The distributions of the local score and the phoneme length can now be used to determine the probability of a boundary occurring at a specific frame in a speech signal.

Consider a signal consisting of N+1 frames. Now let the time of occurrence of each frame correspond to a state of a HMM as shown in Figure 3, where M is the maximum allowed number of frames per segment and S_0 is the time of occurrence of the first frame of the signal. The vertical dashed arrows between S_1 and S_1, S_2 and S_2, and between S_{N-1} and S_{N-1} indicate an expansion of the same HMM state.

When a state is visited by a path through the Markov model shown in Figure 3, a segment boundary is considered to occur at the corresponding speech frame. The transition and emission probabilities are calculated according to Equations 7 and 8 respectively, where SL refers to the segment length, LS to the local score, and SB to the occurrence of a segment boundary.

$$a_{i,j} = P(S_j|SL(S_j, S_i)) \tag{7}$$

$$b_j = P(SB|LS(S_j)) \tag{8}$$

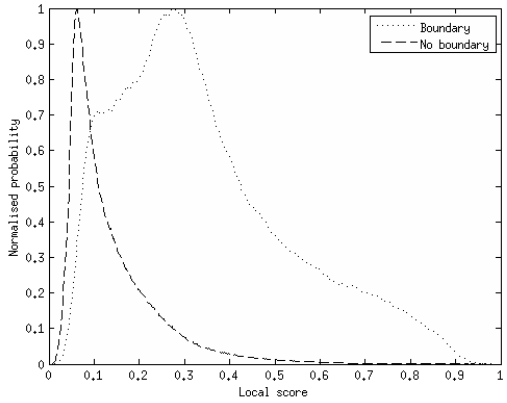


Fig. 2. Probability distribution estimates of local score values at, and away, from phoneme boundaries for Equation 6 applied to the FFT magnitudes.

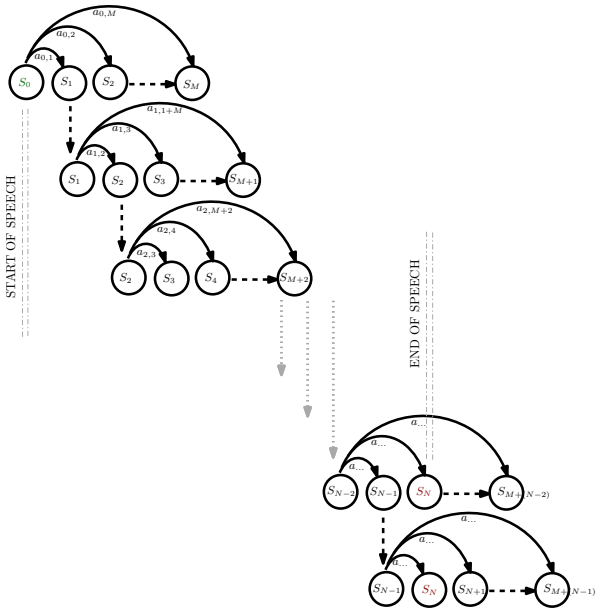


Fig. 3. DP-based segmentation cast as a HMM.

The segment length in Equation 7 is equal to the time step between two consecutive frames multiplied by the number of states separating the currently visited state and its parent state, as shown in Equation 9, where S_j is the current state, and S_i is the parent state.

$$SL(S_j, S_i) = (j - i) * step \tag{9}$$

Hence the transition probability is dependent only on the elapsed time between states. The emission probability at state Sj, as shown in Equation 8, is dependent on the local score $LS(Sj)$. To calculate the emission probability, Bayes rule is applied as shown in Equation 10, where $!SB$ refers to the absence of a segment boundary.

$$P(SB|LS(S_j)) = \frac{P(LS(S_j)|SB)P(SB)}{P(LS(S_j)|SB)P(SB) + P(LS(S_j)|!SB)P(!SB)} \tag{10}$$

The prior probability of a segment boundary can be estimated by dividing the number phoneme boundaries in the TIMIT annotations by the number of frames, as shown in Equation 11.

$$P(SB) = \frac{number\ of\ phoneme\ boundaries\ in\ TIMIT}{number\ of\ frames\ in\ TIMIT} \quad (11)$$

The probability that a boundary occurs at a particular frame can now be calculated by using Equations 9 and 10 in conjunction with estimates of the various probability distributions.

B. Optimal path

To find the globally optimal path from S_0 to S_N, all possible transitions shown in Figure 3 must be considered. This can be accomplished by using a DP algorithm. The states that were visited along the optimal path will identify the optimal segmentation. It is important to note that S_0 and S_N are always included in the path, and therefore the algorithm assumes that segment boundaries are always present at the start and the end of the speech signal. This means that any initial and final silence must be removed before applying the algorithm.

C. Normalising for path length

During the Viterbi decoding, many probabilities are multiplied together for any given path. When determining the optimal path, shorter paths (which contain fewer multiplications and thus longer segments) may be preferred, even when these have low associated emission and transition probabilities. We compensate for this effect by modifying the emission and transition probabilities as shown in Equations 12 and 13.

$$a_{i,j} = P(S_j|SL(S_j, S_i))^{SL(S_j,S_i)} \quad (12)$$

$$b_j = P(SB|LS(S_j))^{SL(S_j,S_i)} \quad (13)$$

These modifications normalise the path probability and remove the bias towards segmentations containing fewer segment boundaries.

IV. Assessing segmentation accuracy

In order to assess the quality of automatic-generated segmentations, we will determine how closely they correspond to the TIMIT phonetic segmentations. This provides a useful measure of segmentation accuracy. However it is dependent on the segmentation conventions used in TIMIT. For example, even though it is common practice for the /p/ to be segmented as a single phone in human annotations, the silence (closure) associated with the stop is considered a separate acoustic event in TIMIT. We found that the automatic segmentation algorithms could detect these closures quite accurately, and therefore decided to adhere to the original 61 TIMIT phone definitions without modification.

A. Comparing segmentations by DP

Comparing two sequences of segment boundary times can again be achieved by DP. We will proceed by first determining the best alignment between two sequences of boundary times. Then we will use this alignment to calculate a path cost. The alignment procedure uses a matrix of path costs as shown in Figure 4.

The first boundary in both sequences must coincide, and this corresponds to the bottom left cell of the matrix. Three alternative scenarios are then considered: (i) a hypothesised boundary $P_H(i)$ is paired (matches) a boundary $P_R(j)$ in the reference segmentation, (ii) a hypothesised boundary $P_H(i)$ is not paired with any boundary in the reference transcription (insertion) or (iii) there is no hypothesised boundary that can be paired with a boundary $P_R(j)$ in the reference transcription (deletion).

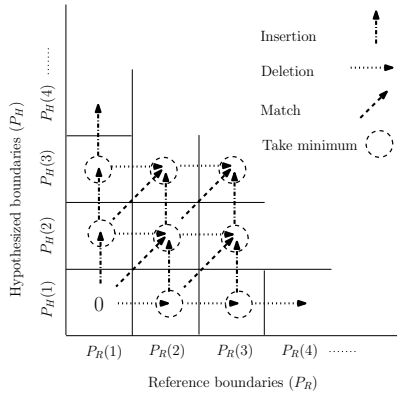


Fig. 4. Alignment matrix for segmentation scoring.

All possible paths from the bottom left to top right in the matrix shown in Figure 4 are computed recursively by dynamic programming. Starting from the bottom left of this matrix, each path can be extended upwards, to the right, or diagonally up and to the right, indicating an insertion, a deletion or a match between boundaries respectively. Each of these possibilities has a specific associated cost. When a reference boundary falls between two hypothesised boundaries, or vice versa, the cost is calculated by considering the distance to the nearest of the two boundaries. When paths meet, only the path with the lowest cost survives.

This procedure is applied iteratively, until all paths have reached the top right cell, which will then contain the final alignment cost between the two sequences. This cost reflects the difference between the hypothesised and reference sequences since it is the cumulative cost of every match, insertion, and deletion in the alignment. Furthermore, the cost has dimensions of time. By dividing it by the number of reference boundaries, the cost in seconds per reference boundary can be obtained. This is the average time difference between a hypothesised- and reference boundary and it will be used as a figure of merit in our later experiments. In addition, the number of insertions, deletions, and matches can be obtained by tracing back along the optimal path.

B. Fixed margin method

It appears to be standard practice in related research to consider a hypothesised and a reference segmentation boundary to be a match whenever they occur within 20ms of one another [4]. All non-matching boundaries are then either insertions or deletions. In order to make our results more directly compatible with those of others, this scoring framework has also been employed. An error measure termed the **average error** is defined, which is the average of the percentage insertions and deletions taken with respect to the number of reference boundaries in a speech signal. Furthermore, this interpretation of insertions, deletions and average error will be used.

V. Experimental setup

A. Data

Our experimental evaluations are based on the TIMIT database. The development set specified in [8] was used to optimise all

parameters, and the core test set defined in [8] was used exclusively for final testing. There is no speaker overlap between these two sets. The use of an explicit development set avoids biased results which would be obtained if the performance of the algorithm was measured on the same data used to optimise its hyperparameters. In the literature dealing with automatic segmentation, the separation of development and testing data was found not to be common. Leading and trailing silences were removed to account for the assumption that each utterance begins and ends with a segment boundary.

B. Feature vectors

We have chosen three feature vector configurations popular in literature on automatic speech segmentation for comparative experimentation.

1) FFT: Unprocessed 128-point FFT magnitudes
2) MFCC: 12 MFCCs and log energy
3) MFCC+Δ+$\Delta\Delta$: MFCC with appended first and second derivatives

By considering the local scores separately for the MFCCs, for the delta and for the acceleration features, it was found that a peak for the MFCCs or the acceleration components always coincides with a valley for the delta component, and vice versa. To account for this, the overall local score was calculated by averaging the local scores calculated for MFCCs and acceleration components, and the negative of the local score for the deltas.

C. The local score

Three local scores were investigated:

1) The cosine distance (C) shown in Equation 1,
2) The Euclidean distance (E) shown in Equation 3, and
3) The normalised city block distance (NCB) shown in Equation 6.

In our experiments, f and g were taken to be the averages of two frames to the left and two to the right of the inspected frame respectively. Depending on the local score, boundaries are expected to occur at either peaks or valleys (local maxima or minima) of the local score. Equation 10 is therefore only calculated at frames which coincide with local maxima or minima of the local score and a probability of 0 is assigned to all other frames.

D. The probability weights

As it stands, the DP segmentation algorithm will give equal weight to the transition and emission probabilities, due to the segment length and local score respectively. However, it may be beneficial to shift the balance more strongly towards one or the other. By multiplying the log values of the emission and transition probabilities by positive constants that sum to one, this shift in balance can be achieved, and will allow deletions to be traded for insertions and vice versa. Optimal performance on the development set was achieved by assigning a heavier weight to the emission probability (0.6–0.7) than to the transition probability (0.3–0.4). This gives a stronger preference to higher emission probabilities and leads to a reduction in insertions.

VI. Experimental results

A. Smoothing window size

In the following experiments a frame size of 16ms and a frame shift of 4ms were used. Before calculating the local score, each resulting MFCC and FFT value were smoothed by taking the average within a window centered on the feature vector in question. Subsequently, the average DP cost (Section IV-A) and the average error (Section IV-B) was calculated on the development set for different smoothing window sizes applied to different local score and feature vector combinations. Figures 5, 6, and 7 respectively show these results for the cases in which the cosine distance is applied to the FFT, the normalised city block distance is applied to MFCCs, and the Euclidean distance is applied to MFCC with first and second derivatives. For each configuration, all other parameters were optimised on the development set.

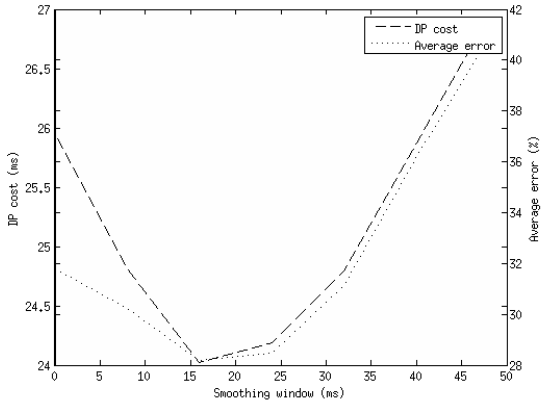


Fig. 5. DP cost (Section IV-A) and average error (Section IV-B) for different smoothing window sizes on the development set for the cosine distance applied to the FFT.

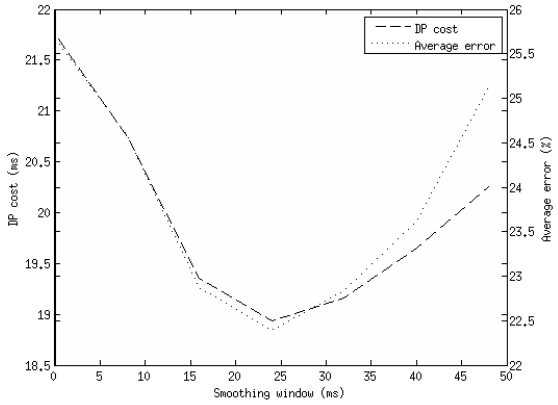


Fig. 6. DP cost (Section IV-A) and average error (Section IV-B) for different smoothing window sizes on the development set for the normalised city block distance applied to the MFCCs.

The results show that the optimal smoothing window sizes are similar for the FFT and MFCC parameterisations (16–24ms). A longer window (around 40ms) is required by the MFCC+Δ+$\Delta\Delta$ parameters, however. We believe that the introduction of first and second differentials introduces additional local maxima into the local score, which can lead to an increase in insertions. By lengthening the smoothing window, this is compensated for.

B. Choice of feature vector and local score

The performance of the DP segmentation algorithm when using the three different feature parameterisations and the three different local score formulations was compared experimentally, and results are shown in Table I. For each configuration, the length of the smoothing

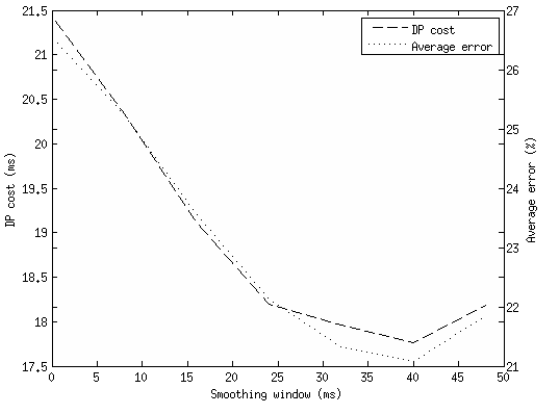


Fig. 7. DP cost (Section IV-A) and average error (Section IV-B) for different smoothing window sizes on the development set for the Euclidean distance applied to the MFCC with their first and second derivatives.

window as well as the probability weights are optimised on the development set, and segmentation accuracies determined on the test set. Both the DP path cost, in milliseconds per reference boundary, and the fixed margin average percentage error are shown.

TABLE I
DEVELOPMENT- AND TEST-SET PERFORMANCE OF THE DP SEGMENTATION ALGORITHM FOR THREE CHOICES OF FEATURE VECTOR AND FOR THE NORMALISED CITY BLOCK (NCB), EUCLIDEAN (E) AND COSINE (C) LOCAL SCORE (LS) FORMULATIONS.

Configuration		**DP Cost (ms)**		**%ERR**	
Feature	**LS**	**Dev**	**Test**	**Dev**	**Test**
FFT	NCB	18.62	18.42	20.21	19.80
MFCC	NCB	18.94	18.82	22.40	22.80
MFCC+Δ+ΔΔ	NCB	19.28	18.83	21.81	22.07
FFT	C	24.02	24.13	28.20	27.62
MFCC	C	19.01	18.98	22.53	22.85
MFCC+Δ+ΔΔ	C	18.94	18.72	21.56	21.72
FFT	E	28.06	27.93	33.27	33.13
MFCC	E	18.49	18.22	22.67	22.85
MFCC+Δ+ΔΔ	E	17.77	17.58	21.08	21.40

The normalised city block distance delivers the best overall performance. When applied to the MFCC+Δ+ΔΔ parameterisation, the Euclidean distance achieved similar performance. A configuration that stands out from the rest is the normalised city block distance applied to the FFT, which greatly outperforms all other combinations with the FFT feature vector. The FFT in general is the feature which is most sensitive to the remaining parameters, and was seen to be prone to over-segmentation. The FFT also has a higher dimensionality than the other parameterisation. It appears from the results that the normalised city block distance is most robust to this variation in dimensionality. Thus, the the normalised city block distance with a weighting leaning towards the emission probability (0.7) to reduce insertions gives very promising results. Among the feature parametrisations, the MFCC and the MFCC+Δ+ΔΔ are most competitive.

When comparing performance on the development and on the test sets, it is evident that the same patterns emerge from both. In the experiments that follow, each local score's best overall performing configuration will be used. These are the normalised city block distance for the FFT, the cosine distance for the MFCC+Δ+Δ, and the Euclidean distance for MFCC+Δ+Δ. These will henceforth be referred to as configuration C1, C2, and C3 respectively.

C. Silence removal

Many TIMIT sentences contain regions of silence in which temporal changes nevertheless occur. In order to avoid the hypotheses of segment boundaries in these regions, all boundaries were removed at frames when the ratio of the average energy content from 30ms before to 30ms after the frame in question, to the mean energy of the signal fall below a certain threshold. Different threshold values were investigated, and a typical result is shown in Figure 8. A threshold of 0.2% (i.e. a value of 0.002) delivered optimal performances for all configurations.

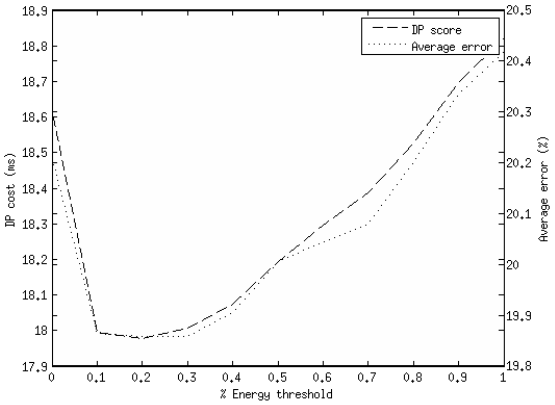


Fig. 8. DP cost (Section IV-A) and average error (Section IV-B) against % energy threshold on the development set for configuration C1.

D. Comparison with other segmentation algorithms

In the previous sections, an optimal configuration for the DP segmentation algorithm proposed in this paper is determined by experimentation. In this section we will benchmark the performance of this optimal configuration against two recent approaches to speech segmentation found in literature [4] [5]. Both approaches belong to the class of segmentation algorithms that rely on transient events in the acoustical information, as described in Section II-A. The method described in [4] claimed to achieve the same or better performance than many earlier approaches, while [5] is an algorithm with which the authors have had good prior experience.

Each method compensates for silences in its own way. The algorithm given in [5] scales the local score by the log frame energy to attenuate points of low energy, while the algorithm in [4] uses a similar approach to that proposed in this paper, but uses the average energy measured over the interval from -8ms to +30ms about the point of interest, and a threshold which is a multiple of the minimum energy for the signal. In the evaluation presented in the following, the parameters of each method were optimised on the development set.

Table II presents the DP cost in milliseconds per reference boundary, the percentage insertions and deletions with respect to the number of reference boundaries, and the average error for the optimised cases on the development set. The values shown for configurations C1, C2 and C3 are those achieved after silence removal.

When applying these parameter values to the core test set, the results shown in Table III are obtained.

TABLE II
PERFORMANCE COMPARISONS ON THE DEVELOPMENT SET AFTER SILENCE REMOVAL.

Method	DP Cost (ms)	% Ins	% Del	%ERR
DP (C1)	17.98	15.56	24.16	19.86
DP (C2)	18.12	15.28	26.97	21.13
DP (C3)	17.04	18.15	23.05	20.60
Räsänen	18.91	17.92	26.99	22.46
ten Bosch	25.07	26.19	27.37	26.78

TABLE III
PERFORMANCE COMPARISONS ON THE CORE TEST SET AFTER SILENCE REMOVAL.

Method	DP Cost (ms)	% Ins	% Del	%ERR
DP (C1)	17.92	14.49	24.53	19.51
DP (C2)	18.23	14.80	28.04	21.42
DP (C3)	17.13	17.14	24.93	21.03
Räsänen	19.40	17.18	28.19	22.68
ten Bosch	25.17	25.36	28.28	26.82

For illustrative purposes, the segmentations produced by the three algorithms for the same sentence, dr6-fbch0-sa1, are shown in Figures 9, 10, and 11, where configuration C3 was used for the DP algorithm. Each figure shows the first two seconds of the sentence as well as the TIMIT phone boundaries. The dashed vertical lines show the hypothesised boundaries, and the solid vertical lines show the reference boundaries.

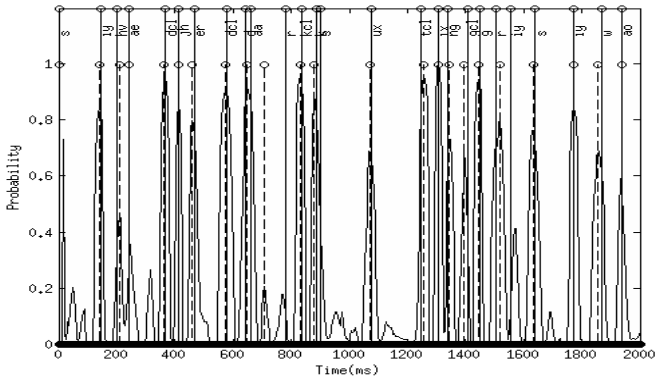


Fig. 9. Segmentation results for the DP algorithm on dr6-fbch0-sa1.

The vertical axis in Figure 9 for the DP algorithm shows the emission probabilities. Unlike the other two approaches, there is no threshold under which boundaries are ignored. Thus, even when the local score results in a low emission probability, a boundary can be hypothesised if the transition probability is high. This is clear, for example, at the boundary that is hypothesised at the 'aa' phoneme. The converse may also be true, i.e. even when the emission probability is high, a segment boundary may be suppressed by a low transition probability, as illustrated at the second 'iy'.

Figure 10 shows the output of the 'minmax' filter described in Section II-A of the Räsänen algorithm. Notice that all peaks falling within 32ms of each other have been combined by temporal peak masking, and that the threshold in this case is 0.07, below which all peaks are ignored. These parameter values were determined to be optimal for the development set.

The local score of ten Bosch's algorithm has been multiplied by the log energy to reduce the insertion of boundaries in regions of

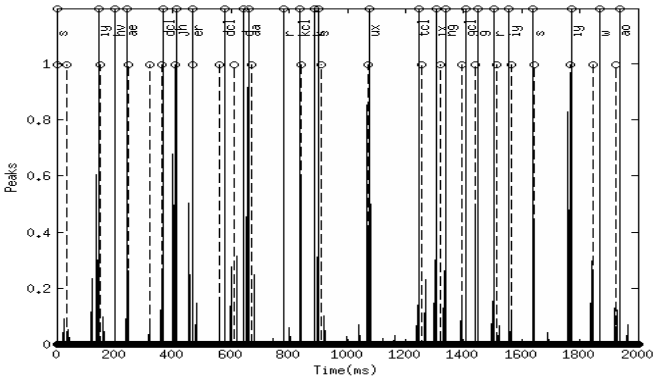


Fig. 10. Segmentation results for the Räsänen algorithm on dr6-fbch0-sa1.

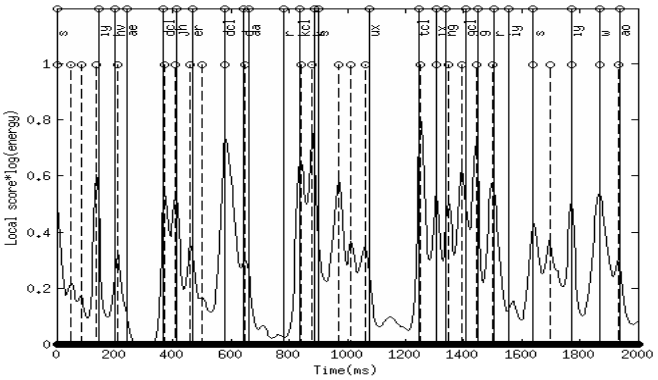


Fig. 11. Segmentation results for the ten Bosch algorithm on dr6-fbch0-sa1.

silence, which are characterised by very low energy. Unfortunately this also results in the introduction of unwanted small peaks when the log energy increases while the local score decreases, or when the energy decreases while the local score increases. Because the segment boundaries usually coincide with the peaks of the local score, these newly added peaks lead to insertions as shown, for example, in Figure 11 at 'er' and at each 's'. This leads to over segmentation, which is clear when looking at the higher percentage insertions in Table III. From the development set it was found that the optimal threshold for the ten Bosch algorithm is 0.13.

E. Combined methods

By inspection of the segmentation results produced by the DP algorithm, it was found that there regularly are small emission probability peaks present between the boundaries of very long segments. When these peaks coincide with high probability segment lengths, as determined by the segment length distribution, boundaries are hypothesised at these locations, resulting in unwanted insertions. With some experimentation it was found that better results can be obtained by applying a threshold to the emission probability (Equation 8) before searching for the optimal path by DP. All probabilities above the threshold are unchanged, and the probabilities below the threshold are reduced to 0. A variety of threshold values were investigated on the development set for each of the chosen three DP configurations, with all other parameters fixed at their previously found optimal values. Figures 12 and 13 show the resulting effect on the DP cost and

on the average error for configuration C1. By inspecting the average error graph, there is a point at which the reduction in insertions is greater than the rise in deletions. However, the average error can only be reduced to a certain point, after which the hypothesised and reference boundaries rapidly become misaligned. This is indicated at the point of DP cost increase. The DP cost is therefore the best way to determine the optimal threshold.

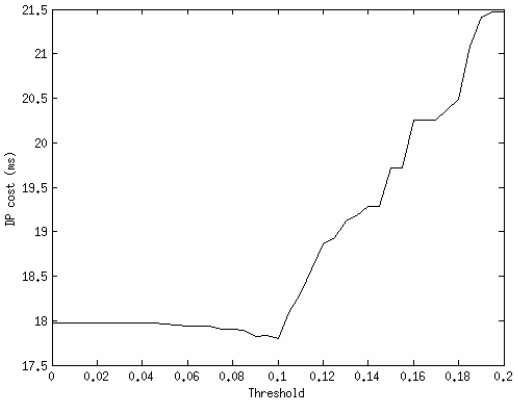


Fig. 12. DP cost (Section IV-A) against emission probability threshold on the development set for configuration C1.

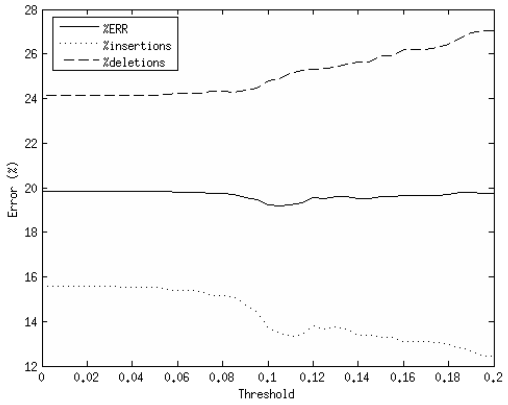


Fig. 13. Average error (Section IV-B) against emission probability threshold on the development set for configuration C1.

It was found that thresholds of 0.1, 0.5, and 0.1 lead to optimal performance on the development set for configurations C1, C2 and C3 respectively. When applied to the core test set, this leads to the results in Table IV.

TABLE IV
METHOD COMPARISONS, AFTER EMISSION PROBABILITY THRESHOLD WERE APPLIED, ON THE CORE TEST SET.

Method	DP Cost	% Ins	% Del	%ERR
DP (C1)	17.68	12.83	24.96	18.89
DP (C2)	18.08	13.91	28.44	21.17
DP (C3)	16.99	16.65	25.02	20.83
Räsänen	19.40	17.18	28.19	22.68
ten Bosch	25.17	25.36	28.28	26.82

By comparing the results in Tables III and IV, improvements in performance for all three configurations are seen. Two key values that stand out from Table IV are the small DP cost obtained by configuration C3, and the small average error obtained by configuration C1. The overall best, and most consistent configuration thus far, is configuration C1, which has the normalised city block distance and the FFT.

VII. SUMMARY AND CONCLUSION

We have proposed an algorithm based on the principle of dynamic programming for the automatic segmentation of continuous speech into phoneme-like units. A measure of the local dissimilarity between feature vectors is combined with a statistical description of the expected segment lengths within the dynamic programming framework in order to determine the optimal locations of segment boundaries within the speech utterance. We find that this approach leads to performance improvements relative to two alternative methods drawn from the literature. Analysis of the strengths of the individual techniques revealed that further improvements can be obtained by a hybrid approach employing aspects of each. We conclude that the use of dynamic programming as a basis for speech segmentation is a successful approach. In future work we plan to analyse the occurrence of insertion and deletion errors more carefully with respect to the type of phoneme within which they occur, as well as the role of context in the placement of segment boundaries. The effectiveness of our DP-based segmentation will also be tested on other languages using the distributions created from TIMIT to see how universal the segment boundary behaviour is. Furthermore, we will investigate the sensitivity of the segmentation algorithms to parameter changes, and the effect of increased parameters.

REFERENCES

[1] Y. jun Kim and A. Conkie, "Automatic segmentation combining an hmm-based approach and spectral boundary correction," in *Proceedings of the International Conference on Spoken Language Processing, ICSLP*, 2002, pp. 145–148.

[2] T. Svendsen and F. Soong, "On the automatic segmentation of speech signals," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP*, vol. 12, apr 1987, pp. 77 – 80.

[3] M. Sharma and R. Mammone, "'blind' speech segmentation: automatic segmentation of speech without linguistic knowledge," in *Proceedings of the Fourth International Conference on Spoken Language Processing, ICSLP*, vol. 2, oct 1996, pp. 1237 –1240.

[4] Okko Räsänen, U. K. Laine, and T. Altosaar, "Blind segmentation of speech using non-linear filtering methods," in *Ipsic I. (Ed.): Speech Technologies*. InTech Publishing, 2011, pp. 105 –124.

[5] L. ten Bosch and B. Cranen, "A computational model for unsupervised word discovery," in *Order A Journal On The Theory Of Ordered Sets And Its Applications*, 2007, pp. 1 – 4.

[6] A. Sarkar and T. Sreenivas, "Automatic speech segmentation using average level crossing rate information," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP*, vol. 1, 2005, pp. 397 – 400.

[7] Y. P. Estevan, V. Wan, and O. Scharenborg, "Finding Maximum Margin Segments in Speech," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP*, 2007, pp. 937 –940.

[8] A. K. Halberstadt, "Heterogeneous Acoustic Measurements and Multiple Classifiers for Speech Recognition ," Ph.D. dissertation, Massachusetts Institute of Technology, MIT, 1998.

Robust single image noise estimation from approximate local statistics

Yuko Roodt, Wimpie Clarke
Resolution Circle
University of Johannesburg
South Africa
Email: yukoroodt@gmail.com,
willemc@uj.ac.za

Philip E. Robinson
HyperVision Research Laboratory
School of Electrical Engineering
University of Johannesburg
South Africa
Email: philipr@uj.ac.za

André Nel
School of Mechanical Engineering
University of Johannesburg
South Africa
Email: andren@uj.ac.za

***Abstract*—A novel method for estimating the variance and standard deviation of the additive white Gaussian noise contained in an image will be presented. Only a single image is used to estimate the noise properties. Local image outliers are discarded, this allows us to separate the additive zero mean white Gaussian noise contained in a noisy image from the original image structure. Local variance estimates can then be calculated from the extracted noise. These local variance estimates are weak and can be influenced by misclassified image information. Robust statistics are then used to fuse the weak local variance estimates to obtain a robust global noise variance estimate. This method of estimating the noise properties is computationally efficient and provides reliable estimation results in synthetic and real-world imagery. The accuracy and processing complexity of the proposed algorithm will be compared against the current state-of-the-art noise estimators.**

I. Introduction

Images are often corrupted by noise which could have been introduced during the transmission or acquisition phase of the imaging process [1]. Poor imaging sensors and low lighting conditions can increase the presence of noise. Consider the noisy image formation model provided in Eq. 1:

$$I(x,y) = f(x,y) + n(x,y). \tag{1}$$

where I is the observed image affected by noise, f is the uncontaminated image and n is uniform distributed white Gaussian noise. The input image is contaminated by additive white Gaussian noise with an unknown standard deviation.

An estimate of the level of noise is required by many image processing and computer vision algorithms. Noise removal[2] and de-blurring algorithms[3] can benefit from accurate noise estimates. Other image processing task such as edge and feature detection can be improved by selecting optimal thresholds to limit the impact of noise [4].

Noise can be estimated from multiple images in stationary sequences, this is considered to be an over-constrained problem. The images are fused together to remove the noise and to approximate the original clean image [5]. A temporal mean is calculated, multiple pixel samples are combined over time to estimate the original pixel value.

The estimation of the noise variance from a single image is however an under-constrained problem [6]. If a smoothness assumption is made on the local image structure, neighbouring image samples can be utilized to approximate a better or more robust estimate of the current pixel. A spatial mean is calculated removing some of the noise present in the image. This has the drawback of losing some high frequency image detail that does not hold to the smoothness assumption.

Aja-Fernandez et al. used local statistics to estimate the variance of the introduced noise. A simple method was proposed that calculates the Mode of the distribution of local variances estimates calculated in an image [7]. This method will be referred to as Mode09 and fails if the image does not contain a sufficient amount of low-variability areas.

A fast noise variance estimator was proposed by Immerkaer, referred to in the text as Immerkaer96 [8]. This method only requires that a 3×3 mask be processed over the image, the results are then summed and multiplied by a constant to obtain the noise variance. The mask is separable and only 14 integer operations per pixel is required, making this the fastest algorithm. Immerkaer notes that in highly textured regions in an image, lines will be perceived as noise. This will result in unusable noise variance estimates in images dominated by high frequency information.

TaiYang08 was designed to exclude image detail and structure from the noise variance estimation process [9]. Image detail is first detected through edge detection, these detected regions are then excluded from the variance estimation calculation. The edge detection threshold parameter is adaptively tuned to the image content. Tai et al. state that the algorithm performs well over a large range of noise variance levels.

A detailed break down of the proposed noise estimation algorithm will be provided in Section 2, each of the processing steps will be discussed in detail. In Section 3 we will describe the experimental setup used for testing and in Section 4 we will provide estimation accuracy and performance results of the different noise estimators, as-well as a comparative analysis of the results.

September 18, 2012

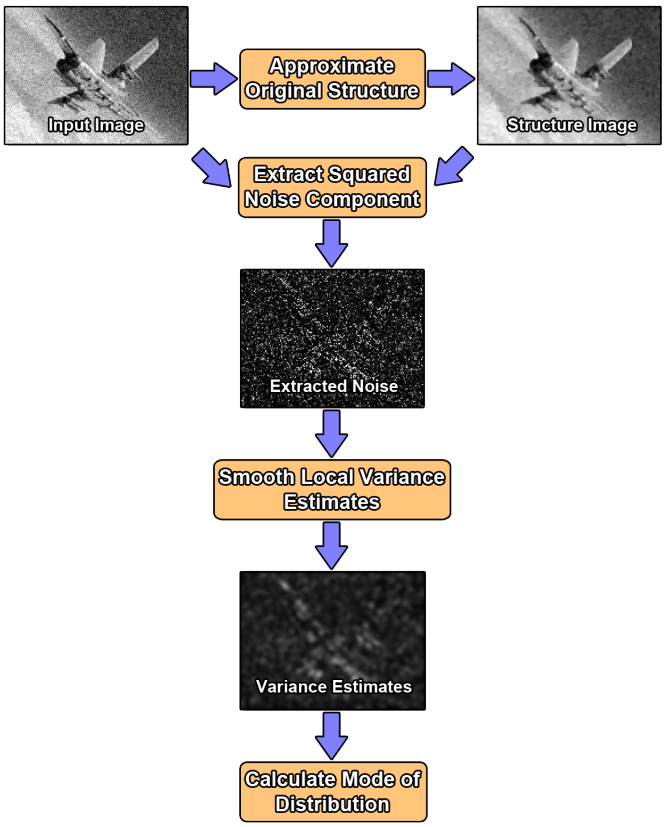


Fig. 1. Overview of the single image noise variance estimator

II. Noise Estimation

An overview of the steps involved in determining the noise variance from a single image is provided in Fig. 1. Two small median filters are used to remove outliers from the noisy image. This enables us to approximate the original image structure and preserve detail. The structure image and original input image can then be used to separate the noise component from the image data. This is achieved by locating differences between the noisy image and the structure image. Since the structure image is only an approximate reconstruction of the original image, some image data can be classified as noise. To limit the misclassified image data's contribution to the noise variance estimate. The statistical mode of all the local variance estimates are calculated. This provides a robust statistical measure of the noise variance located in an image. Each step will now be discussed in more detail.

A. Approximate original structure

Many noise estimation algorithms attempt to remove noise from the original image by convolving the image with a Gaussian kernel. Convolution with small filtering kernels are not resilient against image outliers. The noise values have a large influence on the neighbouring pixels which reduces the accuracy of the estimation. The median filter is robust against outliers, assuming there is structure associated with the content of the image. Noise will be recognized as potential outliers.

The 2D median filter is edge preserving but not corner preserving. The noisy image should be filtered with the smallest possible median filter to preserve as much image detail as possible [10]. Large median filters have a tendency to remove noise as well as high frequency information. Two small 1D median filter were used, first to remove horizontal artefacts and secondly to remove vertical artefacts. The 3×1 and a 1×3 median filter preserved edges as well as most corners. The resulting image is an approximation of the original image before noise was introduced.

B. Extract squared noise components

Now that we have obtained an approximation of the original image. We are able to separate the noise component from the noisy image. The noise component for every pixel can be extracted from the noisy input image and the structure image as defined by Eq. 2:

$$noise\_component = (structure\_image - input\_image)^2 \tag{2}$$

where the $input\_image$ is the original input image contaminated by noise and the $structure\_image$ is an approximation of the original image. At every position in the image the structure image is subtracted from the noise image. The result is then squared to produce rudimentary local noise variance estimates.

C. Smooth local variance estimates

The local variance estimates could still be substantially affected by inherent image detail. A smoothing operation is performed to force the variance estimates to be more locally coherent. Empirical selection refers here to the process of experimentation with different parameters and observing the resulting variance estimation errors. The Gaussian kernel with $\sigma = 2$ was empirically selected as a good smoothing function for the variance estimates. The separability property of the Gaussian function was used to optimize the convolution process [11]. The local variance estimates were then filtered with a 13×1 filter and then by a 1×13 filter. This would have had the same effect as filtering with a 13×13 filter, but reduces processing resources.

D. Calculate Mode of distribution

We will now try to find the variance estimate or variance estimate range that occurs most often in the local variance estimates. The statistical Mode operation was used [12]. A histogram was generated from the local variance estimates, each local variance estimate was placed in a list of buckets. Its position in this list was determined by its noise variance value. The first bucket would contain all the smallest local variance estimates and the last bucket would contain all the largest variance estimates. The list size was determine using Eq. 3:

$$bucket\_count = floor\left(\frac{estimate\_count}{bucket\_size}\right) \tag{3}$$

where $estimate\_count$ is the total number of local variance estimates, $bucket\_size$ is the minimum number of values that would have been placed in each bucket if the values were uniformly distributed. A $bucket\_size$ of 10 was selected to provide accurate and reliable results. Selecting small bucket sizes will result in more accurate variance estimates. If the bucket size is selected to be to small, duplicate maximum buckets could occur resulting in unstable noise variance estimates. The next step is to find the bucket containing the most variance estimates. This estimate is the local variance value that occurred the most frequently in the image. The middle of the variance estimate range corresponding to the largest bucket is selected as the variance estimate of the noise in the image.

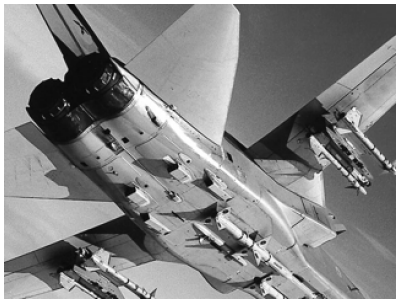
III. Experimental Setup

The accuracy of the noise variance estimators was determined by adding white Gaussian noise of a specific variance to an image. The noise estimator then had to estimate the amount of noise that was added. Since small amounts of noise exist even in high quality photos, a ground truth dataset could not be established. To limit the amount of noise in the test dataset. Large natural photos were down sampled to reduce the affect of noise. The noise would be reduced due to the averaging process. Since all noise could not be removed from the natural photo an additional synthetically generated dataset was also used for testing. These images are free of noise but do not have the complexity of natural photos.

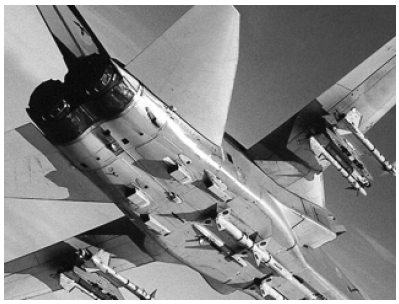
Both the natural and the synthetic datasets comprised of images that contain a large amount of high frequency information as well as images dominated by low frequency information. Some of the tested noise estimation algorithms perform better when there were an abundance of uniform areas in the image. Others would perform better at estimating small amounts of noise accurately, but would underestimate the noise when large amounts of noise was introduced. The opposite is also true, many noise estimation algorithms can determine the variance of large amounts of noise but would overestimate small amounts. The proposed noise estimation algorithm was designed to perform well over the whole range of noise levels.

The reliability of each estimator was determined by testing over a range of noise variance levels. The noise levels added to the original image range from low to severe levels of noise. Examples of the noise variance range used for testing can be seen in Fig. 2, the image pixel values were scaled to the range of $[0..1]$. Each candidate noise estimator was given the opportunity to estimate the noise variance contained in the resulting noisy image. This process was repeated 10 times and the average variance determined by the estimators was logged. At each iteration a new set of random noise variables of a specific variance were generated. This was then added to the original image.

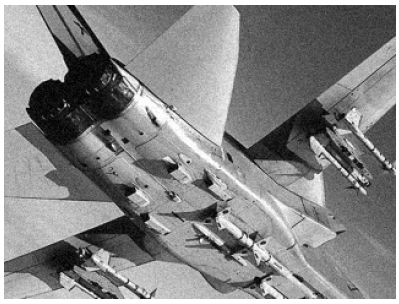
The estimation error percentage was calculated using Eq. 4. The estimated variance and ground truth variance for the test is converted to a standard deviation. From the standard deviations a similarity ratio is calculated.



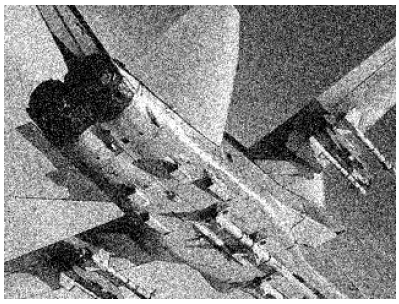
(a) Original image



(b) Noise Variance=0.00008



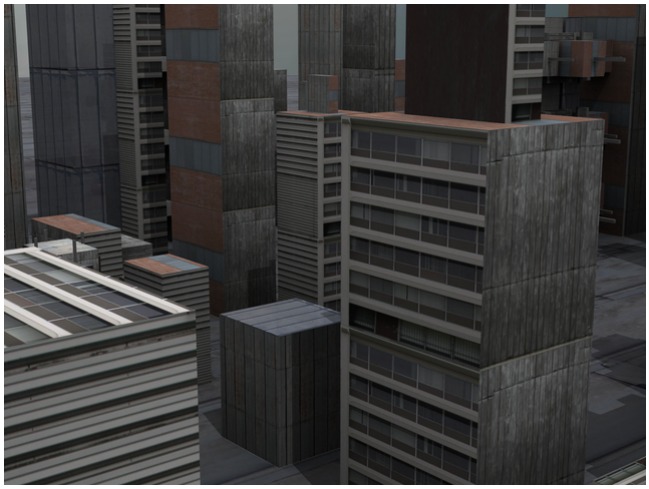
(c) Noise Variance=0.00128



(d) Noise Variance=0.02048

Fig. 2. Images containing different levels of additive Gaussian noise

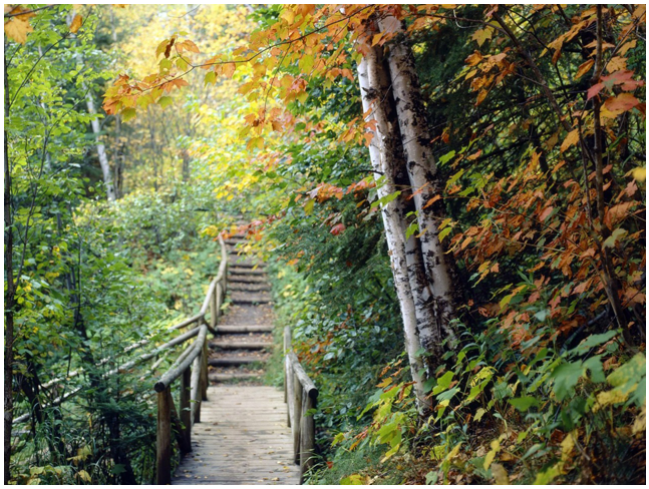
(a) Grove



(b) Urban

Fig. 3. Synthetic images from Middlebury stereo dataset [13].

(a) Aircraft



(b) Forest

Fig. 4. Natural image dataset

$$estimation\_error = \left| 1.0 - \frac{\sqrt{estimated\_variance}}{\sqrt{true\_variance}} \right| \times 100.0 \quad (4)$$

A. Synthetic dataset

Synthetic images from the Middlebury College stereo optical flow dataset were used. These images are well known in the image processing community. The first frames from the Grove and Urban datasets were used to evaluate the noise estimators and can be seen in Fig. 3. The Grove image contains an abundance of high frequency information while the Urban image has more uniform regions. An accurate estimation of the noise variance should be obtainable with these noise free images. High frequency information can adversely influence the noise estimation accuracy and image detail is often misclassified as noise. Large amounts of high frequency image data can make it hard to differentiate between noise and information.

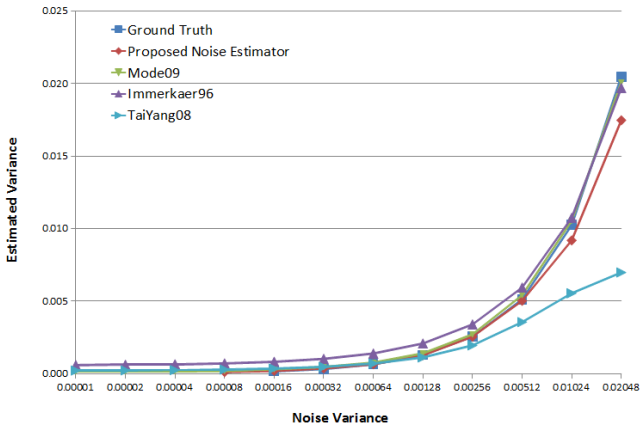
B. Natural image dataset

For the natural image test, two photos where selected containing high and low frequency information. They can be observed in Fig. 4. Exact estimation of the noise variance is difficult. The test images already contain small amounts of noise introduced during the image formation process. For this test we will assume that the small amount of noise already present in the image is negligible. Since we do not know the

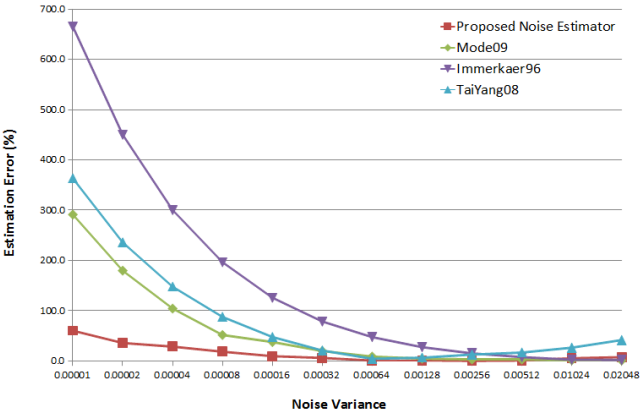
noise characteristics of the test images the noise estimator will overestimate the noise slightly. This is to compensate for the presence of existing noise.

IV. Experimental Results

A. Synthetic tests



(a) Noise variance estimates for various degrees of additive gaussian noise

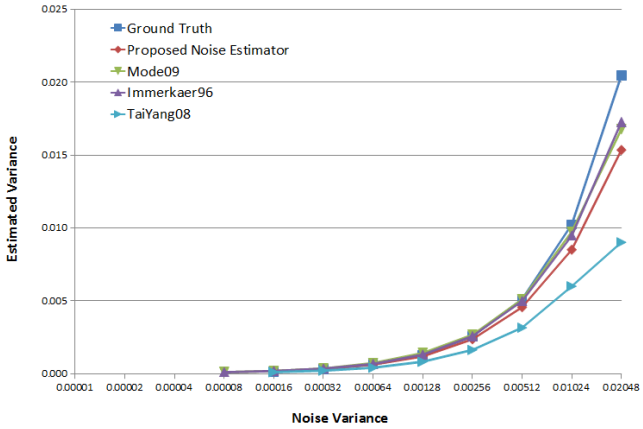


(b) Estimation error for tested noise variance

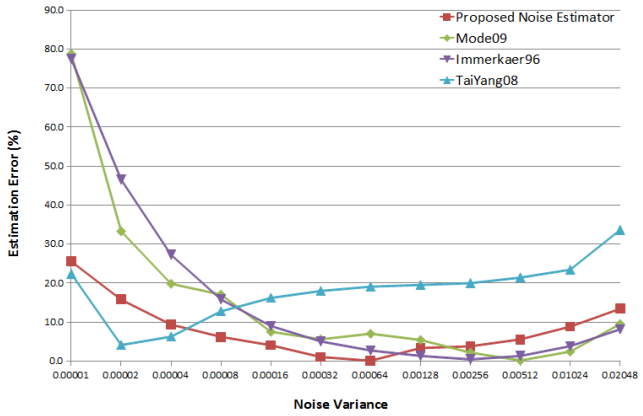
Fig. 5. Grove noise variance estimation results

In Fig. 5 it can be observed that in images with some uniform regions our proposed estimator excels. It can accurately estimate the variance of the noise over the whole spectrum. The other methods perform poorly in low noise situations and overestimate the variance. This can be attributed to the poor bit resolution of the images, which makes it difficult to distinguish between small intensity changes.

In the Urban test, the estimation methods did not overestimate the noise variance as severely as in the Grove example. This can be seen in Fig. 6. Even though the geometry represented in the image is simple, they are textured with high resolution material textures. Due to the dense pixel



(a) Noise variance estimates for various degrees of additive gaussian noise



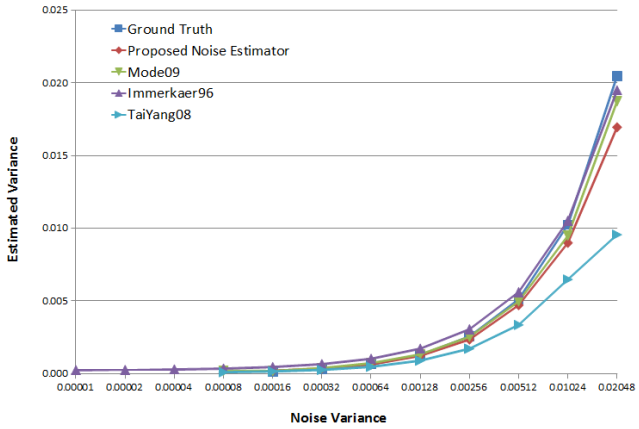
(b) Estimation error for tested noise variance

Fig. 6. Urban noise variance estimation results

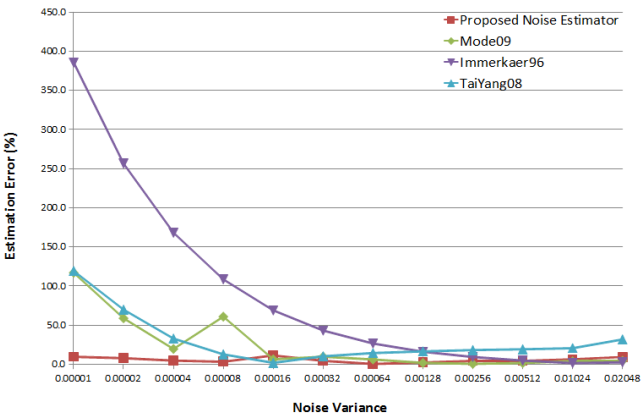
information, noise classification is more difficult. The proposed method provided the highest estimation accuracy. Some algorithms provided more accurate result for single estimation tasks but performed poorly in others. The Mode09 gave very erratic results due to the bucket size used for histogram mode calculations. This made tuning over the whole testing spectrum difficult. The proposed noise estimator provided consistent results over all the synthetic tests.

B. Natural image test

Immerkaer96 did not do so well in the natural image tests and was outperformed by almost all competing methods except in the high noise variance cases. The proposed method achieved high accuracy over the whole variance range. Especially in the low noise scenarios which occur regularly in low cost sensors and low lighting environments. The high noise test is a bit extreme and does not occur regularly in visible light cameras.

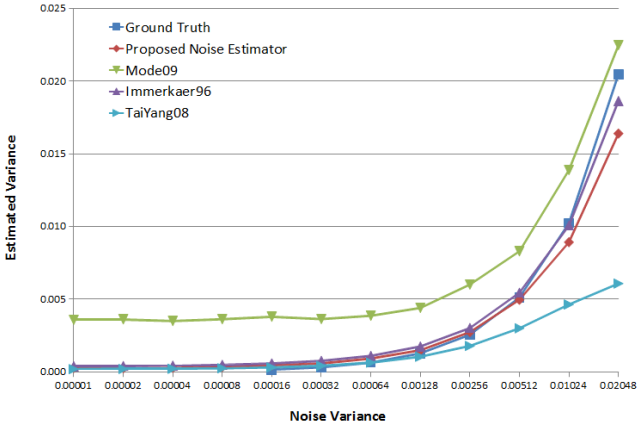


(a) Noise variance estimates for various degrees of additive Gaussian noise

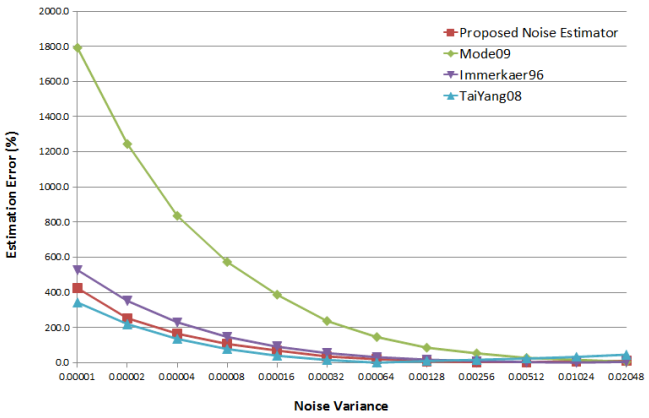


(b) Estimation error for tested noise variance

Fig. 7. Aircraft noise variance estimation results



(a) Noise variance estimates for various degrees of additive gaussian noise



(b) Estimation error for tested noise variance

Fig. 8. Forest noise variance estimation results

The estimation accuracy dropped substantially in the Forest test as seen in Fig. 8. The Forest image does not contain large uniform areas such as in the previous test image. Mode09 did not provide consistent results between the two tests sets. In the low frequency test it performed well, accuracy was reduced in the high frequency test. The other methods provided reasonable accuracy except TaiYang08 which performed poorly in the high noise situations.

C. *Normalized processing complexity*

The amount of noise in an image does not affect the processing complexity of any of the tested algorithms. This means that over the whole tested noise variance range each algorithm took approximately the same duration to complete. The total processing time required by each method to process the Grove, Urban, Aircraft and Forest dataset images were normalized to obtain an estimate of the processing complexity compared to each other. An algorithm with higher processing complexity will take longer to process the same image, processed by a lower processing complexity algorithm. As can be seen in

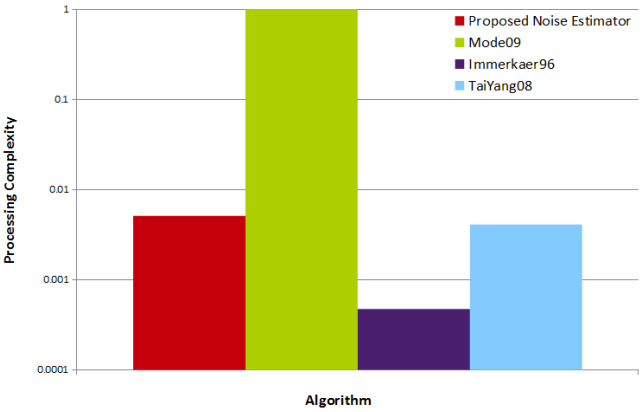


Fig. 9. Normalized processing complexity of noise estimation algorithms

Fig. 9 the proposed method had a relatively low processing complexity for the acquired accuracy. The Mode09 required the most processing resources, while Immerkaer96 was the fastest but with reduced accuracy.

V. Conclusion

A robust and computationally efficient single image noise estimation algorithm was presented. This method removed noisy artefacts from the observed image and calculated an approximation of the original image by rejecting image structure outliers. The reconstructed image is then used to separate the noise component from the observed noisy image. Weak local noise variances estimates were then calculated and combined to produce a reliable global noise variance estimate. This reduced the influence of misclassified image information on the noise variance estimation process. The proposed method produced superior estimation results compared to the current state-of-the-art noise estimation algorithms. It also produced consistent result over a range of synthetic and natural images, containing high and low frequency information. Reliable results over a large range of noise levels were also obtained, ranging from low to extreme imaging conditions.

References

[1] Gaoyong L., "Fast Wavelet Image Denoising Based on Local Variance and Edge Analysis", International Journal of Electrical and Computer Engineering, Vol 1, No 6, 2006.

[2] Rajashekar U. and Simoncelli E.P., "Multiscale Denoising of Photographic Images", Academic Press, The Essential Guide to Image Processing, 2nd edition, Chap 11, pp 241 - 261, 2009.

[3] Su L. and Li F., "Deconvolution of defocused image with multivariate local polynomial regression and iterative wiener filtering in DWT domain", Mathematical Problems in Engineering, 2010.

[4] Canny J., "A Computational Approach To Edge Detection", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 8, No 6, pp 679 - 698, 1986.

[5] Healey G.E. and Kondepudy R., "Radiometric CCD camera calibration and noise estimation", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 16, No 3, pp 267 - 276, 1994.

[6] Liu C., Szeliski R., Kang S.B., Zitnick C.L. and Freeman W.T., "Automatic estimation and removal of noise from a single image", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 30, No 2, pp 299 - 314, 2008.

[7] Aja-Fernandez S., Vegas-Sanchez-Ferrero G., Martin-Fernandez M. and Alberola-Lopez C., "Automatic noise estimation in images using local statistics. Additive and multiplicative cases", Image and Vision Computing, Vol 27, No 6, pp 756 - 770, 2009.

[8] Immerkaer J., "Fast Noise Variance Estimation", Computer Vision and Image Understanding, Vol 64, No 2, pp 300 - 302, 1996.

[9] Shen-Chuan T. and Shih-Ming Y., "A Fast Method For Image Noise Estimation Using Laplacian Operator and Adaptive Edge Detection", 3rd International Symposium on Communications, Control and Signal Processing, pp 1077 - 1081, 2008.

[10] Perreault S. and Hebert P., "Median Filtering in Constant Time", IEEE Transactions on Image Processing, Vol 16, No 9, pp 2389 - 2394, 2007.

[11] Yip H.M., Ahmad I. and Pong T.C., "An Efficient Parallel Algorithm for Computing the Gaussian Convolution of Multi-dimensional Image Data", Journal of Supercomputing, Vol 14, pp 233 - 255, 1999.

[12] von Hippel P.T., "Mean, Median, and Skew: Correcting a Textbook Rule", Journal of Statistics Education, Vol 13, No 2, 2005.

[13] Baker S., Scharstein D., Lewis J.P., Roth S., Black M.J. and Szeliski R., "A Database and Evaluation Methodology for Optical Flow", International Journal of Computer Vision, Vol 92, No 1, pp 1 - 31, 2011.

Gaussian blur identification using scale-space theory

Philip Robinson, Yuko Roodt and Andre Nel
Faculty of Engineering and Built Environment
University of Johannesburg
South Africa
philipr@uj.ac.za, yukoroodt@gmail.com, andren@uj.ac.za

***Abstract*—Image deblurring algorithms generally assume that the nature of the blurring function that degraded an image is known before an image can be deblurred. In the case of most naturally captured images the strength of the blur present in the image is not known. This paper proposes a method to identify the standard deviation of a Gaussian blur that has been applied to a single image with no *a priori* information about the conditions under which the image was captured. This simple method makes use of a property of the Gaussian function and the Gaussian scale space representation of an image to identify the amount of blur. This is in contrast to the majority of statistical techniques that require extensive training or complex statistical models of the blur for identification.**

Keywords-Gaussian blur, blur identification, blur estimation, scale space.

I. INTRODUCTION

In almost all vision systems, biological or mechanical, the phenomenon of blur can be observed. Blur manifests itself as a degradation of spatial detail or high frequency visual information. This results in a reduction of edge sharpness and loss of the finer detail. There are many causes of blur but the most fundamental is the diffraction limit of a vision system that contains an aperture [1]. Some other causes of blur are defocus, motion during exposure, atmospheric turbulence and upscaling of images [1, 2, 3].

Blurring is a distortion of an image that reduces the amount of information contained in that image. While it is impossible to build a physical system that can capture arbitrarily sharp images it is mathematically possible to reconstruct a portion of the lost information [4]. This process is called image deconvolution or image restoration and is essentially an inverse filtering process. The blurring effect is modelled as a convolution of the original image with a blurring kernel or Point Spread Function (PSF) with some additive white Gaussian noise as shown in the following equation [5].

$$i(x,y) = f(x,y) * h(x,y) + n(x,y), \qquad (1)$$

Where *i(x,y)* is the distorted 2D image with the 2 dimensions denoted by *x* and *y*, *f(x,y)* is the undistorted image, *h(x,y)* is the blurring function PSF which is convolved with the input image and *n(x,y)* is the additive white Gaussian noise present in the scene [5].

Usually it is assumed that the PSF of the blurring distortion is known. An operation is then performed that is the inverse of that distortion to attempt to undo that distortion [4, 5].

The image deconvolution problem has been explored quite thoroughly in the literature. The basic approaches of inverse filtering, least squares filtering and iterative filtering can be found in most image processing textbooks such as [5, 6]. More modern methods have also been discussed in [7, 8, 9] to name a few.

When the parameters of the PSF of the blurring function is not known and has to be estimated from the input image the problem becomes known as a blind deconvolution problem [5]. Blur identification techniques need to be employed to estimate the nature of the blur in the observed image. Numerous approaches to this problem have been proposed in the literature. The vast majority of approaches make use of image statistics to provide an estimate of the blur. In [10] a maximum likelihood estimation technique is used, [3] uses an autoregressive–moving-average (ARMA) process and [11] uses a regularization approach.

Non-statistical approaches also exist, for instance in [12] the original unblurred image is estimated and used to estimate what blur was applied to result in the degraded image. The approach that most resembles ours is a parametric approach where the blur is considered to conform to an assumed blur model with a single parameter. A search space of possible blur parameters is traversed and the input image is deconvolved with each parameter value. A sharpness metric is used to determine which parameter results in the sharpest output image. In this case the sharpness metric used was kurtosis [13].

There are a variety of types of blur found in images but we will focus on Gaussian blur. This blur approximates the blur caused by upsampling an image fairly well and is a very good approximation of blur introduced to an image by capturing a scene through atmospheric turbulence [5].

The technique proposed in this paper focuses on identifying the standard deviation (σ) of a Gaussian blur applied to an input image. An interesting property of the Gaussian function is employed to identify the variance of the Gaussian blur in the input image by examining its scale-space representation [14]. The scale-space representation has been used previously in [15] to detect edges. In [15] edges are considered to be ideal step functions that have undergone blurring due to lighting and focal characteristics of the imaging system through which they

were captured. These blurs were modelled as Gaussian blurs. Through analysis of the derivatives of an image at various scales in the scale-space it was possible to locate blurred edges and identify the degree to which they were blurred.

The remainder of this paper will be structured as follows. Section II will present the theory employed in this algorithm. Section III will describe the algorithm itself. Section IV will present some experiments and discussion of their results and finally Section V will be the conclusion.

II. BACKGROUND

A. An Interesting Property of the Gaussian distribution

In image processing the most common operations use kernel filters that are panned around the image. The Gaussian equation used to produce these types of kernels is considered to have a zero mean. Thus the one dimensional Gaussian equation we are using is defined as follows:

$$G(x, \sigma^2) = ae^{\frac{-x^2}{2\sigma^2}}, \quad (2)$$

Where *a* is the amplitude of the curve and σ^2 is the variance of the Gaussian and its square root σ is the standard deviation [14]. For generating kernels for image processing *a* is generally considered to be 1.

The Gaussian equation exhibits self-similarity and thus the cascade property where if two Gaussians are convolved with each other they produce a new Gaussian as follows [14]:

$$G(x, \sigma_A^2) \otimes G(x, \sigma_B^2) = G(x, \sigma_A^2 + \sigma_B^2), \quad (3)$$

In this paper we exploit an interesting feature of the Gaussian equation. Given a Gaussian with a constant standard deviation σ_1, if we convolve this Gaussian with another Gaussian with standard deviation σ_2 we get a resulting Gaussian with the standard deviation of $\sqrt{\sigma_1^2 + \sigma_2^2}$. If we then subtract the resulting Gaussian from the original Gaussian with standard deviation σ_1 and absolute the result we get a measure of the difference or error between the original Gaussian and the new Gaussian. This process is described in the equations below and figure 1.

$$E = |G(x, \sigma_1^2) - G(x, \sigma_1^2) \otimes G(x, \sigma_2^2)|, \quad (4)$$

$$E = |G(x, \sigma_1^2) - G(x, \sigma_1^2 + \sigma_2^2)|, x \in [-B; B] \,\&\, \boldsymbol{Z}, \quad (5)$$

Where *x* is an integer that ranges between integer bounds defined by *-B* and *B*.

If you perform this process using a chosen value for σ_1 and a range of values for σ_2 and then plot the resulting errors you will find the response shown in figure 2. What is interesting is that the error curve contains a point of inflection where the concavity of the curve changes. To find the exact point of inflection we must look for extrema in the first derivative of the error curve which is shown in figure 3.

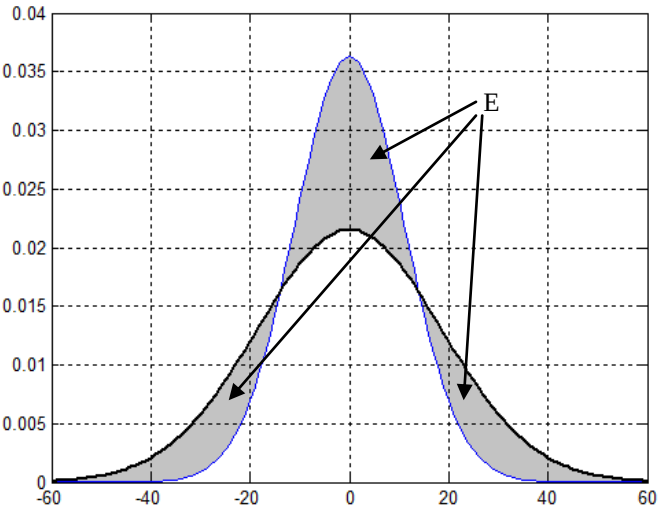


Figure 1: Error between 2 Gaussians

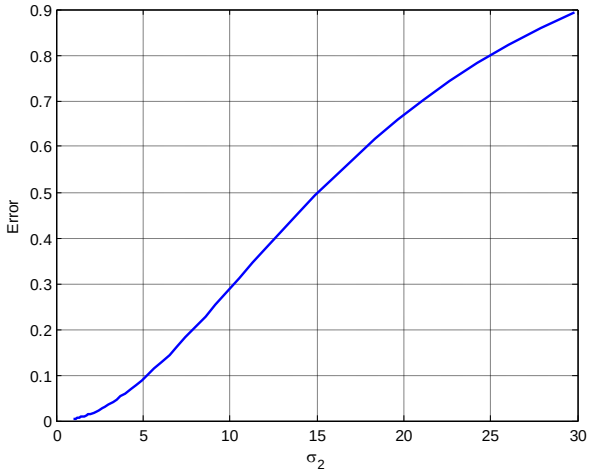


Figure 2: Error between a Gaussian with constant standard deviation σ_1=11 and a second Gaussian with standard deviation $\sqrt{\sigma_1^2 + \sigma_2^2}$ where σ_2 is varied over a range.

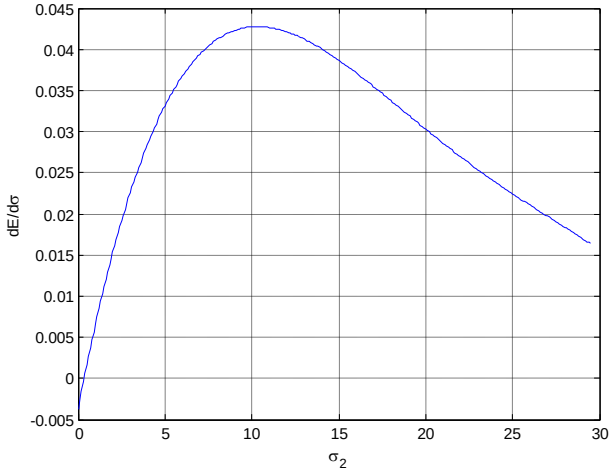


Figure 3: First derivative of the Error with respect to σ_2

As can been seen from figure 3 the maximum value of the first derivative of the error corresponds to the point of inflection in the error curve. This also corresponds with the chosen value for σ_1 which in this case was 11. This shows that while the error is increasing monotonically when σ_2 is smaller than σ_1 the error increases at a faster rate than when σ_2 is larger than σ_1. This phenomenon can be used to determine the value of σ_1 by only varying the value of σ_2 and searching for the point of inflection on the error curve.

B. The scale space

Scenes in the world appear very different when viewed from varying scales. For example a tree viewed from 1 meter away would be made up of individual branches, a trunk and leaves while if it was viewed from 1 km away it would appear to be a single solid object. The fact that scale is so important in describing the structure of objects being observed has led to the development of multi-scale representations of images. Being able to isolate the structures contained in an image at a given scale is an immensely powerful tool in being able to extract useful information from an image [14].

A large number of multi-scale representation techniques have been proposed in the literature. One of the first was the quad-tree representation which iteratively divides an image into smaller rectangles based on the information content inside each division [16]. Sampling pyramids have also been widely used. In these algorithms an image is recursively halved in size using a sub-sampling scheme and smoothed at each step to give a pyramid of images where each is half the size of level below. This approach is limited in the size of the steps at which its sampling size is reduced and thus objects at scales that exist between and levels of the pyramid are lost [14].

The scale-space representation was proposed to combat this problem. The scale-space is a representation that comprises a continuous scale parameter and preserves the same spatial sampling at all scales. It is shown in [14] that the only kernel that can achieve this is the Gaussian kernel. This approach takes an input image and blurs the image with a series of Gaussian kernels, each with a larger variance than the last. As the image becomes more and more blurred the finer scale information is averaged out and the larger scale structures are all that are left. In this way we can produce a series of images that each contain a different scale of structures but we do not introduce any quantization noise.

To take this representation a step further we can subtract each level of this multi-scale representation from the one below it to produce a Difference-of-Gaussian (DoG) representation of the image. This representation is essentially the second-order derivative of the images at each scale level. This multi-scale gradient information has been used in many feature detection, object detection and segmentation algorithms of which the most notable is probably the SIFT feature detector [17].

III. ALGORITHM DESCRIPTION

The algorithm described in this paper starts with an input image which we assume has been blurred with a Gaussian kernel as shown in the following equation.

$$I = F \otimes G(x, \sigma_1^2), \tag{6}$$

Where I is the input image, F is the image without the blur and the function *G* is a Gaussian kernel with a standard deviation of σ_1. The goal of the algorithm is to identify the standard deviation of this blur with no *a priori* information about the conditions under which the image was captured.

The next step is to construct a scale space representation of the input image I. This is done by blurring the input image I with a range of Gaussian kernels with increasingly large standard deviations. The range of standard deviations is calculated in a similar fashion to [17]. We start at a standard deviation of 1 and we call each doubling of this initial value an octave of σ values. We choose how many levels to divide each octave into. The range is then constructed as described by the pseudo-code in the following figure. This code assumes we want to construct 5 octaves of σ values with 10 divisions in each octave.

```
octaveDivisions = 10
numOfOctaves = 5
scaleFactor = 2.0^(1.0/octaveDivisions)
numOfLevels = octaveDivision*numOfOctaves+1
sigma(1) = 1;

For s = 2 to numOfLevels
      Sigma(s) = sigma(s-1)*scaleFactor
end
```

Figure 4: Pseudo-code describing generation of σ values for the scale-space representation

To construct the scale-space representation D we then convolve the input image with a Gaussian kernel with each of the σ values in the generated range.

$$D(\sigma_2) = F \otimes G(x, \sigma_1^2) \otimes G(x, \sigma_2^2), \tag{7}$$

Where σ_2 is the standard deviation from our generated range and σ_1 is the standard deviation of the Gaussian kernel we are trying to detect. The next step is to find the absolute error between the input frame and the images in the scale-space representation.

$$\mathrm{E}(\sigma_2) = |F \otimes G(x, \sigma_1^2) - F \otimes G(x, \sigma_1^2) \otimes G(x, \sigma_2^2)|, \tag{8}$$

$$\mathrm{E}(\sigma_2) = F \otimes |G(x, \sigma_1^2) - G(x, \sigma_1^2) \otimes G(x, \sigma_2^2)|, \tag{9}$$

Due to the distributability of convolution it can be seen that the error E contains the equation 4 convolved with the unblurred image F. This implies that the same analysis of the error response of E can be applied to determine the value of σ_1.

Thus once we have the error response for all values of σ_2 in our scale-space we find the first derivative of E with respect to σ_2. We use the basic finite difference technique to estimate the derivative of the range of σ_2 values and E as following set of convolutions.

$$dE = E \otimes [-1\ 1], \tag{10}$$

$$d\sigma = \sigma_2 \otimes [-1\ 1], \tag{11}$$

Where the $[-1, 1]$ term is a discrete kernel with two elements. The final step of the algorithm is to find the maxima of $dE/d\sigma$ and the corresponding σ_2 value. This value is the detected standard deviation of the blur that the input image contained. This process of blurring an image with a series of Gaussians with increasing standard deviations is also used to produce the scale space representation of an image. Thus this algorithm can be cheaply performed in tandem with algorithms that make use of the scale space representation of an image.

We found that the algorithm was fairly sensitive to additive white noise and as such we introduced an iterative median filtering pre-processing stage to the algorithm to aid in suppressing noise. This stage consisted of applying two iterations of a 3x3 median filter to the input image before the above described process is performed.

IV. EXPERIMENTS

To examine the performance of the algorithm at detecting unknown Gaussian blurs in natural images we performed the following experiments. Four test photographs were chosen and are shown as figures 5 through 8 below.

Each image was degraded with a Gaussian blur with standard deviations ranging from 1 to 20. After each blur was applied an additive white Gaussian noise was applied resulting in a signal-to-noise (SNR) of 30 dB (strong noise) and 40 dB (milder noise). The algorithm was then used to measure the amount of blur in the image. The results of these experiments are displayed in figures 9 through 12.

Figure 5: Aircraft test image

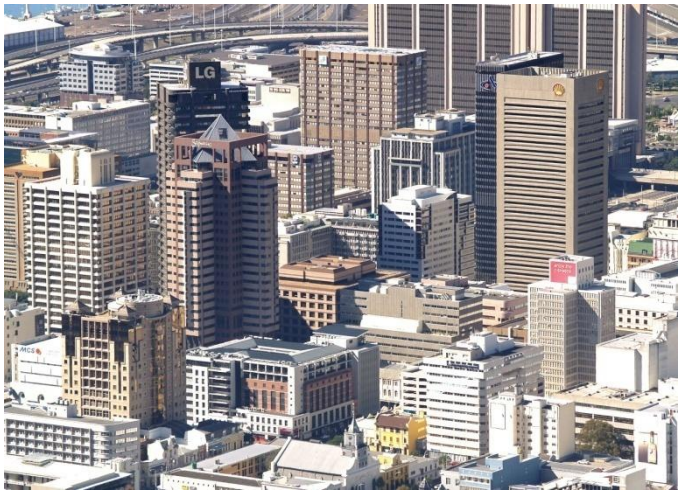


Figure 6: City test image

Figure 7: Bridge test image

Figure 8: Forest test image

As can be seen the algorithm successfully identifies the strength of the Gaussian blur applied to the images quite accurately in a range of standard deviations from 1 to 20

which is a far wider range of sigma values than algorithms currently in the literature. The presence of noise does decrease the accuracy of the identification especially in the Aircraft image which has large areas of uniform colour where noise becomes very apparent but the iterative median filtering does make the algorithm fairly resistant to noise.

It is interesting to note that in the Aircraft test image the strength of the identified blur does get over estimated. This is due to the large uniform coloured areas which have very little high frequency information content. This lack of high frequency content makes the images appear to be more blurred than they really are. In contrast the blur in the City test image is consistently underestimated due to the large amount of high frequency information present in the image. This over abundance of high frequency information makes the image appear to be less blurred than it is.

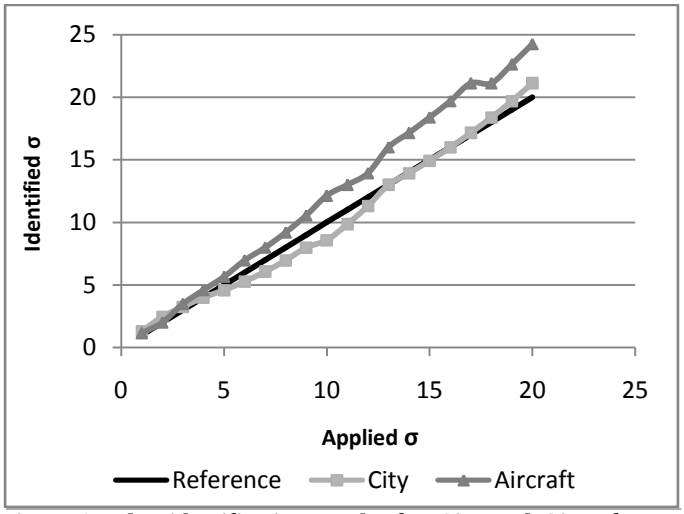


Figure 9: Blur identification results for City and Aircraft test images with 30 dB of additive white noise

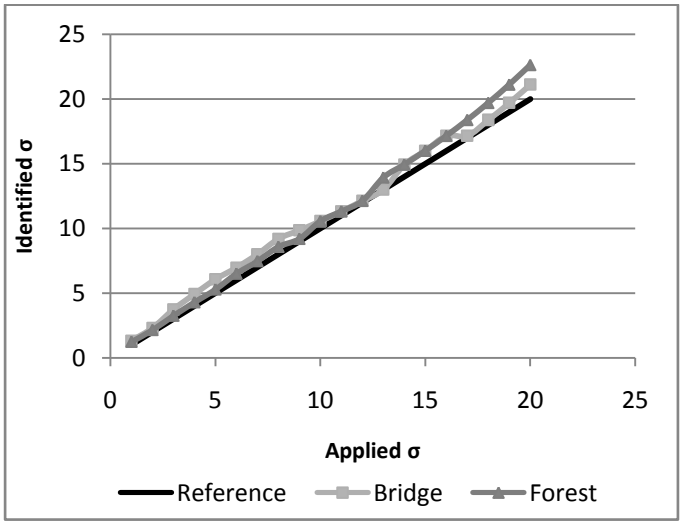


Figure 10: Blur identification results for Bridge and Forest test images with 30 dB of additive white noise

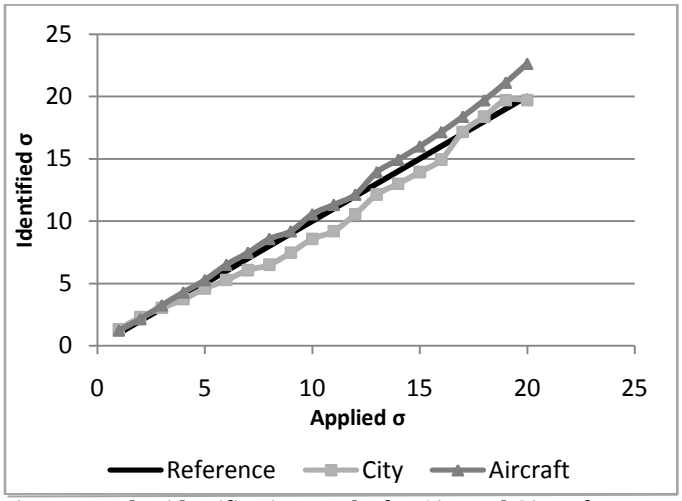


Figure 11: Blur identification results for City and Aircraft test images with 40 dB of additive white noise

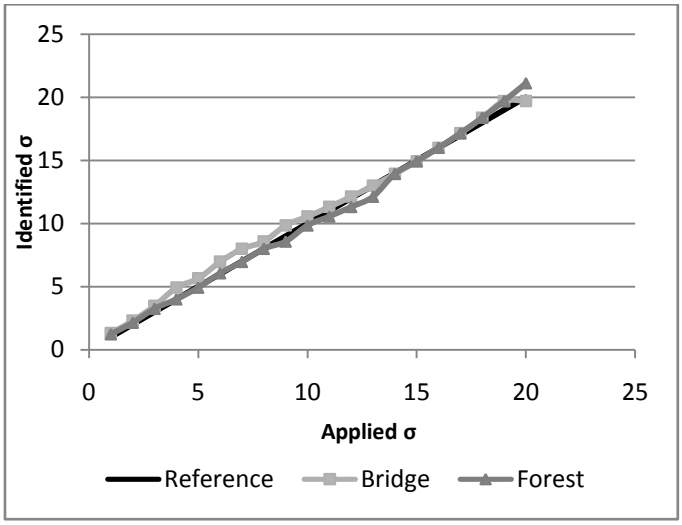


Figure 12: Blur identification results for Bridge and Forest test images with 40 dB of additive white noise

A final test was performed where an image that contains natural blur due to atmospheric turbulence is used as an input image. The strength of the blur is identified and the image is deconvolved using a plain Wiener filter using the identified Point Spread Function (PSF) [5]. The results of this experiment can be seen in figure 13. It is apparent that the identified blur strength is correct and the deconvolution deblurs the image without introducing ringing artifacts associated with an incorrectly identified PSF.

V. CONCLUSION

In this work is was shown that it is possible to detect the standard deviation of a Gaussian blur that has been applied to an image with no *a priori* information about the conditions under which the image was captured. The method uses an interesting property of the Gaussian function. When a series of Gaussians with increasing standard deviations are convolved with the Gaussian to be identified an error is produced. The error response this process produces has an inflection point

where the standard deviations of the Gaussians coincide and allows us to identify the standard deviation of the Gaussian being analyzed. This process is shown to work with a Gaussian blur applied to natural images. This method of blurring an image with a series of Gaussians is also used to produce the scale space representation of an image and can be performed in parallel with any algorithm that uses a scale space representation of an image.

The experiments show that in natural images with the presence of noise it is possible to identify Gaussian blurs with standard deviations that span a wide range without using any sort of statistical methods that require extensive training. It is also shown how this method can be used to identify the blur present in an image blurred naturally by atmospheric turbulence and allows one to deconvolve that image successfully using a basic Wiener filter.

Figure 13: The standard deviation of the blur present in a real image blurred by atmospheric turbulence is identified and used to deconvolve the image using a basic Wiener filter.

REFERENCES

[1] S. Winkler, *Digital video quality*, Wiley, 2005.

[2] H.R. Wu, K.R. Rao, *Digital Video Image Quality and Perceptual Coding*, CRC Press, 2005.

[3] S.J. Reeves, R.M. Mersereau, "Blur Identification by the method of Generalized Cross-Validation", *IEEE Transactions of Image Processing*, Vol. 1, No. 3, 1992.

[4] R.C Puetter, T.R. Gosnell, A. Yahil, "Digital Image Reconstruction: Deblurring and Denoising", *Annual Review of Astronomy and Astrophysics*, Vol. 43, pp 139-194, 2005.

[5] R.L. Lagendijk, J. Biemond. "Basic Methods for Image Restoration and Identification" in *Handbook of Image and Video Processing*. A. Bovik, San Diego: Academic Press, 2000, pp 125-139.

[6] A. K. Katsaggelos, *Digital Image Restoration*. Springer-Verlag, 1991.

[7] N. Joshi, C.L. Zitnick, R. Szeliski, D.J. Griegman, "Image deblurring and denoising using color priors", *Proceedings of CVPR. 2009*, pp. 1550-1557, 2009.

[8] A. Raj and R. Zabih. A graph cut algorithm for generalized image deconvolution. In ICCV '05, pages 1048–1054, 2005.

[9] Y. Harikumar, G.; Bresler. Exact image deconvolution from multiple fir blurs. IEEE TIP, 8(6):846–862, 1999.

[10] R. L. Lagendijk, A. M. Tekalp, and J. Biemond, "Maximum likelihood image and blur identification: a unifylng approach," J. Opt. Eng. 29,422435 (1990).

[11] H. Zheng, O. Hellwich, "Double regularized Bayesian estimation for blur identification in video sequences", *P.J. Narayanan et al. (Eds.) ACCV*, Vol. 3852, pp. 943–952. Springer, 2006.

[12] N. Joshi, R. Szeliski, D. Kriegman. "PSF estimation using sharp edge prediction". *In CVPR '08*, pages 1–8, 2008.

[13] D. Li, R.M. Mersereau, S. Simske, "Blur Identification based on kurtosis minimization", *ICIP 2005*, pp. 905-908, 2005.

[14] T. Lindeberg, "Scale-space theory: A basic tool for analysing structures at different scales", *Journal of Applied Statistics*, vol 21, no. 2, pp 225-270, 1994.

[15] J.H. Elder and S.W. Zucker, "Local scale control for edge detection and blur estimation", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 7, pp 699- 716, 1998.

[16] S. Hanan, R. Webber, "Storing a collection of polygons using Quadtrees", ACM Transactions on Graphics, Vol. 60, No. 2, pp 182 - 222, 1985.

[17] D.G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, Vol. 60, No. 2, pp. 91-110, 2004.

Adaptive Multi-Scale Retinex algorithm for contrast enhancement of real world scenes

Philip E. Robinson and Wing J. Lau
Faculty of Engineering and Built Environment
University of Johannesburg
South Africa
philipr@uj.ac.za, wing.j.lau55@gmail.com

***Abstract*—Contrast enhancement is a classic image restoration technique that traditionally has been performed using forms of histogram equalization. While effective these techniques often introduce unrealistic tonal rendition in real-world scenes. This paper explores the use of Retinex theory to perform contrast enhancement of real-world scenes. We propose an improvement to the Multi-Scale Retinex algorithm which enhances its ability to perform dynamic range compression while not introducing halo artifacts and greying. The algorithm is well suited to be implemented on the GPU and by doing so real-time processing speeds are achieved.**

***Index Terms*—Contrast Enhancement, Retinex, Adaptive, Multi-Scale Retinex, GPU**

I. Introduction

The human eye is a very complex and amazingly versatile imaging system. It exhibits an enormous dynamic range and can change its sensitivity very rapidly to operate in a large range of light levels; this ability is called brightness adaption. However the range of distinct intensities that the eye can distinguish at any one time is quite small compared to the total range of intensities it can adapt to perceive. This means the eye will struggle to discern very dim intensities when simultaneously exposed to very bright intensities. Unfortunately most artificial imaging systems have a much poorer level of brightness adaption than the human eye and as such can capture a very low dynamic range of intensities [1].

This results in many digital images exhibiting poor contrast either globally or in local regions. Contrast refers to the difference between the highest and lowest intensities used to represent an image. The wider the range of intensity values used to represent the information in an image or area of an image the higher the contrast. Contrast can also describe the distribution of intensity values used to represent the structures in the image. If the occurrence of intensity values are evenly distributed over the entire range of possible values it will be easier for a human viewer to to distinguish differing intensities. This is due to the fact that the various intensity levels will be spread further apart and are thus easier for our eyes to tell apart [1], [2].

There are a number of situations that can result in images exhibiting poor contrast. Some examples include images captured over a long range through the atmosphere where scattering and aerosols in the air result in the representation of the scene only occupying a small portion of the possible intensity values [3]. A second example is scenes with a very high dynamic range where portions of the image are in shadow and another portion of the image contains very bright information; this is otherwise known an High-Dynamic Range images (HDR). A final example is in medical scans where information produced by the detectors is very densely packed into the digital image representation [4].

The literature contains many techniques for contrast enhancement. The simplest is to apply an offset and gain to the image intensities based on the minimum and maximum values found in the image. This technique does improve contrast of most images but it is very sensitive to noise and outliers as a single noisy pixel can be found to be one of the extreme values and drastically perturb the scaling [2].

Histogram equalization quickly became a popular form of contrast enhancement and was first applied to medical scan images. These techniques operate based on the histogram of intensity values of an image. They seek to redistribute the intensities in the image in such a way as to achieve a uniform distribution of intensities across the entire intensity range [4]. Basic histogram equalization considers the histogram of the entire image in a global fashion, and as such struggles in images where a small portion of the image exhibits a drastically different intensity distribution which would then throw off the equalization for the rest of the image. To combat this Adaptive Histogram Equalization (AHE) was developed which performed the same process on a per-pixel basis based only on the pixel's neighbourhood. This approach achieves much higher contrast but amplifies noise, often in an extreme manner [5].

One of the most versatile forms of AHE is Contrast-Limited Adaptive Histogram Equalization (CLAHE) which puts a limit on just how drastically an intensity level can be redistributed. This algorithm works extremely well on medical images and fairly well on most real-world images. It has the added advantage of being relatively simple and as such has been implemented in a real-time system using specialized hardware [6]. While there has been an enormous amount of research done into histogram based contrast enhancement algorithms, such as [7], [8], they have some drawbacks. These algorithms tend to produce unrealistic effects when they are applied

to real-world images which is why they have mainly been applied to scientific images like medical, thermal and satellite images. In addition while consumer Graphics Processor Units (GPU) have provided a parallel computing platform that has accelerated the implementation of real-time image processing algorithms the construction of the histogram is awkward on the parallel architecture of the GPU. Efficient implementations of the histogram have been proposed for GPU frameworks like CUDA but for lower level GPU API's like OpenGL the histogram is still costly to compute. This paper explores another approach to contrast enchantment which is better suited to real-world scenes and easily implemented on the GPU.

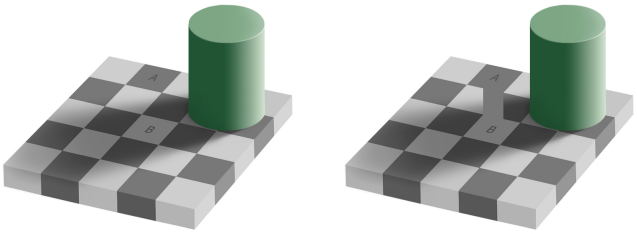
In this paper we are going to make use of Retinex theory to perform contrast enhancement. Retinex theory was first proposed by Land and McCann in [9] to describe a model of how the eye perceives light intensities, which is often at odds with the actual physical intensities the eye experiences [10]. This theory has been greatly expanded for use in image processing since its proposal in papers such as [11]–[14]. This paper aims at furthering this approach which due to its origins in Retinex theory produces very natural looking results and lends itself well to real-time implementation on the GPU.

The remainder of this paper will be structured as follows. Section II will provide a description of Retinex theory and its application to contrast enhancement. Section III will present the proposed algorithm. Section IV will show our results and Section V will conclude the paper.

II. Overview of Retinex-based contrast enhancement

Retinex theory was developed by Land and McCann to model the disparity they observed between the lightness of various parts of a scene perceived by the human eye and the absolute lightness that was actually incident on the eye. What they found was that the eye does not perceive absolute lightness but rather relative lightness. This means that the eye perceives the variations of relative lightness in local areas in the scene [9], [10]. This phenomenon is what gives the human eye its great dynamic range and is illustrated in the classic optical illusion shown in Fig. 1. While it is difficult to believe, square A and square B in Fig. 1 are the exact same colour. We perceive that square B is a lighter colour because it is surrounded by darker squares and in contrast to its immediate neighbours it is indeed lighter. Square A on the other hand appears to be dark because in contrast to its immediate neighbours it is darker. Our eyes and our brain cannot help but perceive the absolute lightness of square B to be brighter than square A even though we can see that they are identical in the second image.

The second element of Retinex theory that we exploit to achieve contrast enhancement is that our eyes exhibit a logarithmic response to lightness. This is to allow us to differentiate a greater number of dim intensities compared to bright intensities [1]. This allows us to operate better in dark environments which are far more challenging for our visual system than bright environments. This means that using a logarithmic mapping Retinex based algorithms map intensities using a response curve that appears more natural to our eyes.



(a) (b)

Fig. 1. (a) The Adelson Checker Shadow Illusion [15] (b) Proof that square A and B are identical intensities

Equations 1 and 2 show the basic formulation of the Single Scale Retinex (SSR) scheme.

$$R(x,y) = \frac{\log I(x,y)}{\log[F(x,y) * I(x,y)]} \tag{1}$$

$$R(x,y) = \log I(x,y) - \log[F(x,y) * I(x,y)] \tag{2}$$

where $I(x,y)$ is the 2-dimensional input image, "$*$" denotes the convolution operator, $F(x,y)$ is the surround function, and $R(x,y)$ is the SSR output. $F(x,y)$ is the function that defines the shape and weighting of the averaging kernel used as a measure of the neighbourhood lightness for each pixel [11]. It can be seen that SSR can be considered to be a logarithmic mapping of the ratio of the current pixel intensity to the average intensity around the pixel. In [11] it is shown that the best choice for the surround function is a Gaussian which not only gives the best results but has the added advantage of being a separable kernel. A separable 2D kernel is one that can be expressed as the outer product of 2 vectors. This means that instead of applying the kernel in its 2 dimensional form one can apply each of the constituent vectors. This approach drastically reduces the number of computations required to apply the kernel to an image. Equation 3 describes the Gaussian function.

$$F(x,y) = Ke^{-\frac{x^2+y^2}{2\sigma^2}} \tag{3}$$

where σ is the standard deviation that controls the scale of the surround. K is chosen to normalize the kernel such that:

$$\iint F(x,y)\,\mathrm{d}x\,\mathrm{d}y \;= 1. \tag{4}$$

SSR does exhibit a few problems in that if the scale is set too small you get good dynamic range compression but you generate a halo effect around edges. If you set the scale too high you get less dynamic range compression and a greying effect can be seen in more uniform areas. In [12] it is shown that applying the Retinex scheme at only a single scale cannot simultaneously provide good tonal rendition and

good dynamic range compression and thus they proposed a Multi-Scale Retinex (MSR) algorithm. This algorithm applies the Retinex technique at several scales and then combines the results using a weighted sum to produce an output as shown in equation 5.

$$R_{MSR}(x,y) = \sum_{n=1}^{N} w_n R_n(x,y) \tag{5}$$

where $R_{MSR}(x,y)$ is the Multi-Scalar Retinex (MSR) output, $R_n(x,y)$ is the output of Single Scale Retinex (SSR) at different scales, and w_n are the weights associated with the different scales. The weights are chosen so that $\sum w_n = 1$, and N designates the number of scale levels used.

The MSR output contains logarithmic values that run from very small negative numbers into the positive domain. As such the final step in the algorithm is to normalize the resulting values to fall between 0 and 1. This is done using a gain/offset scheme as described in equation 6.

$$R_{MSR_i}(x,y) = \alpha[\sum_{n=1}^{N} w_n R_{n_i}(x,y)] - \beta \tag{6}$$

where α is called the gain and β is the offset. β is based on the minimum value in the image and used to ensure that the minimum value in the final resulting image is 0. The gain α is calculated by dividing 1 by the difference between the maximum and minimum values in the MSR output and scales final resulting image so that its maximum value is 1. These values are calculated globally which means that this approach has a similar problem to a global histogram equalization in that if the image contains areas with drastically different intensity distributions the global α and β will not be ideal for all the regions in the image.

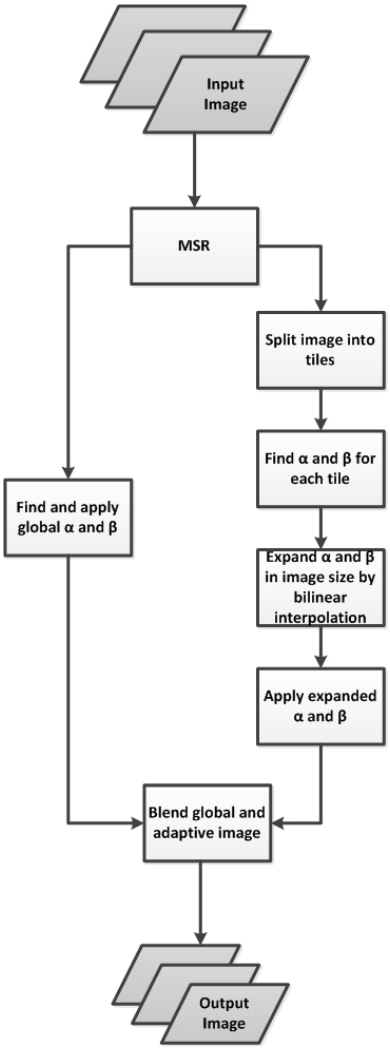


Fig. 2. Overview of proposed algorithm

III. Proposed Algorithm

In this paper we offer an improvement over the classic formulation of the MSR algorithm. To improve the dynamic range compression of the algorithm without incurring the halo artifacts we propose using an adaptive approach to calculating the gain and offsets for the final stage of the algorithm and to blend these results with the those produced by the global calculation. The overview of our proposed algorithm is illustrated in Fig. 2.

Our approach draws from the adaptive techniques used in CLAHE [6]. The image is firstly divided into a set of tiles. The β values are then found for each tile by calculating the minimum intensities. Next the α values are found for each tile by finding the difference between the maximum and minimum intensities. This process produces a 2-dimensional field of α and β values the same size as the number of tiles selected.
The next step is to expand the field of α and β values to be the same size at the image. This is done using bilinear interpolation. This method is chosen because bilinear interpolation is cheap to calculate on the GPU. Once we have expanded the α and β fields we will have values for each pixel of the MSR image. We can now apply the α and β values to normalize the image. An example of the result of applying the adaptive α and β values can be seen in Fig. 3.

As can be seen in Fig. 3 by applying the adaptive α and β values we do achieve good dynamic range compression but in tiles where the image intensities are very uniform we end up drastically amplifying the noise in that tile. When calculating the global α and β values it is very unlikely that the entire image will be a uniform intensity and as such we will not experience this over-gain. Thus we will not experience the same noise amplification we see when using purely adaptive values for α and β. Due to this problem we propose blending the outputs of the global gain/offset correction step and the adaptive gain/offset correction step to achieve a compromise between contrast enhancement and noise amplification.

To facilitate the blending of the Global and Adaptive MSR results we have to produce a blend map. We found that the full sized field of α values gives a good indication of how the two MSR images should be blended. Areas that

Fig. 3. Result after applying the adaptive α and β values to the MSR image of the *Road* input image which can be seen in figure Fig.7. The indicated region shows the over-gain problem experienced in areas of the image where the image intensities are very uniform.

require a very large gain usually are areas that are very uniform in intensity and as such areas that should contain a larger portion of the Global MSR output. Areas that required a low α value should contain a larger portion of the Adaptive MSR output. As such our blend map is produced by first normalizing the interpolated field of α values by dividing by the maximum α value which can be seen in Fig. 4.

Fig. 4. Example of a blend map for the *Road* image

Once we have the normalized blend map we can combine the Adaptive MSR and Global MSR outputs as a weighted sum which can be seen in equation 7.

$$R_{MSR_B} = \phi \times R_{MSR_G} + (1 - \phi) \times R_{MSR_A} \qquad (7)$$

where ϕ represents the normalised blend map image, R_{MSR_G} represent the Global MSR image, R_{MSR_A} the MSR Adaptive image, and R_{MSR_B} the MSR blended image.
The final design decision we had to undertake was to select the number of scales, size of the scales and the weightings of the scales for the MSR algorithm. In [12] it is shown that 3 scales are sufficient to achieve good tonal rendition and dynamic range compression and this observation was confirmed in our experiments. Jobson et al. suggest standard deviations of 15, 80 and 250 for the scales used to enhance images under a megapixel in size. We found that these values produced good results but needed to be scaled for images of differing sizes for optimal results. It was also noted that a Gaussian kernel with a standard deviation of 250 is very large and almost encompasses an entire image with a VGA resolution. In the interest of reducing the amount of computation required for the algorithm instead of computing the surround function averages of the largest scale we considered them to be the mean value of the entire image. This can be computed efficiently and produces very similar results as using the large scale suggested in [12]. For the two smaller scales we used a basic heuristic, which we based on empirical testing, to choose the scale size based on the input image size. The standard deviation of the surround function for the smallest scale was considered to be 1.5% the size of the width of the image. The second scale was considered to be 5% the size of the width of the image. Finally we had to choose the weighing of scales and we found that while the best results were produced by heavily weighting the largest scale it was critical to have an element of the smaller scales in the algorithm output to enhance the contrast of small image structures. The weights we used for the smallest to largest scales were 0.2, 0.1 and 0.7 respectively. We leave the investigation of what the optimal scales and weightings are for future work.

IV. Experimental Results

To demonstrate the performance of our proposed algorithm we have selected three images. The first is a HDR image and the final two are images that have been captured through atmospheric turbulence. The proposed algorithm will be compared to four traditional contrast enhancement techniques. Those techniques include Histogram Equalization (HE), Adaptive Histogram Equalization (AHE), Contrast Limited Adaptive Histogram Equalization (CLAHE), and traditional MSR. The results can be seen in Fig. 5, 6 and 7

In Fig. 5 we can see that our proposed Adaptive Multi-Scale Retinex (AMSR) algorithm gives the most pleasing results for this extreme HDR image. Much of the information in the dark areas on the left of the image that were originally hidden is revealed while also providing good contrast in the bright areas of the image. The HE result is very legible but is can be seen that there is saturation in the brightest and darkest areas in the image which is to be expected for a global approach. AHE gives a very strong contrast but is very noisy and unrealistic. CLAHE we found does not cope well with HDR images and even when the Clipping Limit is manually tuned we could not produce an image where neither the dark or light portions of the image were saturated. Global MSR does perform well for this image but as can be seen

AMSR achieves greater contrast, especially in the darkest and brights areas, while retaining realistic tonal rendition.

Fig.6 is a difficult image because it is exhibits a very low contrast and has a large proportion of areas of uniform intensity. In Fig.6 we see that global HE tends to produce unrealistic results and AHE gives strong contrast but is extremely noisy. CLAHE and MSR both produce decent and very similar results but AMSR manages to produce the best tonal rendition especially in the darker area on the left of the tower. For Fig.7 CLAHE produces better results than standard MSR but we can see that the proposed AMSR algorithm produces the best contrast enhancement consistently across the entire image. Again HE produces unrealistic results and AHE is extremely noisy due to the large uniform regions.
It is interesting to look at the histograms of the images in Fig.7 which can be seen in Fig.8. It is apparent that our AMSR algorithm produces the histogram with the smoothest and widest spread without resulting in saturation at the black or white bounds of the histogram. The smooth histogram produced by AMSR captures the same peaks that can be seen in the histogram of the original image and distributes them very neatly across the intensity range resulting in a high contrast output that appears natural to a human viewer. Unfortunately there are no empirically-based metrics in the literature that have been able to objectively and reliably measure the perception of the contrast of complex real world images by a human observer, however work is being performed to develop such a metric based on the survey of a large sample of human observers [16]. In this paper we employed the classic information metric of entropy [1] as an attempt to quantitatively measure the quality gain the algorithms produce, table I shows these results. Firstly we can see the problem with using these sorts of metrics in the results for the AHE outputs. These images are extremely noisy and the metric perceives the noise as large amounts of information even though noise is not perceived as useful to a human observer. We can however see that for the CLAHE, MSR and AMSR results we get a useful comparison. In the HDR image $Shadow$ MSR out performs CLAHE but AMSR gives the most information gain. In $Road$ and $Tower$ we can see that CLAHE produces more information than MSR but AMSR beats CLAHE in both cases.

	Original	HE	AHE	CLAHE	MSR	AMSR
Tower	5.95	5.4806	7.8496	6.5591	6.3629	6.7471
Road	6.4857	5.8594	7.993	7.3181	7.0171	7.4123
Shadow	6.0406	5.0049	7.8669	6.8229	7.4627	7.7254

TABLE I
ENTROPY TEST RESULTS

The AMSR Algorithm was implemented for the GPU using OpenGL. The algorithm was run on a desktop computer with the specifications show in table II. For comparison we used a GPU implementation of the CLAHE algorithm which uses scattering to produce histograms and is discussed in [17] and the source code can be found [18]. The AMSR and CLAHE implementations were run using the $Tower$ and $Road$ images found in Fig.6 and 7. The results are shown in table III. As can been seen the AMSR algorithm runs faster than the CLAHE algorithm by almost an order of magnitude. This is because the AMSR algorithm is based on a series of basic kernel convolutions and does not require the awkward implementation of the histogram that is required in CLAHE.

CPU	Intel Core I7-2600k 3.4 GHz Processor
RAM	8 GB DDR3 RAM
GPU	nVidia GTX 580 graphics card

TABLE II
SPECIFICATION OF THE DESKTOP COMPUTER USED IN THE PERFORMANCE TESTS

	560x460 resolution	876x592 resolution
CLAHE	30 fps	14 fps
AMSR	296 fps	131 fps

TABLE III
PERFORMANCE TEST RESULTS

V. CONCLUSION

Contrast enhancement is a classic image restoration technique that has been employed to improve the legibility of images and the information they contain since the times of analog image capture. The traditional approach to digital contrast enhancement is to employ a form of histogram equalization. While this approach does improve contrast it often produces an unrealistic and saturated effect which is very apparent when applied to real-world scenes. This paper explores the use of Retinex theory for the purpose of contrast enhancement. An overview of Retinex theory and its use as a digital image processing technique in the form of the Single-Scale and Multi-Scale Retinex algorithms is provided.

This paper proposes an improvement to the traditional global Multi-Scale Retinex algorithm which allows it to improve its dynamic range compression while not producing the traditional artifacts associated with Retinex based methods. The Adaptive Multi-Scale Retinex algorithm makes use of a model of how our eyes naturally perceive scenes and as such the output of the algorithm looks very natural to a human viewer. The experimental results show that for real-world images AMSR produces slightly better results than CLAHE which is currently the most versatile contrast enhancement algorithm in the literature. Our Adaptive Multi-Scale Retinex algorithm is also well suited to implementation on the GPU and achieves speeds around 10 times faster than a GPU implementation of CLAHE as AMSR is based on simple kernel convolutions and does not require the awkward GPU implementation of the histogram.

(a) Original

(b) HE

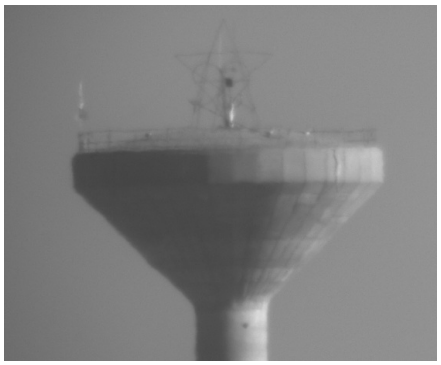
(c) AHE

(d) CLAHE

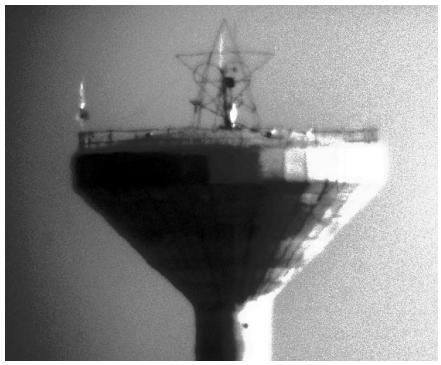
(e) MSR

(f) proposed

Fig. 5. Contrast enhancement results for the HDR $Shadow$ [19] image

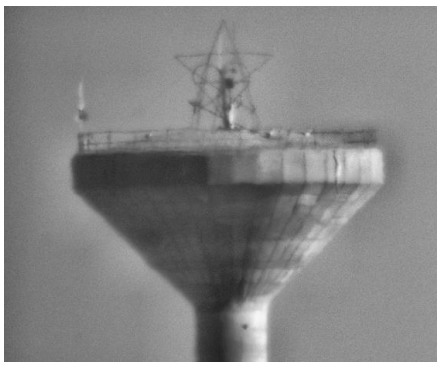


(a) Original

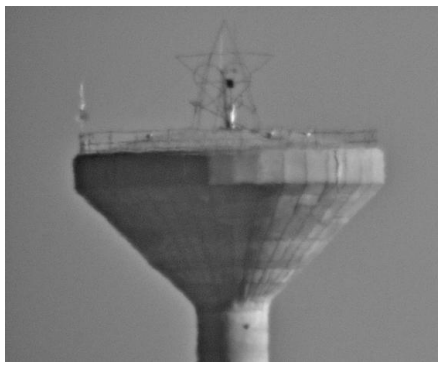


(b) HE

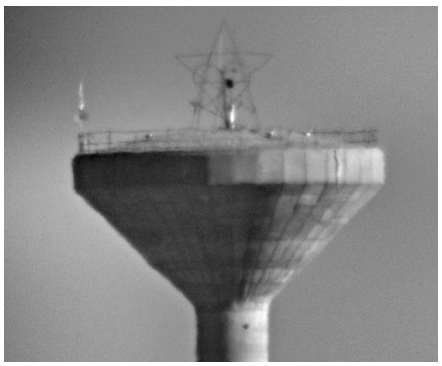
(c) AHE



(d) CLAHE



(e) MSR



(f) proposed

Fig. 6. Contrast enhancement results for image $Tower$ which has been captured through atmospheric turbulence

(a) Original

(b) HE

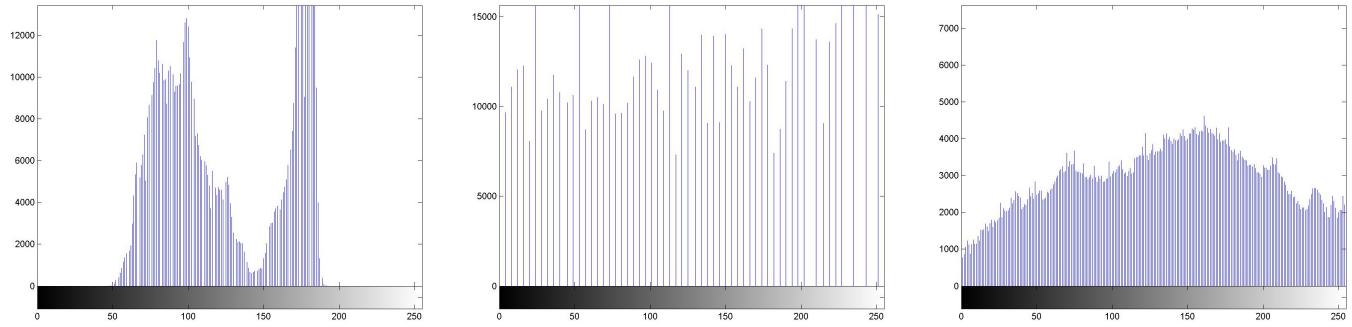
(c) AHE

(d) CLAHE

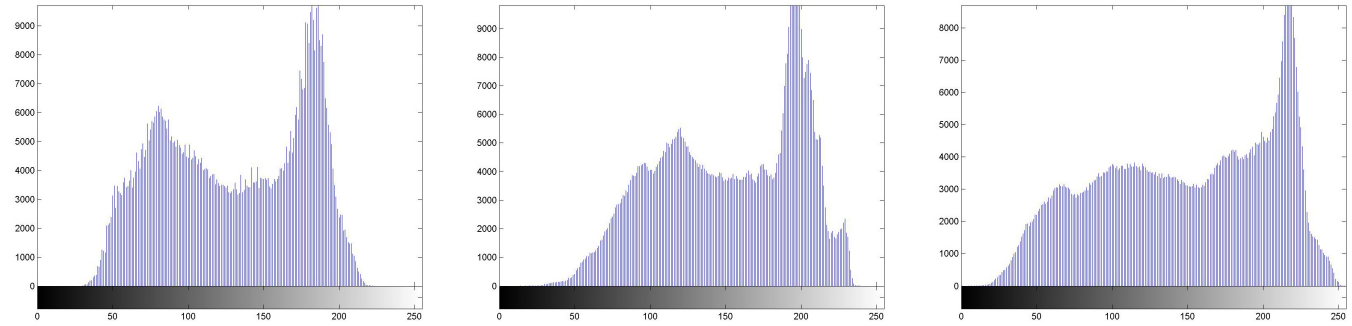
(e) MSR

(f) proposed

Fig. 7. Contrast enhancement results for image *Road* which has been captured through atmospheric turbulence



(a) Original (b) HE (c) AHE



(d) CLAHE (e) MSR (f) proposed

Fig. 8. Histograms for Fig. 7 *Road*.

REFERENCES

[1] R. Gonzales and R. Woods, *Digital Image Processing*. Pearson Prentice Hall, third ed., 2008.

[2] A. Bovik, *Handbook of Image and Video Processing*. Academic Press, 2000.

[3] P. E. Robinson and W. A. Clarke, "Sharpening and contrast enhancement of atmospheric turbulence degraded video sequences," *Proceedings of the Twenty-First Annual Symposium of the Pattern Recognition Association of South Africa*, 2010.

[4] J. Zimmerman, S. Pizer, E. Staab, J. Perry, W. McCartney, and B. Brenton, "An evaluation of the effectiveness of adaptive histogram equalization for contrast enhancement," *Medical Imaging, IEEE Transactions on*, vol. 7, pp. 304 –312, dec 1988.

[5] S. M. Pizer, E. P. Amburn, J. D. Austin, R. Cromartie, A. Geselowitz, T. Greer, B. ter Haar Romeny, J. B. Zimmerman, and K. Zuiderveld, "Adaptive histogram equalization and its variations," *Computer Vision, Graphics, and Image Processing*, vol. 39, pp. 355–368, Sept. 1987.

[6] A. M. Reza, "Realization of the Contrast Limited Adaptive Histogram Equalization (CLAHE) for Real-Time Image Enhancement," *The Journal of VLSI Signal Processing-Systems for Signal, Image, and Video Technology*, vol. 38, pp. 35–44, Aug. 2004.

[7] D. Menotti, L. Najman, J. Facon, and A. de Araujo, "Multi-histogram equalization methods for contrast enhancement and brightness preserving," *Consumer Electronics, IEEE Transactions on*, vol. 53, pp. 1186 –1194, aug. 2007.

[8] Y.-T. Kim, "Contrast enhancement using brightness preserving bi-histogram equalization," *Consumer Electronics, IEEE Transactions on*, vol. 43, pp. 1 –8, feb 1997.

[9] E. H. LAND and J. J. McCANN, "Lightness and retinex theory," *J. Opt. Soc. Am.*, vol. 61, pp. 1–11, Jan 1971.

[10] E. H. Land, "An alternative technique for the computation of the designator in the retinex theory of color vision.," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 83, pp. 3078–80, May 1986.

[11] D. J. Jobson, Z. Rahman, and G. a. Woodell, "Properties and performance of a center/surround retinex.," *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, vol. 6, pp. 451–62, Jan. 1997.

[12] D. J. Jobson, Z. Rahman, and G. a. Woodell, "A multiscale retinex for bridging the gap between color images and the human observation of scenes.," *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, vol. 6, pp. 965–76, Jan. 1997.

[13] B. Funt, F. Ciurea, and J. Mccann, "Retinex in matlab," in *Journal of Electronic Imaging*, pp. 112–121, 2000.

[14] K. Barnard and B. Funt, "Investigations into multi-scale retinex," in *Color Imaging in Multimedia*, pp. 9–17, Technology, (Wiley, 1999.

[15] E. H. Adelson, "Checker shadow illusion." http://web.mit.edu/persci/people/adelson/checkershadow_illusion.html. Accessed: 21/09/2012.

[16] A. Haun and E. Peli, "Measuring the perceived contrast of natural images," *SID Symposium Digest of Technical Papers, Session 24: Visual Perception (APV)*, pp. 302–304, 2011.

[17] T. Scheuermann and J. Hensley, "Efficient histogram generation using scattering on gpus," *Proceedings of the 2007 ACM Symposium on interactive 3D Graphics and Games*, 2007.

[18] T. Scheuermann, "Uniform and adaptive histogram equalization." http://sebastien.hillaire.free.fr/index.php?option=com_content&view=article&id=59&Itemid=70. Accessed: 11/11/2012.

[19] Z. Doob, "Siena shadow." http://www.photoflavor.com/index.php?id=477. Accessed: 21/09/2012.

Extended Local Binary Pattern Features for Improving Settlement Type Classification of QuickBird Images

L. Mdakane and F. van den Bergh

Remote Sensing Research Unit, Meraka Institute
CSIR, PO Box 395, Pretoria
South Africa, 0001
Email: lmdakane@csir.co.za, fvdbergh@csir.co.za

***Abstract*—Despite the fact that image texture features extracted from high-resolution remotely sensed images over urban areas have demonstrated their ability to distinguish different classes, they are still far from being ideal. Multiresolution gray-scale and rotation invariant texture classification with Local Binary Patterns (LBPs) have proven to be a very powerful texture feature. In this paper we perform a study aiming to improve the performance of the automated classification of settlement type in high resolution imagery over urban areas. That is, we combined the LBP method based on recognising certain patterns, termed "uniform patterns" with the rotational invariant variance measure that characterises the contrast of the local image texture, then combined multiple operators for multiresolution analysis. The results showed that the joint distribution of these orthogonal measures improve performance over urban settlement type classification. This shows that variance measure (contrast) is an important property when classifying settlement types in urban areas.**

I. Introduction

Rapid and massive growth of population and migration to urban areas results in a rapid and random spread of formal and informal physical infrastructure. Effective and regular monitoring of this spread of infrastructure is vital in delivering basic engineering services such as water, sewerage and solid waste removal, and providing essential services such as health and education. For a successful monitoring system, an effective detection method of this infrastructure is crucial. Traditional methods such as census, gathering demographic data, and mapping using samples are impractical and unsatisfactory for urban management [1]. However, using remote sensing tools an automated system can be used as a detection tool of physical infrastructure [2]. Using high resolution imagery (e.g., QuickBird, a high-resolution commercial earth observation satellite), texture feature algorithms have been shown to be effective in detecting and describing settlement types in urban areas [3].

In a study to compare texture algorithms in urban settlement classification, the Local Binary Pattern (LBP) texture feature algorithm proved to be most effective in classifying the low-income and informal settlement types [4]. A 2-D surface texture has two properties, spatial structure (pattern) and contrast ("amount of texture"). The LBP is simple to compute and by definition is gray scale invariant, that is, it neglects contrast properties which makes the LBP algorithm an excellent measure for spatial structures. However, due to viewing- and illumination-geometry effects, the LBP algorithm was shown to offer less than ideal generalization performance [3]. For settlement classification one would think neglecting the contrast measure would improve performance, but this does not appear to be the case for generalization performance. Contrast may have a significant effect in the classification of settlements.

Ojala [5] showed that combining spatial structure with the gray level contrast can improve performance in classifying texture features. In an attempt to improve performance for urban settlement classification, we apply this theory and evaluate the significance of contrast in urban settlement classification. The proposed algorithm uses the same rotational gray scale and rotational invariant LBP and combines it with a rotational invariant Variance measure which characterises the contrast.

In this paper (using Van den Bergh's [3] work on cross-date imagery for comparative results), we show that adding the rotational invariant variance measure to the gray-scale and rotational invariant LBP improves performance. The performance of the extended LBP algorithm then depends on the number of bins (features) used to calculate the Variance measures.

Section 2 briefly discusses prior and related urban settlements classification algorithms, and a brief derivation of the algorithms used. Section 3 discusses the experimental procedure i.e., extraction of the input images, LBP features extraction, extended LBP feature extraction and classification of the settlement types. Results with discussion are discussed in section 4, followed by conclusions in section 5.

II. Prior and Related work

Image texture analysis methods have been broadly divided into three categories: statistical methods (here a texture image is described by a collection of statistics of the selected feature, e.g., Co-occurence Matrix), model based methods (a texture

image is modeled as a probability model or as a linear combination of a set of basis functions, e.g., Wavelet transform [6] and Markov model [7]) and structural based methods (a texture image is viewed as consisting of many textural elements called texels, arranged according to some placement rules, e.g., Morphological Profiles [8]) [9].

In urban area images stuctural based methods have been shown to be successful in setting apart different settlement types [2]. The LBP (structural method) appeared to be most effective when compared to other known texture algorithms (e.g. Gray-level Co-occurrence Matrix (GLCM), Granulometrics and Discrete Wavelet Transform (DWT)) [4]. The LBP was used for cross-date Quickbird image (Soweto, located in Gauteng, South Africa as study area) urban settlement type classification and the results were not as impressive due to effects of varying viewing- and illumination geometry of satellite images [3].

The cross-date images study [3] involved the classification of two scenes of the same area acquired under different conditions. The images were acquired at different times of the year, which altered the orientation and length of the shadows. An ideal texture feature is one that is insensitive to such changes whilst being sensitive to settlement type. The addition of a contrast component to the LBP features does not directly effect the desired invariance to shadow orientation and length, but it is expected that the richer features will nevertheless improve settlement classification accuracy. The same data set (Soweto case study) will be used as basis for comparison with the extended LBP algorithm.

The extended LBP is a joint distribution of gray-scale and rotational invariant LBP with the rotational invariant Variance measure. We do not go into detail in the derivation of the algorithms but only report the equations used. The full derivations can be found in [5].

A. Gray-Scale and Rotational Invariant Local Binary Patterns

LBPs by definition are invariant with respect to any monotonic transformation of the gray scale. This is achieved by considering just the signs of differences instead of the exact values of the gray scale. Consider texture T

$$T \approx t(s(g_0 - g_c), s(g_1 - g_c), \ldots, s(g_{P-1} - g_c)) \quad (1)$$

in a local neighbourhood with gray levels of $P(P > 1)$ image pixels. Where $g_P(p = 0, \ldots, P-1)$ gray values, g_c being the centre gray value (see figure 2a), and

$$s(x) = \begin{cases} 1, x \geq 0 \\ 0, x < 0 \end{cases} \quad (2)$$

the sign is 1 if positive and 0 if negative. The above is transformed into a unique P-bit pattern code by assigning binomial coefficient 2^P to each sign s$(g_P - g_c)$:

$$\mathrm{LBP}_{P,R} = \sum_{p=0}^{P-1} s(g_P - g_c)2^P \quad (3)$$

LBP features are then calculated using the rotational invariant LBP with "uniform patterns" (uniform circular structures, illustrated in figure 2b):

$$\mathrm{LBP}^{\mathrm{riu2}}_{P,R} = \begin{cases} \sum_{p=0}^{P-1} s(g_p - g_c) & \text{if } U(LBP_{P,R}) \leq 2 \\ P+1 & \text{otherwise} \end{cases} \quad (4)$$

where

$$U(LBP_{P,R}) = \begin{array}{l} |s(g_{P-1} - g_c) - s(g_0 - g_c)| \\ + \sum_{p=0}^{P-1} |s(g_P - g_c) - s(g_{P-1} - g_c)|. \end{array} \quad (5)$$

U("pattern") is a uniformity measure, which corresponds to the number of spatial transitions in the "pattern" and superscript $(\mathrm{riu2})$ is $r$otation $i$nvariant "$u$niform" binary patterns that have a U value of at most 2.

B. Rotational Invariant Variance Measures

Local gray level variance can be used as a contrast measure and can be derived as follows:

$$\mathrm{VAR}_{P,R} = \frac{1}{P}\sum_{p=0}^{P-1}(g_p - \mu)^2, \text{where } \mu = \frac{1}{P}\sum_{p=0}^{P-1} g_p. \quad (6)$$

$\mathrm{VAR}_{P,R}$ is invariant against shifts in gray scale and rotation along the circular neighbourhood.

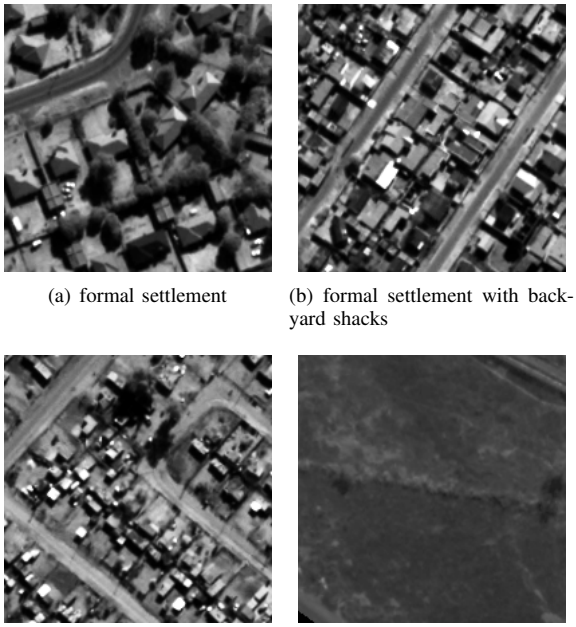
To improve performance of the LBP we consider its joint distribution with the local variance denoted as $\mathrm{LBP}^{\mathrm{riu2}}_{P,R}/\mathrm{VAR}_{P,R}$

III. Experiment

The data consisted of two QuickBird images over the Soweto area: one acquired on 2005-10-18 (early summer, rain season, called d1), and another on 2006-05-30 (early winter, called d2) [3]. QuickBird is a sun-synchronous polar-orbiting remote sensing satelitte with a panchromatic sensor with a 0.6 m ground sampling distance. This high resolution band, together with four multispectral bands at 2.4 m resolution, makes QuickBird ideal for urban monitoring. The study area contains a large variety of formal and informal settlements. Four settlement types are investigated: *formal suburbs(FS), formal settlements with backyard shacks (FSB), ordered informal settlements (OIS)*, and a *non-built-up (NBU)* class to represent vegetation and bare areas. Figure 1 provides some samples of what these settlement classes look like.

The experimental procedure was as follows:

1) *Extract input images*
   Two QuickBird images (Panchromatic images with a resolution of 0.6 m) over the same area at different times with different viewing- and illumination geometries were acquired. From each image, polygons containing examples of different settlement types were extracted, from which multiple non-overlapping examples of each type were extracted. From each polygon, square tiles (120 m $\times$ 120 m) from random locations entirely within the demarcated polygons were extracted. Tiles were



(a) formal settlement (b) formal settlement with back-yard shacks

(c) ordered informal settlement (d) non-built-up

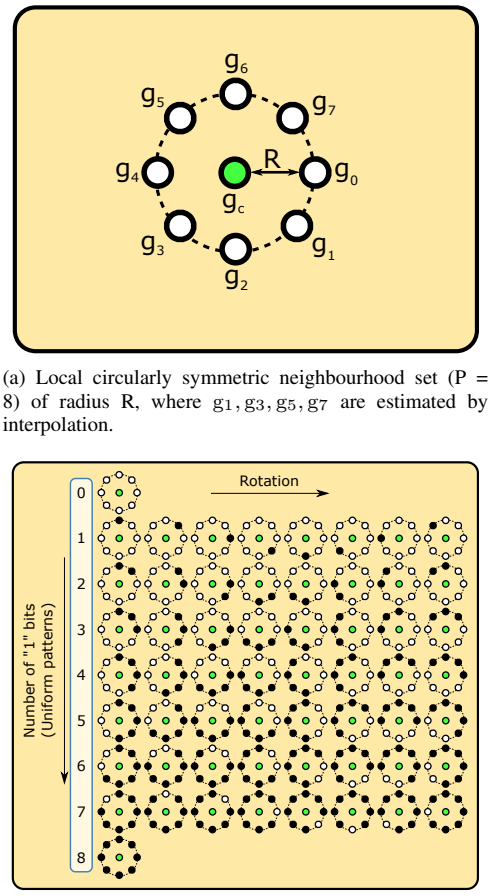
Fig. 1. Examples of the settlements classes found in Soweto

paired, so that the same location is extracted from both dates (images) [3].

2) *Extract LBP features*
We construct regular circular neighbourhoods with P $(\mathrm{P} > 1)$ image pixels and radius R $(\mathrm{R} > 0)$, with the coordinates of the gray values g_P being $(-R\sin(2\pi p/P))$,$(R\cos(2\pi p/P))$ at $g_c(0,0)$. Gray values that do not fall exactly in the centre pixel are estimated by interpolation, see figure 2a. From the circular neighbourhood we measure the LBPs using equation 3. We construct a look-up table that contains all the uniformity measures corresponding to the number of image pixels used, see figure 2b. Using the look-up table LBPs with uniform patterns are extracted. Uniformity measures U with the value of at most 2 are stored as uniform patterns with bin labels $(0 \rightarrow \mathrm{P}-1)$ while the non-uniform patterns are stored as bin label $(\mathrm{P}+1)$, where bins $0 \rightarrow P-1$ correspond to a texture feature from equation 5, see figures 2b and 2c.

3) *Extended LBP features extraction*
Using the circular neighbourhoods as mentioned above we calculate the Variance measure using equation 6. Since Variance measures have continuous outputs, quantization is needed. The bin breaks are constructed such that they are evenly spaced according to the variance percentiles, that is, by adding up all the feature distributions for every single model image in a total distribution and using R[1] we calculate the bin breaks for different number of bins ($3 \rightarrow 20$ bins in this case). We then constructed a simple 2D joint distribution histogram

[1]http://www.r-project.org/



(a) Local circularly symmetric neighbourhood set (P = 8) of radius R, where g_1, g_3, g_5, g_7 are estimated by interpolation.

(b) The Look Up Table (LUT): To achieve rotation invariance the LUT is used to store all the posible the uniform patterns to their unique code i.e., for P = 8, nine "uniform" patterns with the numbers (0 – 8) corresponding to their unique $\mathrm{LBP}^{\mathrm{riu2}}_{8,\mathrm{R}}$ codes.

(c) A sample of nonuniform patterns, all of which are labeled as code 9.

(d) Different texture primitives detected by the uniform patterns of LBP.

Fig. 2. Black and white circles correspond to bit values of 0 and 1 in the 8-bit pattern of the operator.

i.e LBP/VAR for each bin size for different P and R values. Multiresolution features are obtained by simply concatenating LBP features extracted at multiple radii (R parameter in equations 4 and 6).

4) *Training (Subset A) and Testing (Subset B)*
We determine the generalization performance for the different texture features algorithms by evaluating the

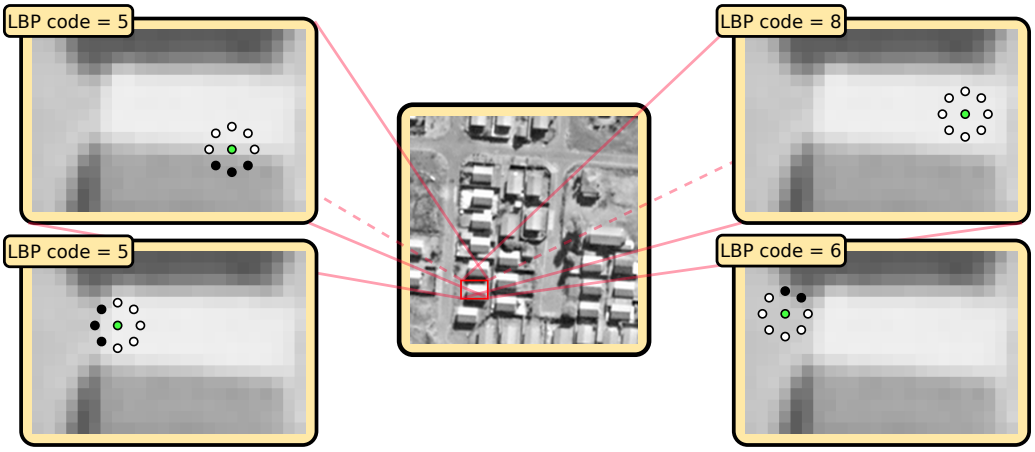


Fig. 3. Extraction of LBP features for P = 8.

performance of the Support Vector Machine (SVM) classifier over the six possible combinations (i.e. $A_{d1} \rightleftharpoons B_{d1}$, $A_{d2} \rightleftharpoons B_{d2}$, $A_{d1} \rightleftharpoons A_{d2}$, $A_{d1} \rightleftharpoons B_{d2}$, $B_{d1} \rightleftharpoons A_{d2}$, $B_{d1} \rightleftharpoons B_{d2}$ sets). We use Weka's[2] Sequential Minimal Optimization (SMO) algorithm for training the support vector classifier.

IV. Results and Discussion

In a comparative study investigating the best algorithm for settlement classification, the LBP algorithm showed excellent performance [4]. However, it did not perform well when it was tested for cross-date imagery over the same area as the study mentioned above [3]. A study extending the LBP by adding variance measures (contrast properties) for texture classification, showed the extended LBP to be very powerful. We repeated the cross-date study for LBP [3] and implemented this new extended LBP with variance measures for urban settlement classification where the details are reported in table I.

TABLE I
The number of patterns in each class, for each subset.

	FS	FSB	OIS	NBU	Total
Subset A	557	2820	2059	1358	6794
Subset B	496	3915	1969	1180	7560

To obtain the standard deviations on various classification results, the following procedure was used to evaluate a given configuration using data sets X and Y (where $X = A_{d_1}$ and $Y = B_{d_1}$):

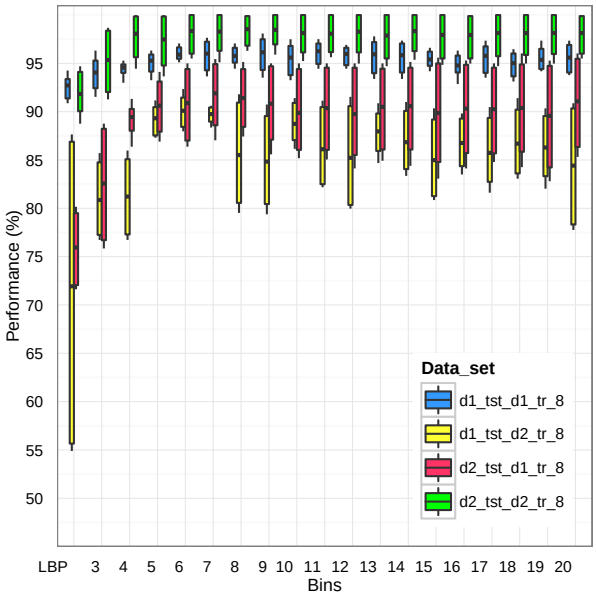
1. Train a support vector machine (SVM) using the whole of set X.
2. Partition set Y in 10 folds using stratified sampling to preserve related class frequency.
3. Evaluate the SVM (trained on X) on each of the 10 folds of Y, obtaining the one accuracy figure for each fold.
4. Exchange X and Y, and repeat 1–3.

This process, denoted by $X \rightleftharpoons Y$, produces 20 individual values for each accuracy metric, which were then used to calculate a mean and standard deviation for each metric. We distinguish between two classes of test, namely same-date (when both training and test sets are derived from the same-date satellite image) and cross-date (when two different satellite scenes were used). The difference in performance between these to classes highlights the degree to which a particular classifier is invariant to changes in shadow orientation and length.

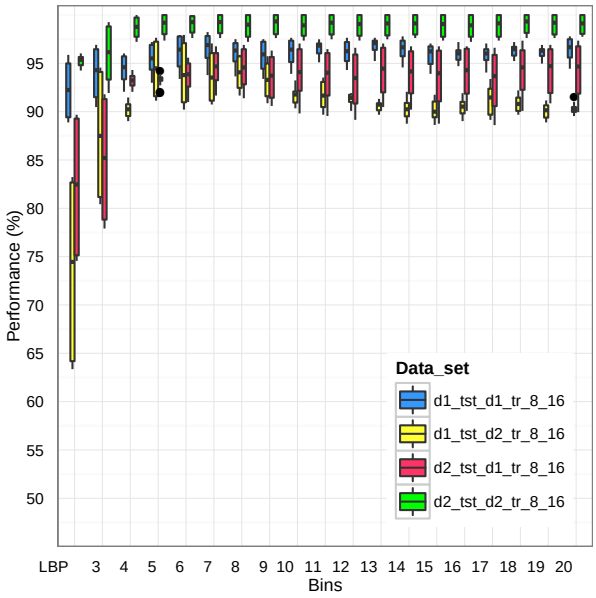
From figure 4 it is clear that the extended LBP outperforms the LBP with no variance measures, achieving accuracies close to 95% and more for same-date experiments (e.g., $A_{d1} \rightleftharpoons B_{d1}$). Even for cross-date data set (i.e., $A_{d1} \rightleftharpoons B_{d2}, A_{d2} \rightleftharpoons B_{d1}$) the $\mathrm{LBP}^{\mathrm{riu2}}_{\mathrm{P,R}}/\mathrm{VAR}_{\mathrm{P,R}}$ achieved close to 90%.
Overall performance is slightly improved by using the multiresolution features (see figure 4(a) and 4(b)), with the improvement being higher for $\mathrm{LBP}^{\mathrm{riu2}}_{8,1}/\mathrm{VAR}_{8,1} + \mathrm{LBP}^{\mathrm{riu2}}_{16,2}/\mathrm{VAR}_{16,2} + \mathrm{LBP}^{\mathrm{riu2}}_{24,3}/\mathrm{VAR}_{24,3}$ achieving accuracies close to 95% on cross-date performance.

A more detailed comparison of the LBP and extended LBP with multiresolution is reported in table II, where we can clearly see the effects of adding the variance measures, bin sizes, and multiple resolutions. In all cases the $\mathrm{LBP}^{\mathrm{riu2}}_{\mathrm{P,R}}/\mathrm{VAR}_{\mathrm{P,R}}$ features outperform the $\mathrm{LBP}^{\mathrm{riu2}}_{\mathrm{P,R}}$ features. The performance drastically improves with the increase of bin sizes (3–6) but then fluctuates slightly as the number of bins are increased. Table II also shows a slight improvement from $\mathrm{LBP}^{\mathrm{riu2}}_{\mathrm{P,R}}/\mathrm{VAR}_{\mathrm{P,R}}$ (8,1) to $\mathrm{LBP}^{\mathrm{riu2}}_{\mathrm{P,R}}/\mathrm{VAR}_{\mathrm{P,R}}$ (8,1 + 16,2) but for $\mathrm{LBP}^{\mathrm{riu2}}_{\mathrm{P,R}}/\mathrm{VAR}_{\mathrm{P,R}}$ (8,1 + 16,2 + 24,3) we observe slightly higher increase on accuracies. On all the figures in figure 4 we observed a peak around bin 7, and as the number of bins increased the performance was not meaningfully better. We then took bin 7 as the optimum number of bins and

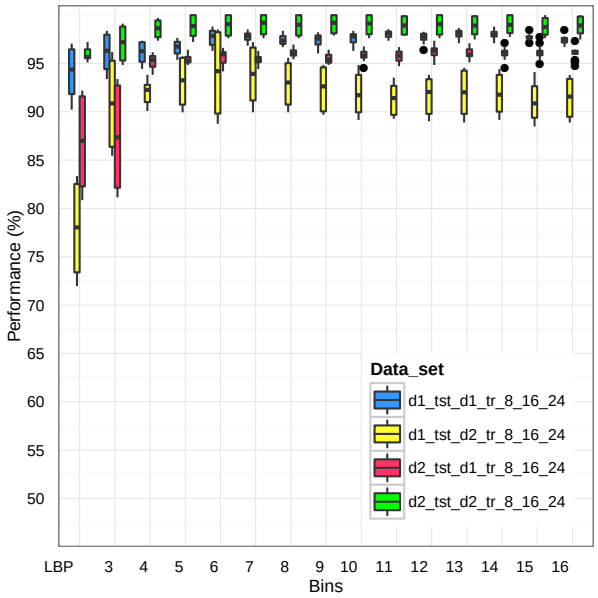
[2]www.cs.waikato.ac.nz/ml/weka/



(a) P,R = 8,1



(b) P,R = 8,1+16,2



(c) P,R = 8,1+16,2+24,3

Fig. 4. Boxplots showing the overall performance(%) for $\mathrm{LBP}^{\mathrm{riu2}}_{\mathrm{P,R}}$ (LBP) and $\mathrm{LBP}^{\mathrm{riu2}}_{\mathrm{P,R}}/\mathrm{VAR}_{\mathrm{P,R}}$ (Bins) over same date and cross date imagery with multiresolution data sets (b) and (c).

investigated its accuracies in terms of per-class true positive (TP) rate (see table III). It is clear that the performance of the algorithm is depended on the settlement type, where we see that for formal settlements with backyard shacks (FSB) and non-built up (NBU) classes the TP values are much higher than those of formal settlements (FS) and ordered informal settlements (OIS) classes. The standard deviations for $A_{d1} \rightleftharpoons B_{d2}$ are high in all cases except for the NBU class where its standard deviation is high at $A_{d1} \rightleftharpoons A_{d2}$. Table III also reveals that 100% classification accuracy was attained for certain classes, which is too good to be true, and calls for further investigation of the algorithms with a larger data set.

TABLE II
A SAMPLE OF THE RESULTS IN FIGURE 4 IN TABLE FORMAT, WHERE THE HIGHEST PERFOMANCE IS HIGHLIGHTED IN EACH DATA SET.

Data set	$\mathbf{LBP^{riu2}_{P,R}}$	$\mathbf{LBP^{riu2}_{8,1}/VAR_{8,1}}$						
	P,R=8,1	3	6	8	10	12	14	16 (bins)
$A_{d1} \rightleftharpoons B_{d1}$	92.44 (1.185)	93.91 (1.507)	**95.99 (0.708)**	95.83 (0.814)	95.36 (1.658)	95.75 (0.932)	95.53 (1.537)	94.85 (1.096)
$A_{d1} \rightleftharpoons A_{d2}$	78.48 (3.610)	84.80 (5.675)	93.30 (0.864)	92.81 (1.982)	**93.89 (1.259)**	92.85 (1.891)	93.67 (1.453)	93.77 (1.552)
$A_{d1} \rightleftharpoons B_{d2}$	71.44 (15.932)	81.09 (3.921)	**90.06 (1.745)**	85.68 (5.302)	88.89 (1.909)	85.50 (5.274)	87.02 (3.251)	86.78 (2.553)
$B_{d1} \rightleftharpoons B_{d2}$	78.44 (14.956)	79.81 (1.549)	87.89 (6.545)	88.08 (6.537)	87.70 (7.207)	88.42 (6.309)	88.25 (6.814)	**88.58 (6.372)**
$A_{d2} \rightleftharpoons B_{d1}$	75.90 (3.820)	82.45 (5.730)	**90.79 (3.757)**	91.29 (3.227)	90.11 (4.327)	89.82 (4.720)	90.20 (4.571)	90.12 (4.843)
$A_{d2} \rightleftharpoons B_{d2}$	91.99 (2.149)	95.22 (3.294)	98.02 (2.055)	**98.33 (1.656)**	98.00 (2.053)	98.01 (2.027)	98.08 (1.946)	97.73 (2.257)
	P,R=8,1+16,2	$\mathbf{LBP^{riu2}_{8,1}/VAR_{8,1} + LBP^{riu2}_{16,2}/VAR_{16,2}}$						
$A_{d1} \rightleftharpoons B_{d1}$	92.26 (2.831)	93.92 (2.624)	96.21 (1.625)	96.11 (1.141)	96.17 (1.358)	95.80 (1.033)	**96.51 (0.970)**	95.92 (0.715)
$A_{d1} \rightleftharpoons A_{d2}$	79.90 (6.158)	87.33 (5.677)	94.62 (1.088)	95.20 (0.699)	94.53 (1.148)	**95.89 (1.359)**	95.24 (1.277)	94.68 (1.445)
$A_{d1} \rightleftharpoons B_{d2}$	73.58 (9.492)	87.48 (6.647)	94.00 (3.186)	**94.01 (1.717)**	91.59 (0.761)	90.17 (0.987)	90.30 (0.962)	90.46 (0.800)
$B_{d1} \rightleftharpoons B_{d2}$	81.26 (12.159)	87.32 (2.321)	92.24 (2.311)	93.96 (1.207)	92.60 (2.442)	92.98 (2.227)	92.88 (2.608)	**93.99 (1.657)**
$A_{d2} \rightleftharpoons B_{d1}$	82.30 (7.146)	85.04 (6.422)	93.67 (1.498)	**94.51 (2.046)**	94.14 (2.461)	93.55 (2.978)	94.04 (2.370)	94.08 (2.505)
$A_{d2} \rightleftharpoons B_{d2}$	95.22 (0.544)	95.93 (2.999)	**98.99 (0.991)**	98.83 (1.178)	98.88 (1.143)	98.87 (1.111)	98.96 (1.070)	98.82 (1.204)
	P,R=8,1+16,2+24,3	$\mathbf{LBP^{riu2}_{8,1}/VAR_{8,1} + LBP^{riu2}_{16,2}/VAR_{16,2} + LBP^{riu2}_{24,3}/VAR_{24,3}}$						
$A_{d1} \rightleftharpoons B_{d1}$	94.03 (2.553)	96.16 (1.956)	97.64 (0.826)	97.45 (0.540)	97.61 (0.559)	97.69 (0.280)	**97.96 (0.433)**	97.39 (0.420)
$A_{d1} \rightleftharpoons A_{d2}$	86.50 (0.658)	92.05 (2.222)	**94.21 (1.594)**	93.92 (0.699)	93.96 (0.548)	94.16 (0.673)	93.41 (0.361)	94.05 (0.785)
$A_{d1} \rightleftharpoons B_{d2}$	77.95 (4.768)	90.85 (4.571)	**94.00 (4.330)**	92.89 (2.253)	91.85 (2.182)	91.05 (1.842)	91.79 (2.069)	91.44 (2.051)
$B_{d1} \rightleftharpoons B_{d2}$	79.43 (1.220)	92.57 (0.624)	96.36 (1.469)	**97.23 (1.561)**	95.74 (0.640)	96.61 (0.693)	96.31 (0.553)	96.76 (0.645)
$A_{d2} \rightleftharpoons B_{d1}$	86.83 (4.893)	87.36 (5.396)	95.67 (0.708)	96.09 (0.430)	95.85 (0.543)	**96.16 (0.636)**	96.04 (0.541)	96.09 (0.545)
$A_{d2} \rightleftharpoons B_{d2}$	96.00 (0.693)	97.05 (1.797)	98.93 (1.069)	98.89 (1.083)	98.98 (1.027)	98.81 (0.904)	**98.99 (0.973)**	98.94 (0.972)

TABLE III
OVERALL CLASSIFICATION ACCUARACY FOR MULTIRESOLUTION $\mathrm{LBP^{riu2}_{P,R}/VAR_{P,R}}$ AT LOWEST BIN SIZE THAT YIELDS OPTIMUM PERFORMANCE.

P,R	**Bins**	**Data set**	**Overall Accuracy(%)**	**FS TP(%)**	**FSB TP(%)**	**OIS TP(%)**	**NBU TP(%)**
8	7	$A_{d1} \rightleftharpoons B_{d1}$	95.782 (1.584)	87.170 (12.486)	96.330 (3.662)	96.060 (4.370)	100.00 (0.000)
		$A_{d1} \rightleftharpoons A_{d2}$	94.061 (0.885)	82.255 (20.412)	98.910 (0.485)	**86.870 (3.202)**	95.895 (4.277)
		$A_{d1} \rightleftharpoons B_{d2}$	89.658 (0.798)	76.580 (24.491)	97.615 (0.767)	65.725 (3.179)	99.900 (0.205)
		$B_{d1} \rightleftharpoons B_{d2}$	87.194 (7.331)	79.032 (20.347)	94.340 (3.273)	62.645 (37.457)	99.950 (0.154)
		$A_{d2} \rightleftharpoons B_{d1}$	91.641 (3.418)	79.412 (19.230)	95.675 (3.214)	82.310 (15.862)	97.905 (2.224)
		$A_{d2} \rightleftharpoons B_{d2}$	98.029 (2.050)	**96.817 (5.791)**	**99.985 (0.067)**	90.085 (10.299)	100.00 (0.000)
8+16	7	$A_{d1} \rightleftharpoons B_{d1}$	96.643 (1.343)	88.608 (10.171)	96.535 (3.387)	**99.380 (0.763)**	100.00 (0.000)
		$A_{d1} \rightleftharpoons A_{d2}$	**95.244 (0.598)**	**91.088 (12.430)**	99.175 (0.505)	85.685 (5.515)	96.220 (3.924)
		$A_{d1} \rightleftharpoons B_{d2}$	93.680 (2.471)	**83.285 (18.089)**	94.890 (1.558)	**87.390 (10.179)**	99.900 (0.205)
		$B_{d1} \rightleftharpoons B_{d2}$	92.882 (1.954)	82.093 (16.429)	96.550 (0.839)	84.915 (14.865)	99.950 (0.154)
		$A_{d2} \rightleftharpoons B_{d1}$	94.572 (1.676)	80.125 (18.795)	98.275 (1.166)	91.720 (7.774)	98.350 (1.913)
		$A_{d2} \rightleftharpoons B_{d2}$	**99.054 (0.987)**	96.537 (5.980)	99.850 (0.199)	97.255 (2.874)	100.00 (0.000)
8+16+24	7	$A_{d1} \rightleftharpoons B_{d1}$	**97.765 (0.476)**	**90.170 (10.245)**	**98.650 (1.402)**	99.080 (1.123)	100.00 (0.000)
		$A_{d1} \rightleftharpoons A_{d2}$	94.312 (1.702)	83.343 (17.415)	**99.890 (0.180)**	82.685 (14.843)	**97.100 (3.007)**
		$A_{d1} \rightleftharpoons B_{d2}$	**93.842 (2.980)**	76.737 (22.774)	**98.735 (1.323)**	**85.930 (14.559)**	**99.950 (0.154)**
		$B_{d1} \rightleftharpoons B_{d2}$	**96.107 (0.904)**	**85.737 (18.993)**	**97.970 (0.683)**	**95.630 (3.977)**	**99.975 (0.112)**
		$A_{d2} \rightleftharpoons B_{d1}$	**95.398 (0.482)**	**83.028 (18.519)**	**98.530 (1.249)**	**92.940 (2.316)**	**99.470 (0.983)**
		$A_{d2} \rightleftharpoons B_{d2}$	98.995 (1.047)	95.340 (7.982)	99.665 (0.367)	98.440 (1.693)	100.00 (0.000)

V. CONCLUSION

This paper presented a settlement classification experiment involving two scenes of the same area, acquired under different conditions, including seasonal changes in vegetation and the length and orientation of shadows cast by the buildings. The results indicate that adding the rotational invariant variance measure to the rotational and gray-scale invariant LBP does improve performance in classifying settlement types in urban areas. We can then conclude that contrast properties are significant in the task of classifying settlement type. Some differences remain between the classification performance in same-date experiments vs cross-date experiments; this is not entirely unexpected, since the addition of the contrast features is unlikely to provide robust invariance to the influence of

shadows. The good news, however, is that the gap between same-date and cross-date classification performance closed somewhat with the addition of contrast features, rather than widen.

Improvements to image features that result in better classifier generalization performance brings us one step closer towards operational implementation of a fully automated settlement type classification system. Once such a system has achieved adequate classification accuracy, the goal of automated change detection in urban areas is within reach.

REFERENCES

[1] D. Maktav, F. S. Erbek, and C. Jürgens, "Remote sensing of urban areas," *International Journal of Remote Sensing*, vol. 26, no. 4, pp. 655–659, 2005.

[2] J. Benediktsson, M. Pesaresi, and K. Amason, "Classification and feature extraction for remote sensing images from urban areas based on morphological transformations," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 41, no. 9, pp. 1940–1949, 2003.

[3] F. van den Bergh, "The effects of viewing- and illumination geometry on settlement type classification of quickbird images," in *Geoscience and Remote Sensing Symposium (IGARSS), 2011 IEEE International*, July 2011, pp. 1425 –1428.

[4] L. Ella, F. van den Bergh, B. van Wyk, and M. van Wyk, "A comparison of texture feature algorithms for urban settlement classification," in *Geoscience and Remote Sensing Symposium, 2008. IGARSS 2008. IEEE International*, vol. 3. IEEE, 2008.

[5] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, no. 7, pp. 971 –987, July 2002.

[6] S. Arivazhagan and L. Ganesan, "Texture classification using wavelet transform," *Pattern Recognition Letters*, vol. 24, no. 910, pp. 1513 – 1521, 2003. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167865502003902

[7] G. Rellier, X. Descombes, F. Falzon, and J. Zerubia, "Texture feature analysis using a Gauss-Markov model in hyperspectral image classification," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 42, no. 7, pp. 1543–1551, 2004.

[8] M. Pesaresi and J. Benediktsson, "A new approach for the morphological segmentation of high-resolution satellite imagery," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 39, no. 2, pp. 309–320, 2001.

[9] J. Zhang and T. Tan, "Brief review of invariant texture analysis methods," *Pattern Recognition*, vol. 35, no. 3, pp. 735–747, 2002.

On the rendering of synthetic images with specific point spread functions

F. van den Bergh

Remote Sensing Research Unit, Meraka Institute
CSIR, PO Box 395, Pretoria
South Africa, 0001
fvdbergh@csir.co.za

***Abstract*—Most image processing and machine vision algorithms are evaluated on synthetic images, usually of known target patterns, to determine their effectiveness under controlled conditions. Such synthetic images are often rendered using an area-weighted strategy, which implies that the point spread function (PSF) of the simulated optical system is a box function. This paper discusses several rendering strategies that can be employed to extend the generation of synthetic images to more general point spread functions. In particular, high-accuracy algorithms for rendering Gaussian and circular aperture diffraction PSFs are presented.**

I. Introduction

Machine vision algorithms are typically hard to implement correctly because small errors in the implementation may not necessarily lead to easily observable errors in the output. To guard against such implementation errors it is prudent to test the algorithm under controlled conditions. This usually requires synthetic images with known properties, such as dynamic range, signal-to-noise ratio, noise distribution, and optical system point spread function.

A quick review of the literature will reveal that many such experiments simplify the synthetic image generation process by assuming that the noise is additive in nature, with a Gaussian distribution, and that the optical system PSF can be approximated as a Gaussian. These assumptions make for an efficient implementation, especially if a Gaussian blur is used to simulate the effect of the PSF at the target resolution of the synthetic image. Although these assumptions are not inherently poor for the evaluation of many machine vision algorithms, it is desirable to have more realistic simulation methods available for evaluating those methods that require greater accuracy in PSF and noise simulation.

A few algorithms can be tested rigorously using relatively simple test images. One example is the slanted edge algorithm that estimates optical system resolution by computing the Modulation Transfer Function (MTF) of a knife-edge target [1]. For this algorithm the synthetic image can be a simple step function in intensity, with the edge rendered at a specific angle, and with a known PSF. Other examples include super-resolution methods, where multiple low-resolution images are combined to construct a higher resolution image [2]. These algorithms can be evaluated by presenting them with synthetic images of simple geometric shapes (e.g., black polygons on white backgrounds), and measuring the resolution of the super-resolved output using the slanted edge algorithm mentioned above. Lastly, the accuracy of algorithms designed to extract simple features from images, such as rectangle-, circle- or ellipse-detection algorithms [3] can be evaluated on simple synthetic images consisting of black polygons on white backgrounds.

In all of the above cases the algorithms are best evaluated on synthetic images where the PSF closely matches the PSF of the expected real-world application, which typically requires modeling at least lens aperture diffraction and photosite aperture effects. Surprisingly, details on rendering synthetic images with PSFs that accurately capture the desired properties are not often included in papers relying on such synthetic images for validation experiments.

This paper discusses several algorithms that may be used to render synthetic images with specific point spread functions, focusing on some common PSFs, including a box function PSF, a Gaussian PSF, a circular aperture diffraction PSF, and a birefringent crystal optical low pass filter PSF.

II. Background

A. Point spread functions

The *point spread function* describes the finite impulse response of an imaging system, in other words, the response when imaging a point source. The PSF is defined in the spatial domain, and has a frequency domain analogue that is called the *modulation transfer function* (MTF), which can be obtained via the Fourier transform.

Unless the PSF is itself an impulse function, it will distribute the light originating from a point source over a region with non-zero area. Visually, this spreading is perceived as blur; for a point source, we will observe a larger blob. If the true object being imaged is not a point source, the effect of the PSF may be more complex in appearance. In practice, the interaction of the PSF and the discrete sensing elements (photosites) of a digital image sensor can be thought of as "placing the PSF at the centre of each photosite, and weighting the light coming from the true object according to the PSF". The resulting image is thus the convolution of impulse functions placed at the photosite centres, the PSF, and the true object.

If the PSF is *shift invariant*, i.e., identical across the focal plane, then the convolution can be implemented efficiently in the Fourier domain as the product of the Fourier transform of the true object and the MTF, followed by an inverse Fourier transform to return to the spatial domain.

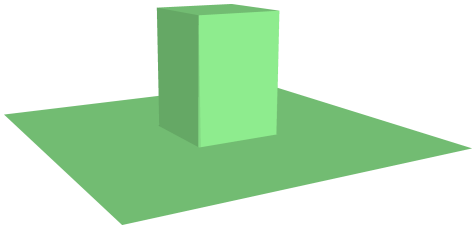


Fig. 1. Box function PSF

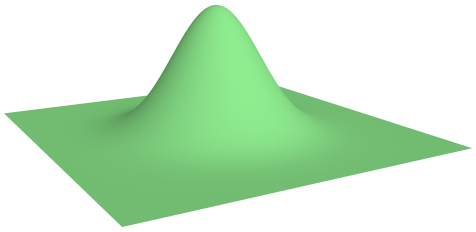


Fig. 2. Gaussian PSF

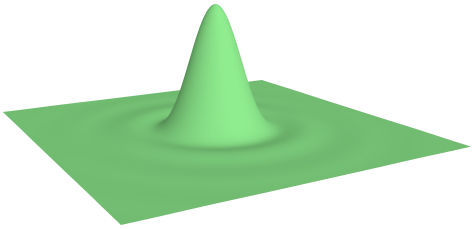


Fig. 3. Circular aperture diffraction PSF

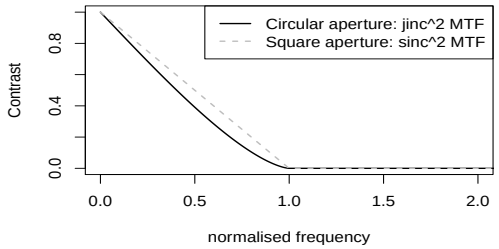


Fig. 4. Circular aperture diffraction MTF

B. Common point spread functions

Some of the commonly occurring point spread functions follow:

1) Box PSF: The box function PSF corresponds to the rectangular photosites of a matrix sensor. As such, any image formed by a matrix sensor will ultimately involve a convolution step with the box function. A fill factor of less than 100% may imply that the effective box function is narrower than the photosite pitch, and a non-square photosite geometry (e.g., L-shaped) may be in effect; both these factors can be modelled if desired by piecewise decomposition into multiple smaller box functions.

The 2D box function is visualised in Figure 1. Fortunately, the 2D box function PSF lends itself to a highly efficient implementation when rendering polygon shapes: each photosite's intensity is simply proportional to the area of the pixel covered by the target polygon. This intersection can be computed using the Sutherland-Hodgman polygon clipping algorithm [4], for example.

2) Gaussian PSF: The Gaussian PSF is often used to introduce a blur effect into synthetic images. Although the Gaussian PSF does not correspond to any common physical phenomenon, it does serve as a coarse approximation to diffraction effects. The primary reason for its popularity appears to be ease of implementation and use.

A 2D Gaussian PSF is shown in Figure 2. Although a direct implementation of this PSF is straightforward, it is rather more involved to obtain highly accurate synthetic images; one such method is discussed below in Section III-D.

3) Circular aperture diffraction PSF: Light passing through a circular aperture is affected by Fraunhofer diffraction to produce a light intensity distribution known as the Airy pattern [5]. The width of this pattern is inversely proportional to the diameter of the aperture, thus smaller apertures produce wider Airy pattern point spread functions. For incoherent light, the Airy pattern is defined as

$$I_0 \cdot \left(\frac{2J_1(x)}{x} \right)^2 \tag{1}$$

where J_1 is the Bessel function of the first kind, of order one, and I_0 represents the peak intensity. Note that $x = \frac{\pi q}{\lambda N}$, where λ is the wavelength of the light, N is the aperture f-number, and q is the radial distance from the axis passing through the centre of the aperture. This PSF is illustrated in Figure 3. The concentric side-lobes of the pattern are barely discernible after the second cycle, however, the support of the Airy pattern is infinite, and the side-lobes never quite reach zero.

The Fourier transform (or Modulation Transfer Function, MTF) of the Airy pattern is the *Chinese hat* function:

$$\mathrm{chat}(s) = \frac{2}{\pi} \left(\cos^{-1}(s) - s\sqrt{1 - s^2} \right) \tag{2}$$

where $0 < s \leq 1$ represents the *normalised spatial frequency*, which is defined such that $s = \lambda N f$, with f denoting unnormalised frequency. This function is illustrated in Figure 4, which clearly shows that the Airy pattern PSF acts as a low-pass filter.

The Airy pattern is of particular importance to the rendering of synthetic images produced by a lens, since even in the absence of a physical aperture stop the lens itself acts as an aperture. For a wavelength of 550 nm and a photosite pitch of 5 micron, diffraction will reduce system resolution for apertures with an f-number greater than f/5.6. For even smaller photosite pitch values, this maximum allowed f-number must be decreased even further to prevent loss of resolution owing to diffraction.

In the frequency domain, the Airy pattern MTF reaches exactly zero and remains at zero beyond the critical frequency $f = \frac{1}{\lambda N}$. This property is poorly approximated by a Gaussian PSF (which also has a Gaussian MTF), which does not decay quite as rapidly as the Airy pattern MTF. Should one wish to approximate the Airy pattern with a Gaussian regardless of

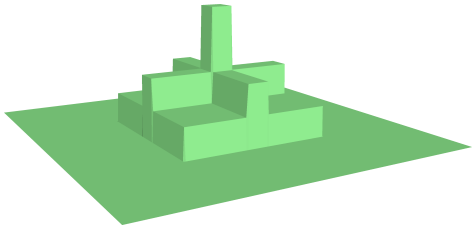


Fig. 5. 4-dot birefringent OLPF PSF with a 0.75 photosite pitch displacement

its limitations, the best-fitting Gaussian approximation in the least-squares sense can be obtained by choosing the standard deviation to be $\sigma \approx 0.425\lambda N$.

4) Birefringent OLPF PSF: An optical low-pass filter (OLPF) can be used to suppress the power at frequencies above the Nyquist limit for a given photosite pitch. Strictly speaking, this is a requirement to ensure correct sampling, and aliasing artifacts may appear in images captured with an optical system that lacks an OLPF. If the lens aperture is chosen carefully with respect to the photosite pitch, it is possible to employ diffraction to act as a low-pass filter, but this approach is not practical for larger photosite pitches (e.g., larger than 5 micron) when used with large relative apertures. If a Bayer colour filter array is included in the sensor design, it becomes even more important to minimise aliasing, which may manifest as colour interpolation errors.

One method of constructing an OLPF is through the use of a birefringent material, i.e., a crystal that forces photons to take different paths depending on their polarization. One such material is Lithium Niobate, which can be used to split an unpolarised beam into one horizontally polarised beam and one displaced parallel beam containing only the vertically polarised photons [6]. If an image passes through such a filter, the resulting image leaving the filter will be the superposition of the image and copy of the image displaced by a distance d, which effectively blurs the image in the direction of the displacement. Two such filters can be combined (with an appropriate depolariser in between) to effect a blur in both directions.

A 4-dot birefringent OLPF PSF is illustrated in Figure 5. The exact shape of the PSF is dependent on the displacement, d, effected by the birefringent plates. In general, it is desirable to choose the displacement as a function of the photosite pitch so that the filter cut-off frequency is related to the Nyquist frequency of the sensor.

An implementation of the 4-dot OLPF for synthetic image rendering is a straightforward extension of the method used for a box function PSF. The process is simply repeated four times with four displaced box function PSFs.

C. Combining point spread functions

As already alluded to above, the system PSF is a combination of the individual PSFs encountered along the optical path. Provided that phase effects can be ignored, such as when light passes from the lens onto the sensor, the PSFs can be combined by direct convolution. Equivalently, the MTFs of the various components along the optical path can simply be multiplied to obtain the system MTF. This approach can be used to combine the lens (diffraction) response, OLPF response (if present) and photosite aperture response to obtain the system response. Unfortunately, the effects of defocus cannot be integrated with this approach, and are therefore not considered in the sequel.

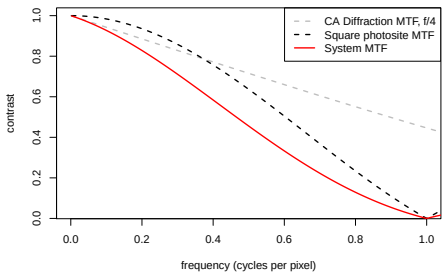


Fig. 6. Diffraction, photosite aperture and combined MTF

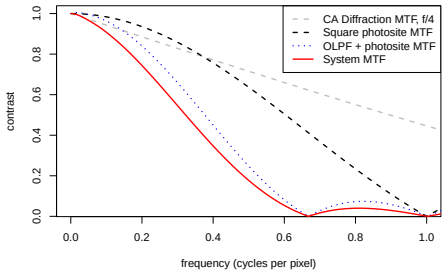


Fig. 7. Diffraction, Photosite aperture, OLPF and combined MTF

Some useful combinations, corresponding to typical configurations encountered in real optical systems, will now be considered.

1) Circular aperture diffraction + photosite aperture: This system corresponds to a monochromatic matrix sensor and lens combination. It is also appropriate for Bayer CFA sensors that do not contain an OLPF. The MTFs of the components, as well as the combined system MTF, are illustrated in Figure 6.

2) Circular aperture diffraction + 4-dot OLPF + photosite aperture: This configuration is common for large-photosite Bayer CFA systems, such as commercial Digital Single Lens Reflex (DSLR) cameras. The OLPF helps to suppress colour interpolation artifacts as well as regular aliasing artifacts. An example of an OLPF MTF curve is shown in Figure 7, using a beam separation distance $d = 0.75$ pixels. This effectively attenuates the system response strongly at frequencies above 0.67 cycles per pixel, but does not completely eliminate power above Nyquist (0.5 cycles per pixel).

III. Rendering strategies

Several strategies for rendering synthetic images will now be discussed. To simplify the discussion, it will be assumed that the target object is a black polygon rendered against a

white background. Furthermore, it is assumed that the edges of the target object are perfect step functions.

One of two basic operations are required to implement the proposed rendering strategies: an indicator function operator, and a polygon-polygon intersection operator. The indicator function operator returns a value of 1 if its argument is inside the target polygon, and 0 otherwise. The polygon-polygon intersection operator returns a real number representing the area of the polygon formed by the intersection of the two polygon arguments to the operator.

Both of these operators can be implemented reasonably efficiently for polygon target objects. A simple point inclusion operator can be defined for some non-polygonal target objects, such as circular and ellipsoidal discs, but these target shapes can be approximated as polygons to the required accuracy if necessary.

All the strategies presented below are attempts to compute the integral that results when convolving the target object indicator function and the desired PSF. Since the extent of the target object is finite, it is convenient to express the intensity of the pixel at location (x, y) in the synthetic image as an integral over the target object, i.e.,

$$I_{(x,y)} = \int_{\mathbb{R}^2} \mathbf{1}_P(\mathbf{x}) f_{(x,y)}(\mathbf{x}) \mathrm{d}\mathbf{x} \tag{3}$$

where $\mathbf{1}_P(\mathbf{x})$ denotes the indicator over polygon P, and $f_{(x,y)}$ represents the PSF centered at location (x, y). This can be simplified to

$$I_{(x,y)} = \int_P f_{(x,y)}(\mathbf{x}) \mathrm{d}\mathbf{x} \tag{4}$$

by restricting the integral to the region bounded by the polygon P, when appropriate.

Except for the box function PSF, approximate solutions to these integrals must be obtained using numerical integration methods. When the PSF itself is the result of the convolution of simpler PSFs, e.g., the combined effect of a square photosite aperture and circular aperture diffraction, the problem is compounded because the PSF itself becomes another integral to be approximated. As is often the case, Monte Carlo integration methods are a convenient way of computing these integrals.

A. Uniform oversampling

Using the indicator function, the synthetic image can be rendered by generating a set of sampling points coinciding with the centre of each pixel in the synthetic image. Each of these points can then be tested against the indicator function to determine whether the sample falls inside the target object, or not, colouring the resulting pixel accordingly.

This strategy is computationally efficient, but leads to severe aliasing, visible as "stair steps" along the edges of the target. The aliasing is due to the low sampling rate, at one sample per pixel, compared to the infinite bandwidth required to render the edge correctly. Two straightforward extensions can be employed to mitigate the aliasing: 1) render the synthetic image at a higher resolution, followed by downsampling to the desired resolution, or 2) oversampling on a uniform grid with sub-pixel spacing (Figure 8).

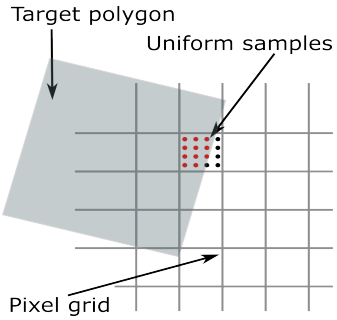


Fig. 8. Uniform oversampling using box PSF indicator function

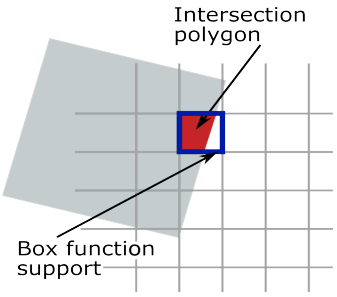


Fig. 9. Area-weighted rendering by polygon intersection

These two oversampling strategies can produce identical results, but the first strategy is computationally more complex, and requires significantly more memory. The additional samples should be weighted according to a properly scaled (spatially) grid of weights representing the desired PSF. Both these strategies introduce distortion of lower frequencies if the PSF is not band limited, i.e., if the support of the PSF is infinite, like in the case of a Gaussian PSF or an Airy pattern PSF. This error is bounded, and clearly an approximation can be constructed at any desired accuracy.

B. Area-weighted sampling

The box function PSF presents a special case for which an exact solution can be obtained efficiently. Note that the support of the box function is finite, with its extent typically being a square with sides equal to the photosite pitch, and that the function is constant over the region where it is non-zero. The result of convolving a box function placed at a given pixel centre and the target polygon is proportional to the area of intersection between the target polygon and the box function's support (Figure 9).

C. Gaussian PSF importance sampling

The Gaussian PSF has infinite support, which implies that any point-based sampling strategy must inherently introduce some error. A naive approach to rendering a synthetic image with a Gaussian PSF would be to use the uniform oversampling strategy (Section III-A), and choosing the individual sample weights from the desired Gaussian function. This strategy has two significant weaknesses: 1) the PSF will be truncated at the boundary of the uniform sampling grid, and 2) the samples that fall in the tails of the Gaussian PSF will

contribute little to the overall integral, yet they outnumber the samples in the central region of the Gaussian where the weights are much larger.

A much better strategy is to compute the convolution integral using Monte Carlo sampling. In particular, *importance sampling* strategies allow us to sample the PSF according to its actual density [7, section 7.6]. If we wish to approximate the integral I over the volume V, then importance sampling reformulates the problem as

$$I \approx \frac{1}{N}\sum_{i=0}^{N-1} f(\mathbf{x}_i) = \frac{1}{N}\sum_{i=0}^{N-1}\frac{f(\mathbf{x}_i)}{p(\mathbf{x}_i)} \tag{5}$$

where $f(x_i)$ represents the integrand, and $p(x_i)$ the probability of sampling point x_i. It is assumed that $\int p(x)\mathrm{d}x = 1$. The benefit of importance sampling is that we can choose a distribution $p(x)$ that is easily invertible, but matches $f(x)$ as closely as possible. Uniform oversampling is simply a special case of importance sampling where all points on the uniform grid are equally likely, and happen to be uniformly spaced.

The sampling strategy is thus to generate a set of sampling positions that follow a chosen distribution $p(x)$, a method known as *inverse transform sampling* [7, section 7.2]. Let F denote the cumulative distribution function of $p(x)$. Then $F^{-1}(U) \sim F$, where U is a uniform variate in the range $[0, 1]$. Thus, starting from a uniform variate u in the range $[0, 1]$, we can obtain a sample x with distribution $p(x)$ by transforming u as $x = F^{-1}(u)$.

This method does not require an analytical form for F^{-1}; a table-based inversion or a polynomial approximation is often adequate. When rendering a Gaussian PSF, we choose to distribute x as $x \sim N(0, \sigma)$, which can be achieved through Moro's inversion [8]. Since we can choose the standard deviation σ to exactly match the desired Gaussian PSF, and generate x with the exact same distribution, we can simplify Equation 5 to

$$\begin{aligned} I &\approx \frac{1}{N}\sum_{i=0}^{N-1}\frac{f(\mathbf{x}_i)}{p(\mathbf{x}_i)}\mathbf{1}_P(\mathbf{x}_i) \\ &\approx \frac{1}{N}\sum_{i=0}^{N-1}\mathbf{1}_P(\mathbf{x}_i), \end{aligned} \tag{6}$$

since $f(\mathbf{x}_i) = p(\mathbf{x}_i)$. If the sampling distribution of $\mathbf{x}_i$ matches the PSF exactly, then the samples $\mathbf{1}_P(\mathbf{x}_i)$ should not be weighted by the PSF at $\mathbf{x}_i$, in contrast to the uniform grid sampling method.

Importance sampling naturally distributes the sampling points according to the weight of the PSF (a Gaussian, in this case), which implies that more samples will be taken close to the centre of the PSF where the relative weight is large. This in turn reduces the variance of the Monte Carlo integral I, which reduces the number of samples required to reach a specified level of accuracy. In addition, the inverse transform sampling method can theoretically generate points in the far tails of the Gaussian, which implies that the PSF is not artificially truncated at a certain size. This minimises the distortion of lower frequencies associated with a fixed-size uniform sampling grid.

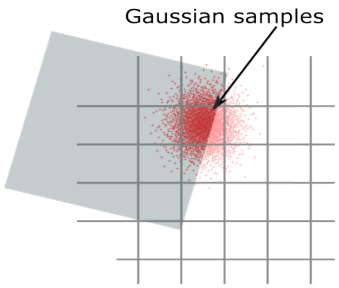


Fig. 10. Importance sampling with a Gaussian distribution

An efficient implementation of this importance sampling approach is to pre-compute the values of $\mathbf{x}_i$ using a Gaussian centered at $(0, 0)$. The sampling positions $\mathbf{x}_i$ can then translated to the pixel centred at $\mathbf{p} = (x, y)$, thereby avoiding the need to recompute sampling points for each pixel (Figure 10).

D. Gaussian PSF numerical integration

An alternative integration technique is applicable to Gaussian PSFs if an acceptably accurate approximation to the error function $\mathrm{erf}(x)$ is available. Starting from Equation 4, the polygon is partitioned into horizontal strips. In the limit, an infinitely thin strip reduces to the one-dimensional integral along the line $y = y_c$:

$$I_{y_c} = \int_{P_l(y_c)}^{P_r(y_c)} f(x)\mathrm{d}x \tag{7}$$

where $P_l(y_c)$ and $P_r(y_c)$ denote the left and right x values of the intersections of the polygon P with the line $y = y_c$. This definition only allows for convex polygons, but the extension to concave polygons will be analogous to that used to rasterise concave polygons.

The $\mathrm{erf}(x)$ function can be harnessed to derive a closed form solution to the integral in Equation 7, to yield

$$I_{y_c} = \mathrm{erf}(P_r(y_c)) - \mathrm{erf}(P_l(y_c)), \tag{8}$$

assuming that appropriate standardisation has been applied to $P_r(y_c)$ and $P_l(y_c)$.

Equation 8 provides a closed-form solution to the integral along any given horizontal slice through the polygon P. This allows us to perform numerical integration, using the adaptive version of Simpson's method, to compute the integral over all of P by integrating over the range of y values spanned by P. Figure 11 illustrates the integral that is computed for a wide Gaussian PSF centered at a pixel close to the boundary of a square target pattern. This method supports general Gaussian PSFs, including astigmatic Gaussian PSFs with full covariance matrices. If the PSF's axes are rotated with respect to the reference frame, then the simplest strategy is to rotate the target polygon to ensure that cross-sections along the integration axes are separable.

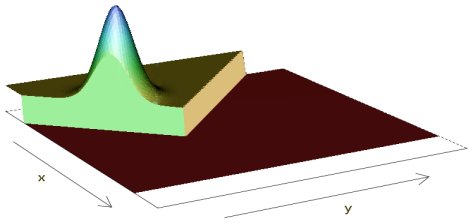


Fig. 11. Gaussian PSF bounded by target polygon. Area under curve is desired image intensity for pixel at centre of Gaussian peak

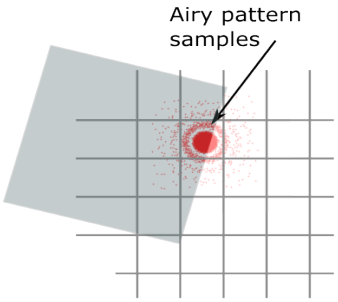


Fig. 12. Importance sampling with an Airy pattern distribution

This particular method can be exceptionally accurate, depending on the parameters of the adaptive numerical integration routine. It is possible to choose these parameters so that the computational complexity is comparable to that of the importance sampling rendering method, but yielding higher accuracy synthetic images.

E. Diffraction + box function importance sampling

Equation 3 is appropriate for rendering simple point spread functions, but does not address compound PSFs, such as the system PSF of a square photosite aperture PSF combined with a circular aperture diffraction pattern PSF. It is possible to perform the convolution of these two PSFs as a preprocessing step, thereby obtaining a single PSF which could be used in a table-driven importance sampling scheme.

A more elegant solution is to combine the area-weighted rendering strategy directly with the importance sampling scheme. Consider the set of sampling positions generated from a Gaussian distribution, as described in Section III-C. Rather than computing the Monte Carlo integral of this Gaussian PSF convolved with the target polygon indicator function, we can replace the indicator function test with a step that computes the area of the intersection of the target polygon and a square polygon (with photosite pitch side lengths) placed at each sampling position. This process thus performs the convolution of the target polygon and the photosite aperture box function first, using this result to compute the Gaussian PSF convolution using importance sampling.

To extend this method to a circular aperture diffraction PSF, we simply replace the Gaussian-distributed sampling positions with samples following the appropriate Airy pattern distribution (Figure 12). The Airy pattern distribution of samples is obtained through a look-up table that approximates the cumulative Airy pattern distribution.

F. Diffraction + OLPF importance sampling

The method described in Section III-E can be extended to render the effects of a 4-dot OLPF. Rather than computing the intersection of a single square with the target polygon at each sampling position, we instead compute the average of four such intersections, with each square placed at the appropriate offset as defined in the OLPF's specification. This approach is, of course, four times more computationally expensive.

G. Spectral sampling

Diffraction effects are wavelength dependent, which may have significant implications on computational complexity if wide-band panchromatic systems are to be simulated, since the most accurate simulation would involve rendering and blending synthetic images at multiple wavelengths, and combining them with the appropriate spectral-response weighting. Simulation of synthetic images intended for algorithms running on a Bayer Colour Filter Array (CFA) sensor (which covers most colour cameras) would require rendering at least three separate synthetic images (one for each band), possibly more if the colour filters are particularly wide. Fortunately, many algorithms (e.g., ellipse-detectors) can be verified at a single wavelength.

IV. Performance Evaluation of Rendering Strategies

A. Comparison of Gaussian PSF rendering accuracy

The following rendering algorithms were tested:

- $11 \times 11\times$UP is a uniform sampling strategy of 11×11 points centered around the target pixel. The sampling positions are truncated to the nearest integer to represent a standard linear filter without any sub-pixel sampling. This is equivalent to applying a Gaussian filter after rendering the synthetic image with one sample per pixel.
- $11\times11\times$U is a uniform sampling strategy of 11×11 points, but the sampling points are scaled relative to the desired Gaussian width. Sub-pixel spacing is used.
- $121\times$IS is an importance sampling method, with 121 (i.e., 11×11) samples from the same Gaussian distribution as that specified in the PSF. Sub-pixel spacing is used.
- $2025\times$IS is an importance sampling method, with 2025 samples drawn from the same Gaussian distribution as that specified in the PSF. Sub-pixel spacing is used.
- NI is a numerical integration implementation relying on an adaptive version of Simpson's rule (Section III-D).

These algorithms were evaluated over a range of images with Gaussian PSFs. Different standard deviation values were selected to evaluate performance over both small and large (relative to pixel size) PSFs. In addition, the sub-pixel position of the step edge was varied over 25 sub-pixel offsets to produce a more accurate assessment of algorithm performance.

The MTF50 metric is defined as the resolution at which the MTF curve reaches a contrast value of 50%, and is generally considered as a measure of resolution that correlates well with subjective human judgement of the sharpness of an image. For a Gaussian PSF, the relationship between MTF50 and standard

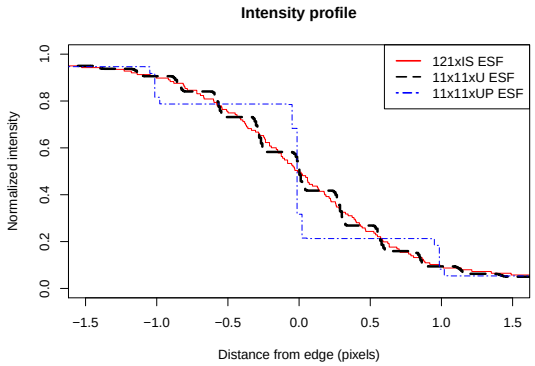


Fig. 13. Edge spread functions of 121×IS, 11 × 11×U and 11 × 11×UP rendering methods.

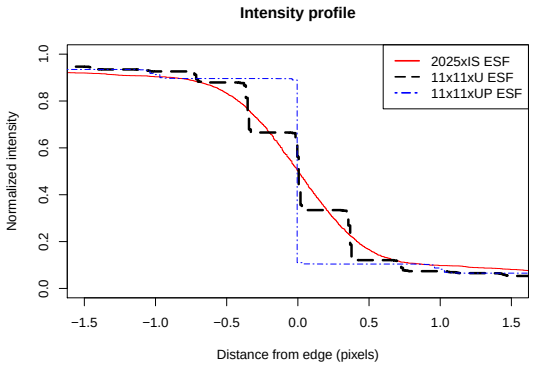


Fig. 14. Edge spread functions of 2025×IS, 11 × 11×U and 11 × 11×UP rendering methods.

deviation is fixed, hence standard deviation may be expressed as an MTF50 value in cycles per pixel. The range investigated in Table I runs from MTF50=0.1 (equivalent to Gaussian SD of 1.874) to MTF50=0.4 (equivalent to an SD of 0.468).

Figure 13 illustrates the intensity profile across the step edge subject to a Gaussian PSF with SD=0.625 (MTF50=0.3), rendered using the 11 × 11×U and 121×IS algorithms. None of the curves are smooth (compared to the expected exact Gaussian integral), but it is clear that the importance sampling algorithm is significantly closer to the desired curve (not shown). Table I confirms that the RMSE of the importance sampling algorithm is roughly 4 times smaller than the uniform grid sampling algorithm for the illustrated case, and that the integer-pixel grid uniform sampling algorithm (11 × 11×UP) fails miserably with such narrow PSFs.

The edge profile of the direct numerical integration algorithm (NI) is so accurate that it differs from the expected analytical profile only in the least significant bit of the 16-bit values used to represent intensities, i.e, differences are of the same magnitude as potential rounding errors. This rendering algorithm is therefore suitable for creating reference images.

To assess the impact of PSF accuracy on a real-world application, the slanted-edge algorithm was used to evaluate the MTF50 values of the various synthetic images. The results are shown in Table II. Even though the RMS errors of the NI method were significantly smaller than those of the 121×IS and 2025×IS algorithms, it appears that this does not translate into smaller errors in the MTF50 values as measured by the slanted-edge algorithm. One potential explanation is that the slanted edge method is more sensitive to lower spatial frequencies, so that the apparent roughness of the 121×IS algorithm (seen in Figure 13) manifests mostly at frequencies above Nyquist. The result is that additional accuracy in the PSF (as offered by the 2025×IS and NI algorithms) offers no real-world advantage for the slanted-edge algorithm.

B. Comparison of Airy pattern PSF rendering accuracy

The accuracy of the algorithms of Section IV-A were evaluated on an Airy pattern PSFs; the NI algorithm cannot be applied to the Airy pattern, and has been replaced by an importance sampling algorithm set to take 40401 samples per pixel. The simulated pixel pitch was fixed at 4.88 micron, and green light (550 nm) was chosen to compute the diffraction pattern. Different numerical apertures were investigated, since this controls the effective width of the Airy pattern relative to the pixel size.

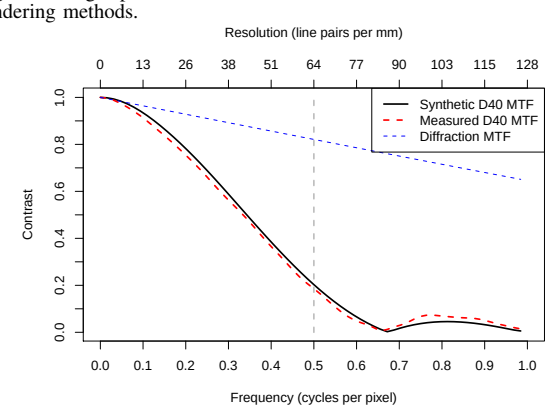


Fig. 15. Comparison of the MTF of a synthetic image with that of a knife-edge target imaged with a Nikon D40 camera.

From Table III is can be seen that the importance sampling algorithms once again have a decisive lead over the uniform sampling strategies (also visible in Figure 14). It does appear that the accuracy improves very slowly with an increase in the number of samples taken. One of the main reasons for this apparent slow increase is the large support of the Airy pattern. Since the importance sampling algorithms have been limited to a radius of 18 units (scaled according to f-number), a significant part of the tail of the Airy pattern is being truncated. This results in a lower limit on the RMSE values computed on the ESF, which cannot be reduced by increasing the number of samples while keeping this radius fixed.

C. Demonstration of system PSF accuracy

The accuracy of the combined PSF rendering strategy discussed in Section III-F is demonstrated in Figure 15. A Nikon D40 camera was used to image a knife-edge target, after which the slanted-edge algorithm was used to obtain the empirical MTF of the combined lens, OLPF and photosite aperture system. Focus bracketing was used to ensure that the system MTF is as accurate as possible, and a lens that is known to be diffraction limited was used. The MTF curve extracted

TABLE I
MEAN RMSE FOR GAUSSIAN PSFS WITH DIFFERENT STANDARD DEVIATIONS, OVER 25 DIFFERENT SUB-PIXEL SHIFTS.

Target	Mean RMS error ± standard deviation				
MTF50	11 × 11×UP	11 × 11×U	121×IS	2025×IS	NI
0.10	1.47e−02 ± 4.00e−05	1.24e−02 ± 1.68e−05	3.04e−03 ± 1.39e−05	4.05e−04 ± 2.01e−06	4.80e−06 ± 1.04e−07
0.15	1.94e−02 ± 7.91e−05	1.00e−02 ± 1.33e−05	2.44e−03 ± 1.44e−05	3.28e−04 ± 2.36e−06	3.88e−06 ± 7.87e−08
0.20	2.32e−02 ± 2.66e−05	8.62e−03 ± 2.01e−05	2.10e−03 ± 1.11e−05	2.80e−04 ± 2.86e−06	3.33e−06 ± 7.72e−08
0.25	2.70e−02 ± 6.81e−05	7.69e−03 ± 1.32e−05	1.87e−03 ± 1.21e−05	2.50e−04 ± 3.05e−06	2.94e−06 ± 8.54e−08
0.30	3.07e−02 ± 3.62e−05	7.00e−03 ± 1.72e−05	1.71e−03 ± 1.57e−05	2.28e−04 ± 2.96e−06	2.70e−06 ± 9.95e−08
0.35	3.47e−02 ± 3.12e−05	6.47e−03 ± 2.65e−05	1.58e−03 ± 1.42e−05	2.11e−04 ± 2.06e−06	2.48e−06 ± 8.16e−08
0.40	3.82e−02 ± 6.14e−05	6.02e−03 ± 1.14e−05	1.48e−03 ± 1.86e−05	1.97e−04 ± 2.77e−06	2.32e−06 ± 6.53e−08

TABLE II
MEAN MTF50 ACCURACY EVALUATION FOR GAUSSIAN PSFS WITH DIFFERENT STANDARD DEVIATIONS, OVER 25 DIFFERENT SUB-PIXEL SHIFTS.

Target	Mean error (%) ± standard deviation				
MTF50	11 × 11×UP	11 × 11×U	121×IS	2025×IS	NI
0.10	3.750 ± 0.016	5.206 ± 0.050	3.296 ± 0.036	3.343 ± 0.025	3.389 ± 0.008
0.15	1.513 ± 0.039	3.722 ± 0.059	1.563 ± 0.077	1.494 ± 0.028	1.521 ± 0.012
0.20	0.819 ± 0.060	3.039 ± 0.066	0.868 ± 0.086	0.787 ± 0.035	0.806 ± 0.014
0.25	0.743 ± 0.086	2.752 ± 0.046	0.555 ± 0.057	0.394 ± 0.030	0.414 ± 0.022
0.30	3.906 ± 0.125	2.420 ± 0.066	0.181 ± 0.177	0.119 ± 0.037	0.137 ± 0.036
0.35	24.573 ± 0.347	2.662 ± 0.080	0.066 ± 0.050	0.113 ± 0.051	0.103 ± 0.051
0.40	150.000 ± 0.000	2.063 ± 0.100	0.278 ± 0.143	0.327 ± 0.073	0.321 ± 0.075

TABLE III
MEAN RMSE FOR AIRY PATTERN PSFS AT DIFFERENT APERTURE VALUES, OVER 25 DIFFERENT SUB-PIXEL SHIFTS.

Relative	Mean RMS error ± standard deviation				
aperture	11 × 11×UP	11 × 11×U	121×IS	2025×IS	40401×IS
f/2.8	3.00e−02 ± 5.76e−05	1.00e−02 ± 4.53e−05	3.55e−03 ± 1.97e−05	1.80e−03 ± 1.43e−06	1.68e−03 ± 1.06e−06
f/5.6	3.85e−02 ± 5.61e−05	1.43e−02 ± 5.83e−05	5.02e−03 ± 1.13e−05	2.52e−03 ± 1.11e−06	2.34e−03 ± 6.48e−07
f/8	4.12e−02 ± 5.00e−05	1.71e−02 ± 5.83e−05	5.94e−03 ± 9.97e−06	2.86e−03 ± 6.12e−07	2.63e−03 ± 3.15e−07
f/16	2.86e−02 ± 2.37e−05	2.41e−02 ± 5.40e−05	7.75e−03 ± 6.24e−06	3.03e−03 ± 9.18e−07	2.84e−03 ± 9.17e−07
f/32	3.22e−02 ± 1.88e−05	3.37e−02 ± 3.63e−05	8.61e−03 ± 7.62e−06	3.15e−03 ± 3.49e−06	2.91e−03 ± 3.26e−06

from the synthetic image matches the empirical camera MTF reasonably well.

D. Rendering time

Due to space constraints, detailed rendering time results have been omitted, but brief results follow. All synthetic images were rendered as 446×446 pixel images, containing a single square target of 250×250 pixels in size. Rendering a Gaussian PSF (standard deviation of 0.625 pixels) and an Airy pattern PSF (f/8, $\lambda = 0.55\mu$m, pitch$= 4.88\mu$m) yields the following rendering times:

Gauss. Alg.:	11 × 11×U	121×IS	2025×IS	NI
Time (s) :	1.1	3.4	4.15	42.38
Airy Alg.:	11 × 11×U	121×IS	2025×IS	40401×IS
Time (s) :	0.8	0.819	3.65	67.7

Rendering times depend somewhat on the diameter of the PSF, with wider PSFs rendering more slowly, owing to an adaptive early convergence test. Including the effects of the photosite aperture is expensive: the 2025×IS rendering times increase to 29.5 s and 119 s for the single-photosite aperture and 4-dot OLPF simulations respectively.

V. CONCLUSIONS

This paper described a variety of rendering algorithms that may be applied to generate synthetic images with specific point spread functions. These algorithms have been demonstrated to be very accurate, while keeping the computational complexity relatively low. The results highlight that simple strategies (e.g., fixed-grid uniform sampling) produces much worse results than the importance sampling methods for the same number of samples.

The rendering methods introduced here can be used to generate reference synthetic images to the desired level of accuracy, and are available in the MTF Mapper project (http://sourceforge.net/projects/mtfmapper). These images can be used to calibrate other algorithms, e.g., the slanted-edge MTF estimation algorithm, or to evaluate super-resolution or shape-detection algorithms.

REFERENCES

[1] K. Kohm, "Modulation transfer function measurement method and results for the orbview-3 high resolution imaging satellite," in *Congress International Society for Photogrammetry and Remote Sensing*, vol. 20, 2004, pp. 12–23.

[2] S. Van der Walt, "Super-resolution imaging," Ph.D. dissertation, Stellenbosch: University of Stellenbosch, 2010.

[3] J. Ouellet and P. Hébert, "Precise ellipse estimation without contour point extraction," *Machine Vision and Applications*, vol. 21, no. 1, pp. 59–67, 2009.

[4] I. Sutherland and G. Hodgman, "Reentrant polygon clipping," *Communications of the ACM*, vol. 17, no. 1, pp. 32–42, 1974.

[5] G. Airy, "On the diffraction of an object-glass with circular aperture," *Transactions of the Cambridge Philosophical Society*, vol. 5, p. 283, 1835.

[6] R. Palum, "Optical antialiasing filters," in *Single-Sensor Imaging: Methods and Applications for Digital Cameras*, R. Lukac, Ed. Boca Raton, FL: CRC Press, Sept. 2008, pp. 105–136.

[7] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery, *Numerical recipes: The art of scientific computing*, 3rd ed. Cambridge university press, 2007.

[8] B. Moro, "The full Monte," *Risk*, vol. 8, no. 2, pp. 57–58, 1995.

Investigating Parameters for Unsupervised Clustering of Speech Segments using TIMIT

Lerato Lerato and Thomas Niesler
Department of Electrical and Electronic Engineering
University of Stellenbosch, South Africa
{llerato, trn}@sun.ac.za

***Abstract*—We investigate the application of agglomerative clustering to short segments of speech signals. The successful direct clustering of such sub-word speech segments has direct application in the automatic derivation of pronunciation variants for use in automatic speech recognition (ASR) systems. We consider several configurations of hierarchical agglomerative clustering in order to determine the best configuration for the speech clustering task. Similarity between segments is computed by dynamic time warping (DTW), within which the application of Euclidean and city-block distance measures were evaluated. The effect of path length normalisation of the DTW score is considered, and finally the application of three different between-cluster distance measures is compared. Experiments are carried out on a subset of the phone segments present in the TIMIT database. We find that the city-block distance in conjunction with a normalised DTW score and the Ward cluster linkage method lead to best results.**

I. Introduction

The objective of this paper is to investigate the parameters that influence the unsupervised clustering of short segments of speech data. Clustering spans many fields of pattern recognition, such as image processing, speech processing and document recognition. We focus on a speech processing application in which short segments of audio taken from a corpus of connected speech must automatically be grouped into different clusters in an effort to group similar sounds. In order to allow controlled experimentation and the evaluation of clustering results, the segments we consider are phone units taken from the TIMIT speech corpus.

The unsupervised clustering of sub-word speech sounds has several applications in speech processing. One of these is the automatic generation of pronunciations for use in an automatic speech recognition (ASR) system. This application was considered in [1], where the authors bootstrap a system using grapheme-based subword models. Later work in which this restriction was removed indicated that careful attention to the clustering of audio segments would be required [2]. In this paper, we address this issue. Other applications of the type of clustering that we consider include automatic keyword discovery [3] in which frequently recurring words or phrases are detected in untranscribed audio. Mareuil *et al* [4] and Imperl *et al* [5] clustered speech segments from multiple languages for applications in multilingual speech recognition and language identification respectively. Mak and Barnard [6] cluster speech segments using agglomerative hierarchical clustering (AHC) in an approach that is similar to ours. They however use Gaussian probability density functions and the Bhattacharyya distance to find the inter-cluster similarity. In contrast to the work covered in [2], we focus exclusively on the clustering problem and experiment with several configurations in order to determine how the parameters affect the performance of the algorithm. Neel [7] performs cluster analysis in various ways on TIMIT speech data. In this work however the number of clusters was fixed. We attempt unsupervised clustering in which the data are clustered purely on the basis of the feature representation.

II. Agglomerative hierarchical clustering

Cluster analysis is the process of discovering the natural groupings of a set of patterns, points or objects [8], [9], [10], [11]. The analysis itself is based on the hypothesis that similarity between related points in the data set should be high while similarity between different points should be low. The points are then grouped according to this similarity. Agglomerative hierarchical clustering (AHC) is one approach to performing the grouping task.

AHC is a bottom-up method that merges pairs of clusters according to a certain similarity measure. Initially each data point (speech segment) forms a singleton cluster. At this stage the number of clusters is equal to the number of speech segments. Subsequently clusters are merged in a pairwise fashion until a single cluster remains. This procedure generates a tree-like hierarchical grouping known as a dendrogram, as illustrated in Figure 1.

In order to determine the similarity between two clusters, the similarity between individual members of the clusters must also be computed. These members are in our case segments of speech, and their similarity will be computed using dynamic time warping (DTW), which will be described in Section II-A. Furthermore, once the similarity between individual cluster members is known, the similarity between the clusters themselves can be computed in a variety of ways. Some of these linkage methods will be described in Section II-B.

A. Dynamic time warping

Dynamic time warping (DTW) is an algorithm that calculates the similarity between two sequences of generally unequal length. DTW was once the basis of template-based isolated-word speech recognition, but has been superseded by statistical techniques such as hidden Markov models (HMMs) [12], [13]. For our application, in which we would like

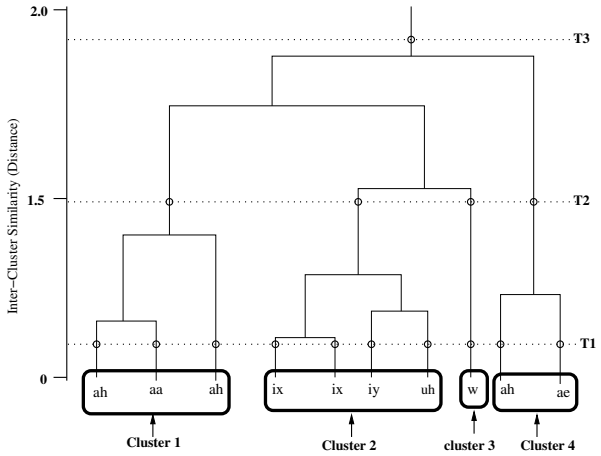


Figure 1. An example of agglomerative hierarchical clustering (AHC) and the associated dendrogram.

to assess the similarity between two specific but otherwise arbitrary segments of speech, it is well-suited.

Let the two speech segments in question be $\mathbf{S_1}(\alpha)$ and $\mathbf{S_2}(\omega)$, where $\alpha = 1, 2, \ldots, A$ and $\omega = 1, 2, \ldots, \Omega$. $\mathbf{S_1} = \{X_1, X_2 \ldots, X_A\}$, $\mathbf{S_2} = \{Y_1, Y_2 \ldots, Y_\Omega\}$ and X_α or Y_ω are the *T*-dimensional feature vectors. Now consider a $\Omega \times A$ local distance matrix, $\mathbf{D}$, whose entries are the distances between all possible pairs of feature vectors from the two segments. The distance measures can be chosen to suit the application, and we will consider the Euclidean and the city-block distances in our evaluation. The Euclidean distance is given by:

$$d(X_\alpha, Y_\omega) = \sqrt{\sum_{i=1}^{T}(x_{\alpha i} - y_{\omega i})^2} \tag{1}$$

while the city-block entry is expressed mathematically as:

$$d(X_\alpha, Y_\omega) = \sum_{i=1}^{T} |x_{\alpha i} - y_{\omega i}| \tag{2}$$

From the matrix $\mathbf{D}$, the best alignment between the sequences $\mathbf{S_1}(\alpha)$ and $\mathbf{S_2}(\omega)$ can be computed recursively by the principle of dynamic programming [13]. The score associated with this best alignment can then be taken as a measure of similarity between the two sequences. By dividing this score by the total length of the alignment path, a measure of the average per-frame similarity can be obtained. Both normalised and unnormalised versions of the DTW score will be considered in our experimental evaluation.

B. Linkage methods

Dynamic time warping allows the similarity between two individual speech segments to be evaluated. However, during clustering, the similarity between two clusters of segments must also be computed. There are several strategies to compute this inter-cluster similarity, and we have chosen three of these *linkage methods* for experimental evaluation: average-link, complete-link and Ward-link [10], [11]. We chose these linkage methods because of their popularity and also that the shapes of data points used has not been analysed in detail. We will use the following notation for the description of the linkage methods:

- U and V are two clusters whose similarity must be measured.
- K and L are the number of elements in U and V respectively.
- When cluster U contains segment c_i, we denote this by $c_i \in U$.
- $d(c_i, c_j)$ is the distance between two segments, as calculated by DTW.

The **average-link** uses the average distance computed between all possible pairs of observations drawn from U and V. The criterion joins clusters with small variances and is less influenced by outliers than many other methods. It can mathematically be presented as:

$$Sim_{ave}(U, V) = \frac{1}{K \times L} \sum_{c_i \in U} \sum_{c_j \in V} d(c_i, c_j) \tag{3}$$

The **complete-link** criterion considers the points in each cluster that are furthest apart. This can make it vulnerable to outliers as such anomalous points will often be the most distant. However it has the advantage of preferring compact clusters. It is calculated as:

$$Sim_{comp}(U, V) = \max_{c_i \in U, c_j \in V} d(c_i, c_j) \tag{4}$$

The **Ward-link** method considers the increase in the total intra-cluster sum-of-squares that results when two clusters are merged. This intra-cluster sum is defined as the sum of squares of the distances between all members of the cluster and its centroid. This method tends to join clusters with a small number of observations. It is mathematically presented as:

$$Sim_{ward} = \frac{\|\overline{c}_U - \overline{c}_V\|^2}{(1/K + 1/L)} \tag{5}$$

where $\|\overline{c}_U - \overline{c}_V\|^2$ is the distance between the centroids, $\overline{c}_U$ and $\overline{c}_V$, of clusters U and V respectively.

III. Cluster evaluation

In general, the clustering process will make errors, for example by placing two dissimilar segments in the same cluster, or by assigning similar segments to different clusters. Ideally, however, each cluster contains segments from only one phone, and all the segments of a particular phone belong to the same cluster. In order to to evaluate the success of the clustering process, we require measures that will indicate the extent to which these competing goals are achieved. Several methods have been proposed to accomplish this [14] and of these we have chosen two for our experimental evaluation. Let us consider our data to consist of N segments that belong to J different *classes*. Ideally[1] the number of clusters K, also referred to as the *cardinality*, should equal the number of classes. Now assume the following notation:

[1]This has the disadvantage of considering alternative groupings of acoustically similar clusters as errorful. We leave the analysis of this effect to future work, in which ASR evaluations are incorporated.

- $\mathbf{G} = \{G_1, G_2, ..., G_K\}$ where $\mathbf{G}$ is the set of clusters and G_k is cluster k that contains speech segments.
- $\mathbf{C} = \{C_1, C_2, ..., C_J\}$ where $\mathbf{C}$ is the set of classes and C_j is a set of phones with the same class.
- $\max_j |G_k \cap C_j|$ represents the number of occurrences of the most frequent phone in cluster G_k.

A. Normalised mutual information

The normalised mutual information (NMI) is based on the mutual information between classes and clusters [9],[14]. The mutual information is denoted by $I(\mathbf{G}, \mathbf{C})$ and is given by:

$$I(\mathbf{G}, \mathbf{C}) = \sum_{G_k \in \mathbf{G}} \sum_{C_j \in \mathbf{C}} P(G_k \cap C_j) \log \frac{P(G_k \cap C_j)}{P(G_k)P(C_j)} \quad (6)$$

where $P(G_k)$, $P(C_j)$ and $P(G_k \cap C_j)$ are the probabilities of a speech segment occurring in cluster G_k, in class C_j and in both cluster G_k and class C_j, respectively.

The mutual information measure, $I(\mathbf{G}, \mathbf{C})$, does not penalise cardinalities. To make it sensitive to the varying number of clusters, it can be normalised by a factor based on the entropy of both clusters and classes. This normalising factor is given by: $1/2[H(\mathbf{G}) + H(\mathbf{C})]$, where $H(.)$ denotes the entropy. $H(\mathbf{G})$ measures cluster cohesiveness [15] and is given by:

$$H(\mathbf{G}) = -\sum_{G_k \in \mathbf{G}} P(G_k) \log P(G_k) \quad (7)$$

$H(\mathbf{C})$ is a measure of class cohesiveness and is calculated similarly. Normalising $I(\mathbf{G}, \mathbf{C})$ in this way makes it respond to cardinality, because entropy tends to increase with the number of clusters. The normalised mutual information criterion is therefore given by:

$$NMI(\mathbf{G}, \mathbf{C}) = \frac{I(\mathbf{G}, \mathbf{C})}{1/2[H(\mathbf{G}) + H(\mathbf{C})]} \quad (8)$$

The NMI is always a number between 0 and 1 where 1 denotes purely clustered data.

B. The F-measure

The F-measure is based on recall and precision for each cluster with respect to each class in the data set [10], [16]. The precision and recall quantities are based on: (1) a true positive decision (TP) where two similar segments are assigned to the same cluster, (2) a true negative decision (TN) in which two dissimilar segments are assigned to two different clusters. The sum of TP and TN are known as the correct decisions. In addition, a false positive (FP) error occurs when two dissimilar segments are assigned to the same cluster. A false negative (FN) error, on the other hand, emerges when two similar segments are placed into different clusters. Precision, Prc, is given by:

$$Prc = \frac{TP}{TP + FP} \quad (9)$$

where $TP + FP = \sum_i^K \binom{|G_i|}{2}$, $TP = \sum_i^K \binom{Q_i}{2} + 1$ and $Q_i = \max_i |G_i \cap C_j|$. Equation 9 is the ratio of segments from the same class in the particular cluster to the total number of segments in that cluster.

Recall, Rec, is given by:

$$Rec = \frac{TP}{TP + FN} \quad (10)$$

where $FN + TN = \binom{N}{2} - (TP + FP)$, $FN = \sum_i^J \binom{V_i}{2} - TP$, $V_i = |C_i \cap G_j|$ and $|C_i \cap G_j|$ is the number of segments of the same phone in cluster j.

The *recall* expression is the ratio of segments from the same class in the particular cluster to the total number of all segments that belong to the same class in all clusters. The F-measure is quantified as:

$$F = \frac{2 \times Prc \times Rec}{Prc + Rec} \quad (11)$$

which can be further refined by introducing a mechanism that allows more weight to be assigned to recall or to precision. Let β be a constant and rewrite the F-measure as:

$$F_\beta = \frac{(\beta^2 + 1) \times Prc \times Rec}{\beta^2 \times Prc + Rec} \quad (12)$$

We select $\beta > 1$ to give more weight to recall . When $\beta = 1$ Equation 12 simplifies to Equation 11.

IV. Data preparation

Our experimental evaluation is based on speech data taken from the TIMIT speech corpus [17]. The speech is parameterised as a series of feature vectors composed of Mel frequency cepstral coefficient (MFCCs). MFCC's are chosen on the basis of their robustness and frequent usage in well performing speech processing systems. In particular, cepstral mean normalisation can be applied to minimise speaker and channel effects.

Due to the large number of inter-segment similarities that must be calculated during clustering, we have based our experiments on a subset of the TIMIT data. A total of 100 speakers were chosen from the seven dialects present in the corpus. Speaker selection was random, but an even distribution across the dialect regions and an equal male/female balance within each region were ensured. For these 100 speakers, the five phonetically compact SX sentences were considered, bringing the total number of utterances in our dataset set to 500.

From each utterance, all occurrences of the phones listed in Table I were extracted for experimentation. The table shows that two sets of data were chosen for experimentation: a short set (*set 1*) and a long set (*set 2*), and that the short set is a subset of the long set. The reason for including two sets of data was to allow contrastive experimentation when investigating the effect of path length normalisation on clustering performance. In particular, the short set was chosen in initial experiments but was found to be rather homogeneous, consisting exclusively of vowels and of segments with fairly similar lengths. The long set, on the other hand, is more diverse since it includes semivowels, and a greater variety of segment lengths, as illustrated in Figure 2.

Phone set	Segments
Set 1 (short set)	*aa, ae, ah, eh, ih, iy, uh*
Set 2 (long set)	*aa, ae, ah, eh, ih, iy, uh, er, ey, ix, aw, axr, l, oy, r, y*

Table I
TIMIT DATA USED FOR EXPERIMENTATION.

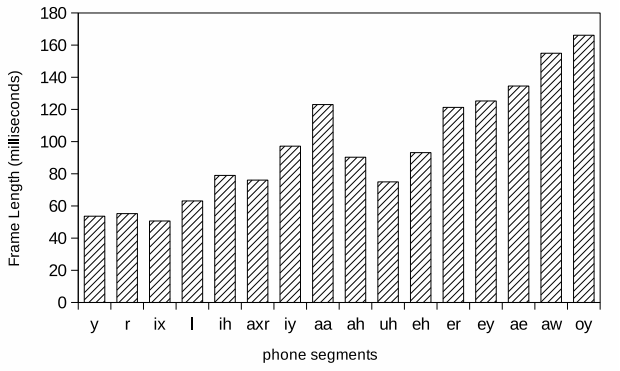


Figure 2. Average durations of the segments in sets 1 and 2.

V. EXPERIMENTAL SETUP

The dynamic time warping distance measures that were considered (Euclidean and city-block) as described in Section II-A were implemented in C++. The linkage methods and hierarchical clustering process detailed in Section II were implemented using the Octave statistical toolbox. Various configurations of the clustering process were applied to the datasets described in Table I with the specific aim of answering the following questions:

1) Does the Euclidean or the city-block distance measure yield better clustering when implemented within the DTW similarity measure?
2) Should the DTW distance be normalised with the path length or not?
3) Which linkage method gives best clustering results: average, complete or Ward?

In each set of experiments, the clustering threshold was varied in order to establish the effect of the number of clusters on performance.

VI. EXPERIMENTAL RESULTS

A. City-block vs Euclidean distance in DTW

First we investigate the effect on clustering performance of varying the method used to compute the the distance between individual feature vectors as part of the DTW alignment. The NMI and the F-measure cluster evaluation metrics are used to assess the quality of every set of clustering results. Figure 3 shows these results for the smaller dataset (set 1). The Ward linkage method was employed throughout as this was found to lead to better results than the other linkage methods, as will be demonstrated later.

Figure 3 shows that optimal performance is achieved for between approximately 15 and 40 clusters, and that the city-block distance generally outperforms the Euclidean distance in this range.

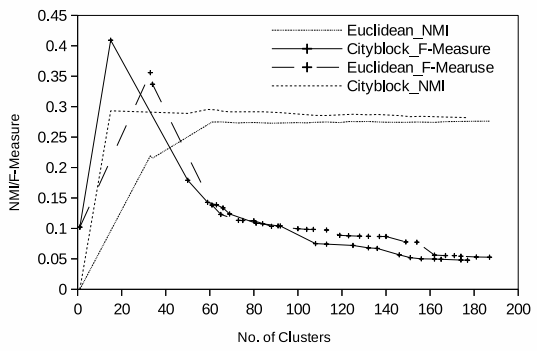


Figure 3. Clustering performance in terms of NMI and F-measure for the city-block and Euclidean DTW distances for data set 1.

B. Normalised path length in DTW

We have already observed in Figure 2 that the phone segments vary in length. The DTW procedure results in the best alignment between two speech segments of arbitrary length. Since the DTW score increases monotonically along the alignment path, it is in principle possible that the alignment of a long and a much shorter segment lead to a better score than the alignment of two longer segments, even when the former are acoustically dissimilar and the latter similar. In order to account for this, the alignment score can be normalised by its length, leading to a per-frame rather than an overall score. Figure 4 shows the effect of this normalisation on the NMI and the F-measure for city-block-based DTW on the smaller dataset (set 1), while Figure 5 shows the same experiment for the larger dataset (set 2).

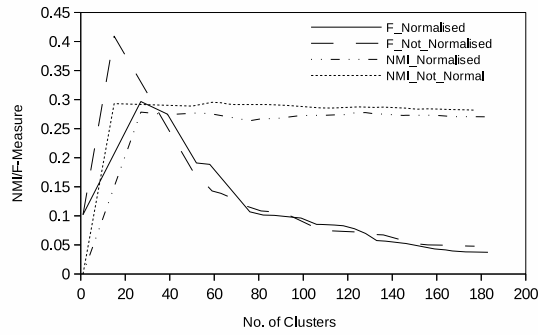


Figure 4. Comparison of normalised and unnormalised city-block based DTW for data set 1.

For the smaller dataset (set 1), path normalisation leads to deteriorated performance, while the reverse is true for the larger dataset (set 2). We ascribe this difference to the relative homogeneity of set 1. Since the phone lengths and also the sounds are fairly similar in this set (all vowels), the scenario in which a very short and a very long segment that are acoustically quite different lead to a better overall alignment score is rare. Since the length of the segment itself includes discriminatory value, its effect on the alignment scores can be beneficial, and its removal by normalisation detrimental. However, when the lengths of the segments, as well as the sounds themselves, are more diverse (set 2 contains both

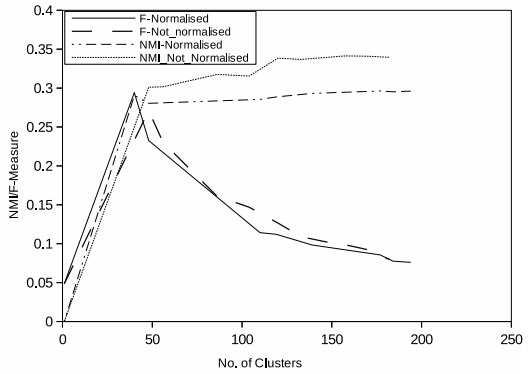


Figure 5. Comparison of normalised and unnormalised city-block based DTW for data set 2.

vowels and semivowels) the benefits of normalisation begin to dominate.

C. Evaluation of the linkage methods

Using the city block distance in conjunction with the unnormalised DTW score for the shorter dataset (set 1), as well as the city block distance in conjunction with the normalised DTW score for the longer dataset (set 2), the effect of varying the linkage method used to determine inter-cluster similarity could be studied. The performance for the respective cases in terms of the F-measure are shown in Figures 6 and 7.

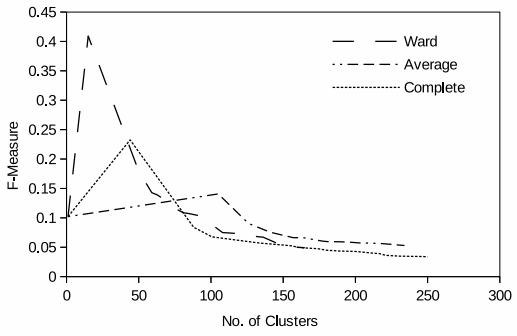


Figure 6. Evaluation of linkage methods for data set 1.

We observe that, for the smaller dataset (set 1), use of the Ward linkage method leads to best performance when the number of clusters is small. The peaks in performance for the average-link and complete-link methods occur when the number of clusters is larger, and are lower. For the longer dataset (set 2), a similar picture emerges.

D. Number of clusters

From Table I it is evident that the 'true' number of clusters in the data is 7 and 16 for set 1 and set 2 respectively. However, the peaks in Figures 6 and 7 correspond to approximately 15 and 40 clusters respectively. It appears therefore that the overall quality of the clusters is better when it is allowed to exceed the 'true' number of clusters by a factor of between 2 and 3.

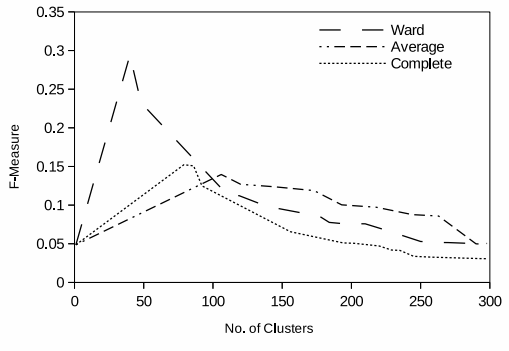


Figure 7. Evaluation of linkage methods for data set 2.

VII. Discussion and conclusions

We have presented a comparative evaluation of several configurations of agglomerative hierarchical clustering applied to the grouping of subword speech sounds. Due to the high computational cost of the experiments, a subset of the TIMIT data was used. Our experiments showed that the best clusters were obtained when calculating the DTW score using the city block distance and normalising it with respect to the alignment path length. Furthermore, the Ward inter-cluster distance let to better clusters than the average and complete linkage methods.

Although the number of clusters leading to best performance was found to exceed the actual number of classes in the data by a factor of between 2 and 3, this may be due to contextual effects. As experience in automatic speech recognition has shown, co-articulation may cause the same phone to differ acoustically from other instances due to differing left and/or right contexts. Similar variability may be introduced by differences in speaker dialect or gender. These factors could also limit the achievable accuracy of the clustering process itself. In future work, this aspect will be more carefully investigated.

The appreciable differences in the results obtained for the smaller and the larger datasets also indicate that experiments on the full set of phones are requited in order to obtain definitive answers to our research questions. Hence the optimisation and parallelisation of the clustering algorithms will also form part of our ongoing work.

References

[1] B. R. R. Singh and R. Stern, "Automatic generation of subword units for speech recognition systems," *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 2, pp. 89–99, 2002.

[2] G. Goussard and T. Niesler, "Automatic discovery of subword units and pronunciations for automatic speech recognition using TIMIT," in *Proc. PRASA*, (Stellenbosch, South Africa), 2010.

[3] A. Park and J. Glass, "Towards unsupervised pattern discovery in speech," in *Proc. ASRU*, (San Juan, Puerto Rico), 2005.

[4] C. C.-A. P.B. de Mareuil and M. Adda-Decker, "Multi-lingual automatic phoneme clustering," in *Proc. ICPhS*, (San Francisco), 1999.

[5] B. H. B. Imperl, Z. Kacic and A. Zgank, "Clustering of triphones using phoneme similarity estimation for the definition of a multilingual set of triphones," *Speech Communication*, vol. 39, no. 4, pp. 353–366, 2003.

[6] B. Mak and E. Barnard, "Phone clustering using the Bhattacharyya distance," in *Proc. of ICSLP*, (Philadelphia, USA), 1996.

[7] J. Neel, "Cluster analysis methods for speech recognition," Master's thesis, KTH Royal Institute of Technology, Stockholm, 2002.

[8] C. Fraley and A. E. Raftery, "How many clusters? Which clustering method? Answers via model-based cluster analysis," *The Computer Journal*, vol. 41, pp. 578–588, 1998.

[9] J. A. E. Amigo, J. Gonzalo and F. Verdejo, "A comparison of extrinsic clustering evaluation metrics based on formal constraints," *Information Retrieval*, vol. 12, no. 4, pp. 461–486, 2009.
[10] C. D. Manning and P. Raghavan, *Introduction to Information Retrieval*. Cambridge University Press, 2008.
[11] A. Jain, "Data clustering: 50 years beyond K-means," *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651–666, 2010.
[12] L. R. C. Myers and A. Rosenberg, "Performance tradeoffs in dynamic time warping algorithms for isolated word recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 28, no. 6, pp. 623–635, 1980.
[13] F. J. Owens, *Signal Processing of Speech*. Macmillan Press, 1993.
[14] J. E. N. X.Vinh and J. Bailey, "Information theoretic measures for clusterings comparison: Variants, properties, normalisation and correction for chance," *Journal of Machine Learning Research*, vol. 11, pp. 2837–2854, 2010.
[15] M. Sileshi and B. Gamback, "Evaluating clustering algorithms: Cluster quality and feature selection in content-based image clustering," in *WRI World Congress on Computer Science and Information Engineering*, (New York, USA), 2009.
[16] B. Larsen and C. Aone, "Fast and effective text mining using linear-time document clustering," in *Proc. ACM SIGKDD*, (New York, USA), 1999.
[17] A. K.Halberstadt and J. Glass, "Heterogeneous acoustic measurements for phonetic classification," in *Proc. Eurospeech*, (Rhodes, Greece), 1997.

Performance Evaluation of Spot Detection Algorithms in Fluorescence Microscopy Images

Matsilele Mabaso*, Daniel Withey‡, Natasha Govender§ and Bhekisipho Twala†
* ‡ §MDS(MIAS)
Council for Scientific and Industrial Research
Pretoria, South Africa,
Email: *MMabaso@csir.co.za; ‡DWithey@csir.co.za; §NGovender@csir.co.za
†Department of Electrical and Electronic Engineering
University of Johannesburg
Auckland Park, South Africa
Email: BTwala@uj.ac.za

***Abstract*—Detection of messenger Ribonucleic Acid (mRNA) spots in fluorescence microscopy images is of great importance for biologists to better understand cell function. Fluorescence microscopy and specific staining methods make biological molecules appear as bright spots in image data. Manual analysis of such data is both time consuming and laborious and can lead to errors. In this study we compare several computer-based methods for detection of spots in fluorescence microscopy images. The algorithms under comparison are, Isotropic Undecimated Wavelet Transform, Feature Point Detection, H-Dome transformation and Laplacian of Gaussian. The performance of the algorithms is validated using synthetic and real image data. The synthetic images were corrupted by Gaussian noise of different levels and the real images were obtained using fluorescence microscopy. Algorithm performance is compared based on detection accuracy.**

I. Introduction

In recent years, advances in molecular and cell biology have triggered the development of a highly sophisticated imaging tool known as fluorescence microscopy [1]. Fluorescence microscopy is used to visualize and study intracellular processes. This is accomplished using a specific staining method to make the biological molecules appear as bright particles (spots) when viewed through a microscope, as shown in Figure 1. These bright particles are local intensity maxima whose intensity level is significantly different from their neighbourhood.

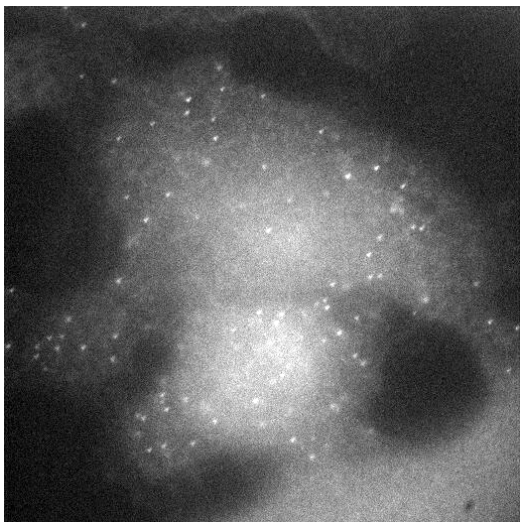


Fig. 1. Sample image obtained using fluorescence microscopy.

Spot detection is a fundamental step for biologists to better understand intracellular processes. The goal of spot detection is to obtain information about the location and properties of the features (spots). However, quantitative analysis of these spots is often still reliant on manual evaluation which is a tedious process involving many hours of human inspection, and is impractical for use on large data sets. Therefore it is useful to use computer based-algorithms to automate this process. Hence, there is a great demand for the automation of spot detection methods and it is attracting increased research attention.

Over the past years, a number of computer based detection methods have been proposed to address the task of manual analysis. These methods are based on gray-scale opening of top-hat filter [2] and adaptive thresholding [3]. These methods do not give satisfactory results with biological images for two main reasons: first, biological images have low signal to noise ratios (SNRs) and second, the image may present an uneven background [4]. Recent methods are based on wavelet transform techniques such as isotropic undecimated transform [4], improved wavelet transform [5] and multiscale variance stabilizing transform [6].

In this work we compare the performance of several detection methods used to detect bright particles in fluorescence microscopy images, using both synthetic images and real images. The methods under comparison are Isotropic Undecimated Wavelet Transform (IUWT) [4], Feature Point Detection (FPD) [7], H-Dome transformation [8] and Laplacian of Gaussian (LoG) [9]. The first three methods have performed well in previous studies [10, 11], however

no comparison was done with LoG, though it has been used successfully in microscope image analysis [9]. Our comparison takes into account various conditions of spots eg. different intensity and radius of spots as well as noise and non-uniform background intensity.

The layout of the paper is as follows: Section II discusses the related work and section III describes the various algorithms used in the experiments. Section IV presents the performance measures and in section V experiments are discussed. Section VI discusses the experimental results, and finally, the conclusions are given in section VII.

II. Related Work

A lot of research effort on spot detection methods has been performed during the last few years. Recent work can be found in [10, 11]. This can be divided into two groups of detection methods, supervised and unsupervised detection methods. Supervised methods require learning as the first step while unsupervised methods do not require learning. In our study we considered four unsupervised detection methods.

Smal et al. (2010) [10] recently performed a quantitative comparison of various spot detection methods used in fluorescence microscopy imaging. The methods under comparison consisted of seven unsupervised and two supervised methods. The experiments were conducted on synthetic data of three different types, for which the ground truth was available. The experiments were also conducted on real image data obtained from two different biological studies. The results suggested that at a very high noise the supervised methods perform best overall. A similar study was conducted in [11] consisting of eleven spot detection algorithms from various application fields and tests were performed using both synthetic and real images. Their studies found major differences in the performance of different algorithms, in terms of both object counts and segmentation accuracies.

Olivo-Marin (2002) [4] introduced a method for detecting spots in 2D fluorescence microscopy images which was further modified to deal with 3D images in [12]. The method is based on the multiscale product of subband images resulting from the a trous wavelet transform [13] of the original image and can extract information such as the number and position of spots in an image. The algorithm showed good detection; however, failed to detect spots when SNR was low and when spots were far from the focal point.

In addition, [14] introduced a technique for detecting spots in fluorescence microscopy images. The method is based on Top-hat transformation by Rotational Morphological Processing (RMP) and a structuring element (SE). The method was reported to perform better than the H-dome transformation and top-hat filter.

III. Spot Detection Methods

The following are detection methods considered in our study.

A. Feature Point Detection (FPD)

The method of feature point detection was proposed in [15] and used for the detection and tracking of particles in cell images in [7].
The algorithm consists of four steps:

1) Image restoration: this step corrects imperfections in the image using a box-car average estimation and simultaneously enhances spot-like structures by convolving with a Gaussian kernel. The convolution kernel is given by:

$$K^w = \frac{1}{K_0^w}\left[\frac{1}{B}exp(-\frac{i^2+j^2}{4\lambda_n^2}) - \frac{1}{(2w+1)^2}\right], \quad (1)$$

where K_0^w and B are normalization factors, λ_n defines the kernel width and w is a user-tunable constant, thus the final image after restoration is given by:

$$I_f(x,y) = \sum_{i=-w}^{w}\sum_{j=-w}^{w} I(x-i,y-j)K^w(i,j), \quad (2)$$

where(x,y) and (i,j) are pixel coordinates in the image and kernel, respectively.
2) Estimating the particle location: this is done by locating local intensity maxima in the filtered image, $I_f(x,y)$. A local maximum is considered to be a spot if it has the highest intensity within a local window and the intensity is in the r^{th} highest percentile.
3) Refining the particle location: this step reduces the standard deviation of the position measurement. It is based on the assumption that the local intensity maximum of point P at $(\hat{x}_p, \hat{y}_p)$ is near the geometric center (x_p, y_p) of the spot. The offset is approximated by the distance to the gray-level cetroid in the filtered image, $I_f(x,y)$:

$$\begin{bmatrix} \varepsilon_x(p) \\ \varepsilon_y(p) \end{bmatrix} = \frac{1}{m_0(p)} \sum_{i^2+j^2\le w^2} \begin{bmatrix} i \\ j \end{bmatrix} I_f(\hat{x}_p+i, \hat{y}_p+j).$$

Factor $m_0(p)$, is the sum of all pixel values over feature point P given as:

$$m_o(p) = \sum_{i^2+j^2\le w^2} I_f(\hat{x}_p+i, \hat{y}_p+j). \quad (3)$$

Then the refined location estimate is determined as:

$$(\tilde{x}_p, \tilde{y}_p) = (\hat{x}_p + \varepsilon_x(p), \hat{y}_p + \varepsilon_y(p)). \quad (4)$$

4) Non-particle discrimination: this step rejects false identifications from sources such as auto fluorescence and dust. This step is based on the intensity moments of order 0 and 2, and identifies true particles as those within a cluster in the m_0, m_2 plane. A detailed description of the discrimination step can be found in [7].

B. H-Dome Transformation

The method of H-dome transformation was proposed in [8]. The method is based on the mathematical morphology:

$$Hdome(I(x,y)) = I(x,y) - \rho_I(I(x,y) - h), \quad (5)$$

where $(I(x,y)-h)$ denotes the result of subtracting a constant, h, from a gray-scale image $I(x,y)$, and $\rho_I(I(x,y)-h)$ is the morphological reconstruction of the gray-scale image, $I(x,y)$ from $(I(x,y)-h)$. The gray-level reconstruction is obtained by geodesic dilation of $(I(x,y)-h)$ under $I(x,y)$. The H-Dome transform enhances local intensity maxima. In our experiments we used the Matlab function, imhmax as the implementation of the H-Dome method.

C. Isotropic Undecimated Wavelet Transform (IUWT)

The method of IUWT was proposed in [4] for the detection of of spots in biological images. The algorithm is based on the assumption that spots will be present at each scale of wavelet decomposition and thus will appear in the multiscale product. The algorithm starts by convolving the image $I(x,y)$ row by row and column by column with a symmetric low pass filter $h = [1,4,6,4,1]/16$, resulting in a smoothed image $I_i(x,y)$. This process is repeated for J scale levels, augmenting the filter with $2^{i-1}-1$ zeros between taps in each case. The corresponding wavelet coefficients, $W_i(x,y)$, are given as:

$$W_i(x,y) = I_{i-1}(x,y) - I_i(x,y), 0 < i \leq J. \quad (6)$$

Then, a hard thresholding is applied to reduce the effect of noisy wavelet coefficients with $t_i = k\sigma_i$, where σ_i is the standard deviation of the noisy wavelet coefficients at scale i and $k = 3$.

$$t_{hard}(W_i, t_i) = \begin{cases} W_i(x,y), & W_i(x,y) \geq t_i. \quad (7) \\ 0, & W_i(x,y) < t_i \quad (8) \end{cases}$$

Thus, after hard thresholding, a multiscale product of each wavelet coefficient is computed to get a correlation image, $P_J(x,y)$,

$$P_J(x,y) = \prod_{i=1}^{J} W_i(x,y). \quad (9)$$

All the values in the correlation image are compared to predetermined detection level, l_d, to discriminate between particle and background, and get a binary image of particles. A spot is accepted only at positions where the correlation is above l_d,

$$P_J(x,y) = \begin{cases} 255, & |P_J(x,y)| \geq l_d. \quad (10) \\ 0, & Otherwise \quad (11) \end{cases}$$

D. Laplacian of Gaussian

The method of Laplacian of Gaussian (LoG) was proposed in [9] for the detection and counting of mRNA spots. This methods counts the number of bright particles (spots) in images. The algorithm is based on the second order partial derivative of the Gaussian kernel:

$$G_\sigma(x,y) = \frac{1}{2\pi\sigma^2} exp\Big[- \frac{x^2+y^2}{2\sigma^2} \Big], \quad (12)$$

where (x,y) are pixel locations.

$$LoG = \frac{\partial^2}{\partial x^2} G_\sigma(x,y) + \frac{\partial^2}{\partial y^2} G_\sigma(x,y). \quad (13)$$

This algorithm reduces pixel noise, and enhances spots in image, $I(x,y)$, by convolving with a LoG filter. Then the method proceeds to find spots using connected components with a user-selected threshold.

IV. Performance measure

In order to test the performance of the four methods, we use two common measures: True Positive Ratio (TPR) and False Positive Ratio (FPR), as used in [10],

$$TPR = \frac{N_{TP}}{N_{TP} + N_{FN}}, \quad (14)$$

$$FPR = \frac{N_{FP}}{N_{FP} + N_{TN}}. \quad (15)$$

Because the number of true negatives (TN) is not known, the modified FPR is given as,

$$FPR^* = \frac{N_{FP}}{N_{TP} + N_{FN}}, \quad (16)$$

where N_{TP} is the number of true positives, N_{FN} is the the number of false negatives and N_{FP} is the number of false positives. Best performance is indicated when TPR is high and FPR^* is low.

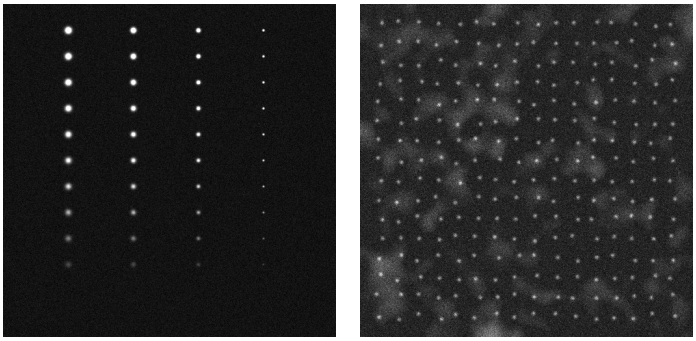
V. Experiments

A. Experiments with synthetic data

We have created two types of synthetic images, Type A and B with a known number of spots, as shown in Figure 2.

Type A images are of size 512×512 pixels containing 10×4 $2D$ Gaussian spots with decreasing intensity across the rows and decreasing radius across the columns, as shown in Figure 2(a). Gaussian noise (σ ranging from 6 to 40) was added to each image resulting in a set of noisy images.

Type B images were obtained using an ImageJ (NIH, USA) [16] plugin called Synthetic Data Generator [17]. Each image contains 256 Gaussian spots with SNR ranging from 1 to 5. These images show large background structures, leading to non-uniform background, as shown in Figure 2(b).



(a) (b)

Fig. 2. Examples of synthetic images used in the experiments. (a) Type A synthetic image, (b) Type B synthetic image

B. Experiments with real images

We also tested the performace of the four methods using real fluorescence microscopy images, as shown in Figure 1. Since the ground truth of these images was not available, we compared the detection results with manual inspection, as shown in Table III, and all the parameters of each method were kept to the same values as in the experiments with synthetic images.

VI. Experimental Results and Discussion

Tables I and II show the results of four detection methods using synthetic images. The results from Table I indicate that almost all methods perform best at σ = 6 except for FPD with lowest true positive ratio of 0.875. However as noise increases the to $\sigma = 40$, there are changes in performance. The results show that the IUWT has the highest average TPR, followed by LoG, FPD and H-Dome. However in terms of average FPR*, the IUWT has the highest value as compared to the other methods with LoG being the lowest. This shows that the IUWT is slightly more sensitive to noise than the other methods.

Noise immunity may be partly dependent on filter selection. The IUWT method applies a one-dimensional filter to each row and each column of the image whereas the FPD and LoG methods each use a two-dimensional, Gaussian filter. The additional filtering provided by the two-dimensional filter may be one reason for better noise immunity. The H-Dome method uses morphological processing and shows very good noise immunity except at the highest noise levels.

Results from Table II show that at high SNR $(= 5)$ almost all algorithms perform well with $TPR \approx 1$ except for the FPD which has the lowest TPR$(= 0.875)$. However as SNR decreases, there is a slight change in performance of the methods. The LoG method performed best with average TPR of 0.78 as compared to the other methods.

The results from real images, Table III, indicate that all methods perform well with TPR$\approx$ 1. Figure 3 shows the detected spots for each method.

TABLE I
RESULTS OF SPOT DETECTION METHODS USING TYPE A SYNTHETIC IMAGES

Dataset		IUWT	FPD	LoG	HDOME
1 (σ=6)	TPR	1	0.875	1	0.9
	FPR*	0	0	0	0
2 (σ=12)	TPR	1	0.875	0.975	0.9
	FPR*	0.05	0	0	0
3 (σ=18)	TPR	0.975	0.875	0.975	0.875
	FPR*	0.15	0	0.025	0.125
4 (σ=24)	TPR	0.95	0.9	0.9	0.825
	FPR*	0.4	0	0	0
5 (σ=34)	TPR	0.925	0.9	0.875	0.8
	FPR*	0.75	0.45	0	0.125
6 (σ=38)	TPR	0.9	0.9	0.9	0.775
	FPR*	0.65	0.75	0.075	0.625
7 (σ=40)	TPR	0.9	0.9	0.8	0.775
	FPR*	0.925	0.85	0.025	0.8
Average	TPR	0.95	0.89	0.918	0.835
	FPR*	0.42	0.29	0.018	0.24

TABLE II
RESULTS OF SPOT DETECTION METHODS USING TYPE B SYNTHETIC IMAGES

Dataset		IUWT	FPD	LoG	HDOME
1 (SNR=5)	TPR	1	0.875	1	1
	FPR*	0	0	0	0
2 (SNR=4)	TPR	1	0.84	1	0.97
	FPR*	0	0	0	0.0039
3 (SNR=3)	TPR	0.98	0.79	0.99	0.93
	FPR*	0.0039	0	0.012	0.066
4 (SNR=2)	TPR	0.44	0.75	0.81	0.84
	FPR*	0	0.066	0.059	0.39
5 (SNR=1)	TPR	0.0313	0.23	0.082	0.058
	FPR*	0.0078	0.21	0	0.32
Average	TPR	0.69	0.69	0.78	0.76
	FPR*	0.00234	0.0552	0.0142	0.156

TABLE III
RESULTS FOR SPOT DETECTION METHODS USING REAL FLUORESCENCE IMAGES

	Manual	IUWT	FPD	LoG	HDOME
TPR	1	0.98	0.95	0.98	0.97
FPR*	0	0.057	0.17	0.07	0.09

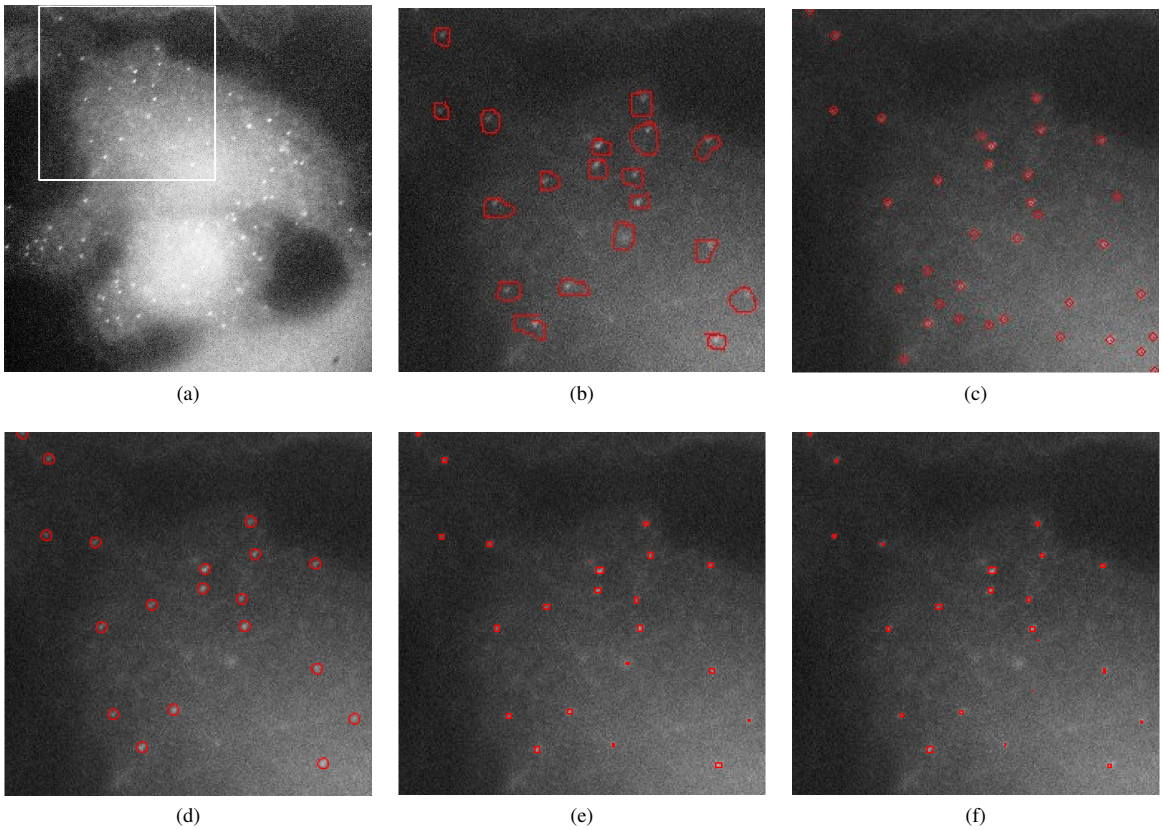


Fig. 3. Results of applying the proposed methods to a real fluorescence image. Detected spots are showed in red circles.(a) Original fluorescence image - with the box showing the zoom region. (b) Manual detection. (c) Detected spots using using FPD. (d) Detected spots with IUWT. (e) Detected spots with LoG. (f) Detected spots using H-Dome

The LoG method applies a user-selected threshold whereas the other methods are fully automatic. This is an advantage for the LoG method in cases where a threshold for accurate spot extraction does, in fact, exist. The LoG method has been used successfully for spot detection in fluorescence microscopy images [9] but it may not perform as well in images with greater background nonuniformity.

VII. Conclusion

We compared the performance of four detection methods, IUWT [4], FPD [7], H-Dome [8] and LoG [9]. Our study included two types of synthetic images as well as a real image obtained using fluorescence microscopy. The results from experiments on the synthetic images indicated that some of the proposed methods are vulnerable to noise as noise increases, with the IUWT showing to be more sensitive on type A images. However, the results from real images indicate that the difference in performance of the methods is comparatively small. The results show that the Laplacian of Gaussian method performed best overall when true positives and false positives are considered.

Acknowledgment

This work was carried out with the financial support of the Council for Scientific and Industrial Research (CSIR) and the Electrical and Electronic Engineering Department at the University of Johannesburg. The authors would like to thank the Synthetic Biology group at the CSIR for providing the fluorescence microscopy images.

References

[1] Cédric Vonesch, François Aguet, Jean-Luc Vonesch, and Michael Unser. The colored resolution of bioimaging. *IEEE, Signal Processing Magazine*, 23(3):20–31, 2006.

[2] Ming Zeng, Jianxun Li, and Zang Peng. The design of top-hat morphological and application to infrared target detection. *Infrared Physics and Technology*, 48(1), 2006.

[3] Nobuyuki Otsu. A threshold selection method from a gray-level histograms. *IEEE Trans. Syst., Man Cybern*, 9(1), 1979.

[4] Jean-Christophe Olivo-Marin. Extraction of spots in biological images using multiscale products. *Pattern Recognition*, 35(9):1989–1996, 2002.

[5] Zhang Yong-Deng, Chen Liang-Yi, and Xu Tao. Application of improved wavelet transform in biological particle detection. *Progress in Biochemistry and Biophysics*, 37(10), 2010.

[6] Bo Zhang, Jalal M Fadili, and Jean-Luc Starck. Wavelets,

ridgelets, and curvelets for poisson noise removal. *IEEE Transactions on Image Processing*, 17(7), 2008.

[7] Ivo F Sbalzarini and Petros Koumoutsakos. Feature point tracking and trajectory anaysis for video imaging in cell biology. *Journal of Structural Biology*, 151(2), 2005.

[8] Luc Vincent. Morphological grayscale reconstruction in image analysis: Application and efficient algorithms. *IEEE Transactions on Image Processing*, 2(2):1–25, 1993.

[9] Arjun Raj, Patrick van den Bogaard, Scott A Rifkin, Alexander van Oudenaarden, and Sanjay Tyagi. Imaging individual mrna molecules using multiple singly labeled probes. *Nature Methods*, 5(10):877–879, 2008.

[10] Ihor Smal, Marco Loog, Wiro Niessen, and Erik Meijering. Quantitative comparison of spot detection methods in fluorescence microscopy. *IEEE Trans on Medical Imaging*, 29(2):282–301, 2010.

[11] Pekka Ruusuvuori, Tarma Äijö, Shrif Chowdhury, Cecilia Garmednia-Torres, Jyrki Selinummi, Mirko Birbaumer, Aimée M Dufley, Lucas Pelkmans, and Olli Yli-Harja. Evaluation of methods for detection of fluorescence labeled subcellular objects in microscope images. *BMC Bioinformatics*, 11:1–17, 2010.

[12] Auguste Genovesio, Tim Liendl, Valentina Emiliana, Wolfgang J. Parak, Maité Coppey-Moisan, and Jean-Christophe Olivo-Marin. Multiple particle tracking in 3-d+t microscopy: Method and application to the tracking of endocytosed quantum dots. *IEEE Transactions on Image Processing*, 15(5):1062–1070, 2006.

[13] Jean-Luc Starck, Fionn Murtagh, and Albert Bijaoui. *Image processing and data analysis: The multiscale approach*. Cambridge University Press, New York, USA, 1998.

[14] Yoshitaka Kimori, Norio Baba, and Nobuhiro Morone. Extended morphological processing: a practical method for automatic spot detection of biological markers from microscopic images. *BMC Bioinfomatics*, 11(373):1–13, 2010.

[15] John C. Crocker and David G. Grier. Methods of digital video microscopy for colloidal studies. *Journal of Colloid and Interface Science*, 179, 1996.

[16] Wayne Rasband. Image processing and analysis in java [online available]. http://rsbweb.nih.gov/ij/. Accessed September 9, 2012.

[17] Ihor Smal. Synthetic data generator plugin [online available]. http://smal.ws/home/SyntheticDataGenerator. Accessed September 18, 2012.

Chorale Harmonization with Weighted Finite-state Transducers

Jan Buys and Brink van der Merwe
Computer Science Division, Department of Mathematical Sciences
Stellenbosch University, South Africa
janbuys@ml.sun.ac.za, abvdm@cs.sun.ac.za

***Abstract*—We approach the task of harmonizing chorales through style imitation by probabilistically modelling the harmony of music pieces in the framework of weighted finite-state transducers (WFSTs), which have been used successfully for probabilistic models in speech and language processing. The framework makes it possible to place domain-specific regular constraints on generated sequences, and to integrate models of different levels of complexity. We divide the harmonization generation process into different steps, each performed by inference through transducers. We present a method for four-part harmonization that models vertical and horizontal structure in the generated harmonizations. The weights in our transducers are learned by maximum likelihood estimation from a corpus of chorales. The predictive power of own model, as measured through entropy, is competitive with that of existing approaches.**

I. Introduction

Music is usually composed through a creative process. However, all pieces of music have structure, and a composer is usually constrained by the style of his composition. Established rules and principles that give the music certain aesthetically pleasing qualities should be followed. However, the characteristics of a good music piece cannot be fully described by such rules. The main reason for this is that a music piece should exhibit acceptable structure at a local and global level. There should be a fine balance between various musical qualities in the piece.

The harmony of a music piece, i.e., the structure of notes played simultaneously, is central to this musical structure. The harmony, usually described by chords, can be seen either as an observed variable of a piece with multiple voices, or as a latent variable of a melody. In music pieces with multiple voices, both the chords formed by the voices at each time-step (also called the vertical structure, due to the way music is written) and the melodic structure of each of the accompaniment voices (called the horizontal structure) are important. In this paper we will be concerned with the particular style of harmonization in 17th-century 4-part chorales, exemplified by the chorales of JS Bach. Chorale harmonization is an important task in Western classical music, and is studied by all students of music theory. The principles employed in chorale harmonization are transfered to many other composition tasks which involve harmony.

In this paper we present a probabilistic model to generate harmonizations for given melodies, using machine learning techniques. We focus on style imitation of chorales by JS Bach, though our model can also be applied to other genres. Bach chorales have been used extensively in music modelling (see for example [1], [2], [3]), due to their abundance, simplicity and good melodic and harmonic form. Our model is predictive, assigning a non-zero probability to every possible sequence over the given alphabet.

The harmonization procedure that we propose models the different steps in the harmonization process, similar to those followed by human composers, in a full probabilistic setting. Each step is performed by inference through a weighted finite-state transducer (WFST) cascade. Separate models are trained for major and minor pieces, as there are significant differences between their musical characteristics. Our model generates a good approximation of the real harmonization, modelling both vertical and horizontal constraints.

In the next section we give some musical background and discuss related work. Section 3 defines weighted finite-state transducers and related algorithms required for our approach. Section 4 presents the harmonization model we propose, while section 5 discusses the implementation of our model. We discuss the evaluation of our system in section 6, and give conclusions in section 7.

II. Background and Related Work

A. Musical Notation

The fundamental units of a music piece are *notes*. A note is a single sound, represented by *pitch* — how high or low the sound is, and *duration* — how long the sound is held. In a standard classical music piece, the pitch and duration of notes are governed as follows:

Pitches are named by their *pitch classes*. There are 12 classes, namely C, C#, D, D#, E, F, F#, G, G#, A, A# and B, each of which forms an equivalence class. An interval of size 12 is referred to as an *octave*. A *scale* is a sequence of pitch classes defined by the starting pitch class of the scale and the intervals between pitches in the scale. The most common scale types are the major and (natural) minor scales. The *key signature* of a piece indicates the scale that forms the basis of notes of the piece. However, a piece can also have *accidentals*, notes that are not in the scale of the key signature.

The *beats* of a music piece are constant time intervals that primarily govern the start of notes in the music. The *tempo* indicates the length of those beats. The *time signature* indicates the meter of the music, the basic grouping beats into *bars*. The

rhythm of a sequence of notes describes the duration of each note in the context of the time signature of the piece. Note durations are expressed as fractions of a "whole note" of 4 beats. Typical durations are a half note, quarter note, eight note and sixteenth note.

In general, a music piece consists of a number of voices, each voice being a single time-dependant sequence of notes. The *melody* of the music piece is the most significant voice, usually the highest. The *harmony* of the music refers to the way that different notes sound simultaneously. The harmony can be described by *chords*, predefined combinations of notes in the scale of the music piece that sound well together. A chord usually consist of three pitch classes, though there are also chords with more pitch classes. The most common chord types, those that we will focus on, are the major and minor chords.

B. Algorithmic Composition

Algorithmic composition, i.e., composition by formalizable methods, has a long tradition, and numerous procedures have been investigated [4]. The most common limitation that these approaches have is the inability to generate longer pieces of music that exhibits acceptable overall structure. One broad approach to algorithmic composition is the application of rules or algorithms chosen by the composer or programmer to create new pieces of art [5]. However, composers seldom publish the formalizable ideas that they use in their compositions [4]. The other approach is to construct a generative theory to describe music pieces in a given style [1]. This generative theory can then be used to generate new pieces of music in the given style. Just as in language modelling, one can distinguish between two approaches to such generative theories. In the first approach, knowledge engineering, rules and constraints are explicitly encoded in some logic or grammar. In the second approach, empirical induction, parameters of a statistical model are learnt from existing compositions.

There are clear parallels between the development of generative models for language and for music. However, in computational linguistics the focus is primarily on using models for analysis, while in music modelling research focuses on generation. Conklin [6] argues that the problem of music generation can be made equivalent to that of sampling from a statistical model; Models that are able to explain the structure of music pieces will also be able to generate acceptable original music pieces. A reason why this might be the case (it is not so in language modelling) is that in music the semantics (meaning) lies primarily in the structure of the music, while in language words can have meaning that lies outside the structure, by referring to objects or actions. However, more research into this relationship, and its implication in evaluating music generation systems, is needed.

Markov models have been used widely in musical style imitation, since they are simple and efficient in training and inference. To model melodies, higher order and variable order Markov chains are usually used (see, for example [7], [1], [3], [8]). The best results found are found by using a middle ground between Markov chains of low order that do not constrain the structure of generated music sufficiently, and Markov chains of high order that reproduce large fragments of the music pieces used for training.

C. Harmonization Models

Allan and Williams [2] applies hidden Markov models (HMMs) to harmonize chorale melodies. An HMM takes the melody notes as the observed sequence and the possible harmonizations (chord configurations) as hidden states. The best harmonization for a given melody is obtained from the HMM. A second HMM is used to model ornamentation, i.e., to add notes with a duration other than that of the beat to the generated harmonization voices. Ornamentation smooths the movement between notes and adds some variation. The MySong automatic accompaniment system [9] uses a similar HMM approach to generate chords to accompany a melody sung by the user.

A probabilistic graphical model approach to harmonization is proposed in [10]. Domain knowledge can be included in the model, and different levels of hidden variables are used to model non-local dependencies in the chord progressions. Specifically, it is found that a tree-structured graphical model for modelling the roots of chords over a given melody has more predictive power than an HMM model for the same task. However, a similar advantage was not found when modelling other voices in the harmony given the chord roots.

III. Weighted Transducers

Weighted finite-state transducers are automata that can be be used for the probabilistic modelling of discrete sequences. Important motivations for the use of WFSTs include the uniform representation of models and the existence of efficient inference algorithms that can be applied to them [11]. WFSTs have been used successfully in speech and language processing (see for example [12]). We now define transducers and the operations that we need to use them as probabilistic models. Then we show how Markov chains and hidden Markov models can be represented as WFSTs.

A. Finite-state Transducers

A *weighted finite-state acceptor* (WFSA) is a finite-state machine that accept a set of strings in the class of regular languages, assigning weights to the accepted strings. It has states and edges between states. Each edge is labeled with a symbol in the alphabet of the language, and a weight. A WFSA accepts a string if there is a path from a start state to a final state such that the concatenation of the symbols on the edges along the path yields the string. The symbol ϵ, denoting the empty string, can also be used as an edge label. The weight assigned to a path is the product of the weights on the edges along that path. The weight of a string is the sum of the weights of all the paths that yields that string.

A *weighted finite-state transducer* is a finite-state automaton similar to a WFSA, where each edge has an input and an output symbol. A WFST assigns weights to accepted pairs of

input-output strings, and can also be considered as a device which transforms a string in one regular language to a string in another regular language. A string is a sequence of alphabet symbols. Below we will refer to strings as sequences.

Formally (see [11]), a weighted finite-state transducer T over a semiring $\mathcal{K}$ is a tuple $(\Sigma, \Omega, Q, E, I, F, \lambda, \rho)$ given by: An input alphabet Σ; an output alphabet Ω; a finite set of states Q; a finite set of weighted transitions E contained in $Q \times (\Sigma \cup \epsilon) \times (\Omega \cup \epsilon) \times \mathcal{K} \times Q$; a set of initial states $I \subseteq Q$; a set of final states $F \subseteq Q$; an initial weight function λ; and a final weight function ρ.

Semiring abstraction allows us to define automata representations and algorithms over different weight sets and algebraic operations. A *semiring* K consists of a set $\mathcal{K}$ with an associative and commutative operation $\oplus$ and an associative operation $\otimes$, with identities $\overline{0}$ and $\overline{1}$, respectively, such that $\otimes$ distributes over $\oplus$, and $\overline{0} \otimes a = a \otimes \overline{0} = \overline{0}$. Here we use the probability semiring, where weights are probabilities and the usual summation and multiplication operators are used. When we construct a transducer the weights do not have to be normalized; we just need to normalize them when we compute probabilities or perform transducer operations dependent on the value of the weights.

Transducers that represent different levels of representation in a model are combined with the operation of *composition*. The composite transducer $T = T_1 \circ T_2$ accepts the sequence pair $(A : C)$ if and only if there exists a sequence B such that T_1 accepts the pair $(A : B)$ and T_2 accepts the pair $(B : C)$. The weight assigned to $(A : C)$ is the sum, over all possible values of sequence B, of the product of the weights given by T_1 and T_2. Composition can also be extended to a cascade of more than two transducers. *Right projection* is a unary operation on a WFST that yields a WFSA that accepts exactly the output sequences that the WFST can produce. The weight of a sequence B in the acceptor is the sum of the weights of all pairs $(A : B)$ in the WFST. Similarly, the *left projection* gives a WFSA over the input sequences of the transducer.

We will use WFSTs by the process of *application*, obtaining the result of the transformation of some input by a transducer or a cascade of transducers. The application can be *forward*, when an input sequence to the transducer is given and we want to find an output sequence, or *backward*, when the output sequence of the transducer in the cascade is given, and we want to find an input sequence that can be transformed to that output sequence. To apply a sequence to a transducer, the sequence is converted to an identity WFST that accepts only that sequence, with weight 1. That WFST is then composed with the given transducer, and the applicable projection of the composite transducer is obtained (right projection for forward application, and left projection for backward application). We can then sample from, or find the most likely sequence of, the resulting WFSA.

B. Markov Models

A *Markov chain* (MC) is a stochastic chain over a discrete number of states. The probability of a state in an nth-order Markov chain is dependent on the values of the previous n states in the chain. A sequence of symbols generated by an MC represents the states of the chain. Therefore, in the sequence $q_1, q_2, ..., q_m$ the following assumption holds if $t \geq n$:

$$P(q_{t+1}|q_t, ..., q_1) = P(q_{t+1}|q_t, q_{t-1}, ..., q_{t-n+1})$$

We can represent a Markov chain as a WFSA as follows: The alphabet of the WFSA is the set of symbols representing the state space of the MC. Each state of the WFSA encode the history of the previous n states in the MC. A transition in the WFSA is labeled with a symbol representing the next state of an MC transition, and its weight represents the probability of that transition. It follows from this representation that, when we ignore probabilities, MCs generate a class of languages (with the state names as alphabet symbols) that is a strict subclass of the regular languages.

We can learn the weights of this WFSA by maximum likelihood estimation, as for an MC. The $(n+1)$-gram counts of the sequences in the training data being modeled are the sufficient statistics. The probability of a transition between states q_t and q_{t+1}, given the state history, is:

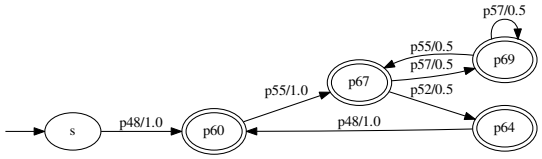
$$\frac{\text{count}(q_{t-n+1}, ..., q_t, q_{t+1})}{\text{count}(q_{t-n+1}, ..., q_t)}$$

To make the model predictive, we add smoothing to the Markov chains we use in our models. We use Katz's back-off model [13], a smoothing method often used in language models for speech recognition. In a higher-order MC, when an n-gram does not occur, we recursively back off to the highest-order MC for which the corresponding m-gram suffix of the n-gram has a non-zero probability. This is an appropriate smoothing technique for music sequences, due to its similarity to the variable-order Markov chains that have been used successfully for melodic modelling. For the models we use here, it was sufficient to use a second-order Markov chain (a trigram model). Counts of n-grams whose frequency is lower than a threshold k (we use $k = 5$, the most common choice for k) are lowered using Good-Turing re-estimation, and the freed up fractional frequency counts are redistributed to assign back-off probabilities to lower-order MCs. Let n_r be the number of n-grams that occur exactly r times in the training data. Then the discount coefficient for n-grams, where $1 \leq r \leq k$, is

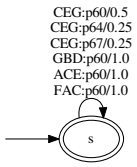
$$d_r = \frac{\frac{(r+1) \cdot n_{r+1}}{r \cdot n_r} - \frac{(k+1) n_{k+1}}{n_1}}{1 - \frac{(k+1) n_{k+1}}{n_1}}$$

.

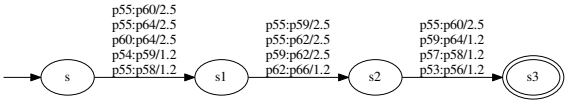
A *hidden Markov model* (HMM) [14] models the relationship between two sequences, a hidden sequence and an observed sequence. A discrete HMM can be represented by a cascade of two WFSTs. The first transducer is an MC for the hidden symbol sequence. The second transducer models the state emission probability distributions. This transducer has a single state, takes as input the hidden sequence, and gives as output the observed sequence. Every transition in this transducer has an input symbol from the hidden sequence alphabet, an output symbol from the observed sequence alphabet, and a



(a) Bass voice acceptor



(b) Chord to pitch transducer



(c) Inner voices transducer

Fig. 1. Example transducers for harmonization generation

weight representing the probability of the observed symbol given the hidden symbol. For a given observed sequence, we can find a corresponding hidden sequence by backward application to the transducer cascade. In our models, we estimate the weights in the HMM by maximum likelihood estimation (separately for the two transducers) on sequence pairs in the training data, as both the observed and the hidden sequences are known during training.

The framework of WFSTs makes it possible to place regular constraints on sequences generated by Markov chains or Hidden Markov models. For example, we can constrain the length of a sequence by constructing an acceptor of all sequences of a specific length. We then compose that acceptor with the Markov chain WFSA to get a state-machine that represents the same MC, but only accepts sequences of the required length. It is also possible to compose a Markov model with a non-Markov model that represent some domain knowledge, and which is representable as a WFST.

IV. Harmonization

We present a model to harmonize given melodies in the style of four-part chorale harmonization. Our harmonization procedure has two steps. Firstly, we find the optimal chord sequence for a given melody. Secondly, we generate three additional voices (the bass voice and two inner voices) so that the implied harmony corresponds to the generated chord sequence. In the approach proposed in [2], the chord representation includes the configuration of notes in the harmonization voices. The advantage of our model is that we are able to model explicitly the voice movement of the harmonization voices, which is important for good harmonizations.

In our harmonization generation system we generate music by performing inference through application to transducer cascades. At each step, the Viterbi algorithm is used to find the optimal sequence. Examples of the WFSTs used in the cascades are given in Figure 1. We give a schematic representation of the transducer cascades used, in Figure 2. Each transducer's input and output sequences are given. For each of the transducer cascades we also give corresponding probabilistic graphical model representations in Figure 3.

A. Chord Analysis

To model the harmony, we first analyze the chords in the music piece (see [15] for an overview of procedures). We use a template-based method to assign a chord to each beat of the music piece. The template chords we use are the 12 major chords, the 12 minor chords, and the empty chord (corresponding to no chord classification made).

For chord classification, the notes in a beat are represented by a vector. Each element in the vector represents the duration of notes in the beat corresponding to one of the 12 pitch classes. We represent our template chords similarly: The three pitch classes are each represented by the duration of a beat, but the tonic of the chord is represented by twice that value, due to its importance. We classify the notes in each beat to the template chord for which the Euclidean distance between the vector representation of the beat notes and the template chord is a minimum.

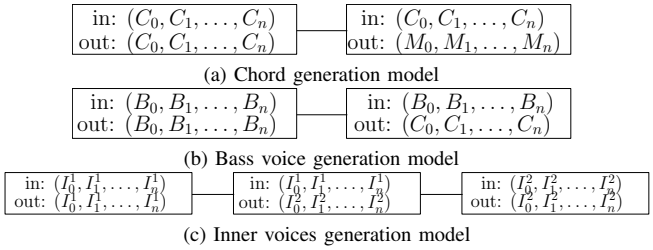
B. Chord Generation

We use an HMM approach to find the optimal chord sequence for a given melody. The chord sequence is seen as the hidden sequence, modelled with a (higher order) Markov chain, and the melody is seen as the observed sequence. The relationship between the chord and melody notes in each beat in the music is modeled.

The melody sequence symbols each represent the pitches and rhythm of a beat in the music. The chord sequence symbols are template chord names. We model chord generation with a WFST cascade, given in Figure 2a, where the first transducer is a Markov chain for chords and the second transducer is a single-state chord to melody transducer. The corresponding graphical model is given in Figure 3a. To do inference, we apply a melody symbol sequence to the cascade, and find the optimal chord sequence with the Viterbi algorithm.

C. Bass Voice Generation

To generate harmonization voices, we first generate a bass voice, and then two inner voices. In the training data, we identify the base voice as the voice that is, on average, the lowest in a music piece. In the model for the bass voice we work with a representative pitch sequence for the bass note sequence. When there is more than one pitch in the same beat, the longest or first pitch is chosen. We model the relationship



(a) Chord generation model

(b) Bass voice generation model

(c) Inner voices generation model

Fig. 2. Transducer cascades for the harmonization model

between the bass voice and the chord sequence with a hidden Markov model, in a similar way to the chord sequence and melody. Now the chord sequence is the observed sequence and the bass note sequence is the hidden sequence. Given a chord sequence, we sample from the distribution of bass notes for that chord sequence. The transducer cascade is given in Figure 2b and the graphical model in Figure 3b.

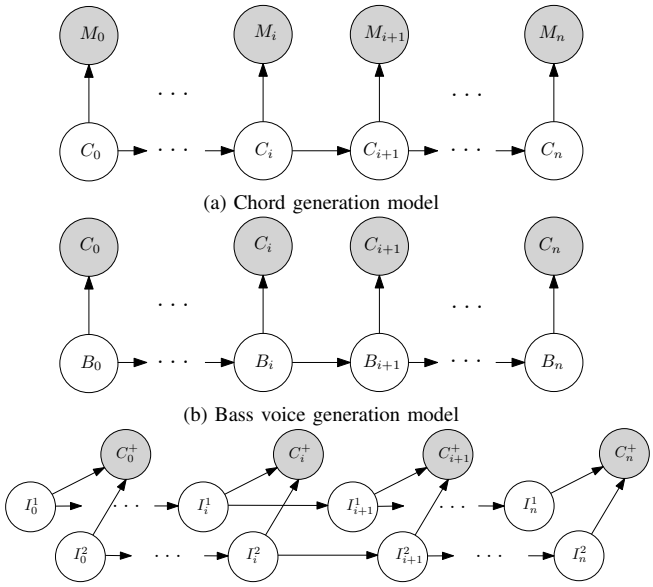
D. Inner Voices Generation

We generate two inner voices that, together with the melody and bass voices, give the implied harmony of the chord at each beat. To do this, we model the vertical constraints on, and the horizontal probability distribution over the generated voices. We use a smoothed Markov chain trained over all the inner voices in the training data to model the horizontal structure. There is usually an overlap in the ranges of the inner voices, and in training a single model for the inner voices we do not need to restrict the training data to exactly four voices.

Vertically, we model the pair of inner voice sequences so that the four voices of the harmony will together represent the chosen chords at each time-step. We restrict the range of the voices such that none of the four voices may cross each other (a lower voice may never have a higher pitch than a higher voice in the same beat). The motivation for this constraint is to remove spurious ambiguity from the model. We want to give a strong preference to inner voice pairs at a beat such that all three pitch classes of the chord should be contained in the four notes at the beat. If there is no configuration satisfying that preference, we give preference to assigning one note, in a pitch class not yet represented by the melody or bass voice of the beat, to both voices.

We want to model the two inner voice sequences given the melody, chord and bass sequences. We represent the already-known sequences with the sequence C^+, where C_i^+ encodes the melody note, bass note and chord at time-step i. The graphical model representation of the joint distribution over these variables is given in Figure 3c. The distribution factorizes as follows:

$$\begin{aligned}P(I^1, I^2, C^+) =& P(I_0^1, ..., I_n^1, I_0^2, ..., I_n^2, C_0^+, ..., C_n^+) \\ =& P(I_0^1)P(I_0^2)P(C_0^+|I_0^1, I_0^2) \\ & \bullet \prod_{i=1}^{n} P(I_i^1|I_{i-1}^1)P(I_i^2|I_{i-1}^2)P(C_i^+|I_{i-1}^1, I_{i-1}^2)\end{aligned}$$



(a) Chord generation model

(b) Bass voice generation model

(c) Inner voices generation model

Fig. 3. Graphical models for the harmonization model

As we are working with the case where $C_1^+, ...C_n^+$ is given, the conditional distribution is:

$$\begin{aligned}P(I^1, I^2|C^+) =& P(I_0^1, ..., I_n^1, I_0^2, ..., I_n^2|C_0^+, ..., C_n^+) \\ =& P(I_0^1)P(I_0^2)\frac{P(C_0^+|I_0^1, I_0^2)}{P(C_0^+)} \\ & \bullet \prod_{i=1}^{n} P(I_i^1|I_{i-1}^1)P(I_i^2|I_{i-1}^2)\frac{P(C_i^+|I_{i-1}^1, I_{i-1}^2)}{P(C_i^+)}\end{aligned}$$

We model this distribution with a transducer cascade as follows: The first transducer is a Markov chain for the first inner voice, the second transducer models the vertical structure — the acceptability of inner voices at every time step (taking the first inner voice as input and giving the second as output) and the third transducer is a Markov chain for the second inner voice. This transducer cascade is given in Figure 2c. From the probability distribution factorization it follows that the weight of a transition between time-steps $i-1$ and i in the second transducer should be $\frac{P(C_i^+|I_{i-1}^1, I_{i-1}^2)}{P(C_i^+)}$.

Let α and β be weights that indicate our preference for pure chords (all three pitch classes are represented) and impure chords (any other note combination) respectively. Here we choose $\alpha = 0.8$, and let $\beta = 1 - \alpha$. The reason for this choice is that we want to give a strong preference to pure chord representations.

Suppose $P(C_i^+) = \frac{1}{m}$, where m is the number of possible chord combinations. Let p_i be the proportion of possible inner voice combinations that represent pure chords, given the bass and melody notes at time step i. The transition weight is then $\frac{\alpha}{\alpha \cdot p_i + \beta \cdot (1-p_i)}$ for pure inner voice combinations and $\frac{\beta}{\alpha \cdot p_i + \beta \cdot (1-p_i)}$ for impure combinations.

E. Ornamentation

In general, harmonization voices are not played in blocks, at every beat in the music. Repeated notes may be combined into one longer note, and extra notes can be inserted to improve voice movement (the most common example is to insert a middle note if there is an interval of a third between two notes). This process is known as ornamentation. An HMM model for ornamentation is proposed in [2]. We implement a similar model. We first encode the pitches and rhythm of a note sequence at each beat as a single symbol. The ornamented note sequence is then modelled as the hidden sequence of the HMM, and the representative note sequence as the observed sequence. The ornamentation transducer cascade can be applied to ornament the bass voice and the two inner voices.

However, this ornamentation procedure is limited in its ability to reproduce ornamentations of quality comparable to the ornamentations in our training data. Another limitation is the inability to model parallel or diverging movement in pairs of voices, as the ornamentation of different voices is modeled independently. Excessive or uncoordinated ornamentation may decrease the quality of harmonizations. We therefore propose that further work should be done on ornamentation, building on the ability of our approach to model vertical and horizontal structure in harmonizations. We do not include ornamentation in our evaluation below.

V. Implementation

In this section we give a brief overview of the implementation of our chorale harmonization generation system. The main steps in the system are:

1) Analyse a corpus of given music pieces.
2) Learn the parameters of a WFST-based model for harmonization.
3) Generate new harmonizations for given melodies, using the trained model.

In our implementation we use the finite-state transducer package Carmel [16] for performing operations on the transducer models. Carmel can train and compose transducers, sample sequences or get sequence probabilities from transducers.

To represent music pieces our system uses MIDI, a standard music file format that represents a music piece by event messages about the music, rather that with an audio signal. In our implementation we use the Java package JMusic [17] to extract a symbolic representation corresponding to standard music notation from a MIDI file. We extract a pitch sequence and a rhythm sequence for each of the voices in a music piece. A pitch value is represented as a MIDI pitch value, an integer between 0 and 127 that represents the number of semi-tones the note is higher than the note 5 octaves below middle C. For our model, we transpose the pitches of all the training music pieces to the key of C major or A minor, for pieces in a major or minor key respectively. The rhythm sequence represents the durations of notes and rests, as well as bar separators.

We represent rhythm (note duration) values with integers directly proportional to the note duration, with 96 representing a whole note. However, as these note values are not always represented precisely in the MIDI files, we had to approximate imprecise values to the nearest discrete value in our representation. We also inferred the bar structure in the music pieces, and used that to ensure the correct alignment of notes to beats.

In our implementation we store the transducers and note sequences as text files in the format required by Carmel. We then use Carmel to perform inference by application to transducer cascades, using the models described above. Finally, the generated sequences for the harmony (in text format) is converted back to a MIDI file of the generated harmonization.

TABLE I
AVERAGE ENTROPY OF HARMONIZATION MODELS ON TRAINING AND TESTING SETS

Model	Major Train	Major Test	Minor Train	Minor Test
$P(C)$	2.503	2.796	2.748	3.171
$P(C\|M)$	1.517	1.798	1.851	2.219
$P(B)$	4.259	4.144	4.382	4.407
$P(B\|C)$	2.463	2.377	2.463	2.377
$P(I^1, I^2)$	3.727	3.957	3.859	4.182
$P(I^1, I^2\|M, C, B)$	2.886	3.137	3.090	3.330
$P(H\|M)$	6.866	7.312	7.364	7.949
$P(H\|M)$ in [2]	3.693	7.069	3.838	7.242

VI. Evaluation

The evaluation of our models is based on a publicly available corpus, in MIDI format, of chorales by JS Bach[1]. From this corpus we used 350 chorales in four-part harmony, evenly split between chorales in major and minor keys. We trained separate models for chorales in major and minor keys. For both models, the chorales were divided into a training set (60%) and a testing set (40%). Our evaluation is based on an estimation of entropy – the negative log likelihood per symbol that a model gives to music pieces in the testing set. For a sequence $S = s_1 s_2 \dots s_N$, the value $-\frac{1}{N}\sum_{i=1}^{N} log_2 P(s_i|s_1, \dots s_{i-1})$ is used.

The average of this measurement is taken over all examples in the testing set. This evaluation method has been used to evaluate harmonization in [2]. It evaluates the predictive power of a model by measuring the likelihood that is assigned by our model to compositions that we are trying to imitate. The lower the entropy, the higher the probability that our model gives to the music pieces. We compute the entropy for different components of our model separately, and then add them to find the entropy of the model. Table I gives the average entropy of our models. We include the entropy of the training and testing sets. For each of the models, we compare the result against a baseline Markov model that is not conditioned on other sequences.

The predicted sequences are represented by the following symbols: The melody, M, the chord sequence, C, and the harmonization voices, H (representing the bass voice B and inner voices I_1 and I_2). We include the results of Allan and

[1] http://www.jsbchorales.net/down/sets/jsb403.zip

Williams [2] for comparison. Note that we convert their scores from $\log$ base e to $\log$ base 2 to represent entropy.

In the evaluation of our models, we find that the entropy of the training data is in each case smaller than that of the testing data, as should be expected, but only by a small margin. This shows that our smoothed models (using Katz's back-off model) are very robust in dealing with sparse data, and performs almost equality well on seen and unseen data. In contrast, the results of [2] show a large difference between the testing and training training data. This gives some evidence that their model overfits the training data.

The results show that for each of the models an improvement is obtained over the baseline model, where the sequence is independent of other sequences. This show that our model is indeed modelling the dependencies between sequences.

The results we obtain from our model is competitive with the results of [2]. One reason our model does not perform better is that, with enough data, that model will also model movement in individual harmonization voices, as all the notes at a beat are encoded in a single symbol. However, our model should be more scalable, as we explicitly model horizontal movement in harmonization voices.

The evaluation approach we follow here allows us to evaluate a model hypotheses quantitatively and to compare different hypothesis. A limitation of this approach is that the model has to be predictive. In practice, one might generate better quality music by placing more hard constraints on the generated harmonizations. Specifically, it might be beneficial to constrain the inner voices to let only pure chord combinations be generated.

An alternative way to evaluate our harmonization system would be to let a music expert panel judge the quality of the harmonizations and their conformity to standard harmonization rules.

VII. Conclusion

In this paper, we proposed a model for the harmony of music pieces, specifically chorales, using the framework of weighted finite-state transducers. This framework is flexible and extendible, making it possible to construct models that encode different sets of dependencies and restrictions. We showed how WFSTs can model different steps in the harmonization process in a probabilistic setting, while encoding algorithmic processes that composers may be following when they compose pieces of music. The results show that our procedure is successful in modelling dependencies in the horizontal and vertical structure of the music.

For future work, models for non-local structure in the chord sequence generation model should be investigated. Specifically, tree-based approaches should be considered – a non-probabilistic tree-based approach for music modelling has been proposed in [18]. The restriction on chords types in our model can also be relaxed.

Another avenue for further investigation is the effect of removing some of the independence assumptions we made by dividing the harmonization process into different steps. Better models for non-local structure in the harmonization should also be developed. The ornamentation procedure we mentioned can be refined. Related to that, approaches to modelling parallel and diverging movement between voices in the harmonization should be investigated. The goal should be to fully model the richly structured harmonizations of JS Bach.

Acknowledgment

The first author would like to thank the financial support of the Wilhelm Frank Bursary fund and the MIH Media Lab.

References

[1] D. Conklin and I. H. Witten, "Multiple viewpoint systems for music prediction," *Journal of New Music Research*, vol. 24, no. 1, pp. 51–73, 1995.

[2] M. Allan and C. K. I. Williams, "Harmonizing chorales by probabilistic inference," in *Advances in Neural Information Processing Systems*, vol. 17. MIT Press, 2005, pp. 25–32.

[3] J. L. Trivino-Rodriguez and R. Morales-Bueno, "Using multiattribute prediction suffix graphs to predict and generate music," *Computer Music Journal*, vol. 25, no. 3, pp. 62–79, 2001.

[4] G. Nierhaus, *Algorithmic Composition: Paradigms of Automated Music Generation.* SpringerWienNewYork, 2009.

[5] M. Edwards, "Algorithmic composition: Computational thinking in music," *Communications of the ACM*, vol. 54, no. 7, pp. 58–67, 2011.

[6] D. Conklin, "Music generation from statistical models," in *Proceedings of the 2003 Symposium on Artificial Intelligence and Creativity in the Arts and Sciences*, 2003, pp. 30–35.

[7] F. P. Brooks, A. L. Hopkins, P. G. Neumann, and W. V. Wright, "An experiment in musical composition," *IRE Transactions on Electronic Computers*, vol. 5, pp. 175–182, 1957.

[8] W. Schulze and B. Van der Merwe, "Music generation with markov models," *IEEE Multimedia*, vol. 18, no. 3, pp. 78–85, 2011.

[9] I. Simon, D. Morris, and S. Basu, "Mysong: Automatic accompaniment generation for vocal melodies." in *Proceedings of the 2008 Conference of Human Factors in Computing Systems*. ACM Press, 2008, pp. 725–734.

[10] J. Paiement, D. Eck, and S. Bengio, "Probabilistic melodic harmonization," in *Proceedings of the 19th Canadian Conference on Artificial Intelligence*, ser. Lecture Notes in Computer Science, vol. 4013. Springer, 2006, pp. 218–229.

[11] M. Mohri, "Weighted automata algorithms," in *Handbook of Weighted Automata*. Springer, 2009, pp. 213–254.

[12] M. Mohri, F. Pereira, and M. Riley, "Weighted finite-state transducers in speech recognition," *Computer Speech and Language*, vol. 16, pp. 69–88, 2002.

[13] S. M. Katz, "Estimation of probabilities from sparse data for the language model of a speech recognizer," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 35, no. 3, pp. 400–401, 1987.

[14] L. R. Rabiner and B. H. Juang, "An introduction to hidden markov models," *IEEE ASSP Magazine*, pp. 4–16, 1986.

[15] N. Jiang, "An analysis of automatic chord recognition procedures for music recordings," Master's thesis, Saarland University, 2011.

[16] J. Graehl, "Carmel," 2008, available at: http://www.isi.edu/licensed-sw/carmel.

[17] A. Sorensen and A. R. Brown, "Introducing jmusic," in *InterFACES: Proceedings of the Australasian Computer Music Conference*, 2000, pp. 68–76.

[18] F. Drewes and J. Högberg, "An algebra for tree-based music generation," in *Proceedings of the 2nd international conference on Algebraic informatics*, ser. Lecture Notes in Computer Science, vol. 4728. Springer, 2007, pp. 172–188.

Multilingual pronunciations of proper names in a Southern African corpus

Jan W.F. Thirion, Marelie H. Davel and Etienne Barnard
North-West University, Potchefstroom, South Africa
E-mail: {thirionjwf,marelie.davel,etienne.barnard}@gmail.com

***Abstract*—We present our process for the development and analysis of a multilingual names corpus, called Multipron-split. It is derived from Multipron, a corpus collected in previous work [1], where names and speakers were drawn from four South African languages, namely Afrikaans, English, isiZulu and Sesotho. The new corpus is more suited for multilingual pronunciation modelling and research as the "words" consist of either a name or surname, rather than a combination of the two. This enables us to model pronunciations from a single language of origin, which has previously been shown to be important in pronunciation modelling for proper names. An algorithm is presented through which the most common pronunciations of names, also called reference pronunciations, can be automatically extracted from the observed pronunciations. We show that the most common pronunciation variants correlate well with the different speaker languages, and that systematic phone substitutions occur when speakers of one language pronounce names from a different language. Also, reasonably accurate automatic pronunciations can be generated with an automatic grapheme-to-phoneme converter, especially when the speaker language agrees with the name language.**

I. Introduction

Various factors such as a speaker's region of origin, mother tongue, age and socio-economic background result in systematic pronunciation differences between speakers [2]. In a multilingual environment, such as in South Africa, this issue is particularly prominent, since most automated speech-processing systems will be required to operate on speech from speakers with a variety of linguistic backgrounds. In particular, it is generally accepted that poor pronunciation modelling can lead to deteriorated automatic speech-recognition (ASR) performance [3]; this is especially true for multilingual proper names as well as loan words, where native pronunciation rules are often inaccurate [4]. For resource scarce environments, such as in South Africa, dealing with this problem adequately remains a challenge [5], [6], especially since resource-scarce languages are currently less important economically to the providers of commercial speech-recognition systems. Speech recognition of proper names is particularly important in applications such as voice search, directory assistance and automated attendants [7], [8], [9].

It is impractical to create a dictionary by hand with all possible pronunciations of all names in all languages (both because of the time and cost involved, and because of the inevitable inaccuracies that will result from such a process). Hence, pronunciation rules are often employed to predict pronunciations [3]. There is a need for a corpus on which the pronunciation rules for South African languages can be trained, where the linguistic origin of the name is taken into account. Earlier work [1] resulted in a multilingual corpus, called Multipron, for four South African languages, but these combined name/surname pairs typically had mixed languages of origin for the names and surnames, making pronunciation modelling problematic.

In this paper we present our process of transforming the Multipron corpus into a "split" corpus, Multipron-split, of individual names and surnames, tagged by their associated language of origin. We then automatically extract the typical pronunciation as would be produced by a native speaker of each name from the pronunciations in the corpus (observations). The Default&Refine algorithm [10] is then used as G2P converter to predict these reference pronunciations. An interpretation of the results gives insight into the structure of the corpus and the variants contained therein.

II. Background

It is well known that knowledge of the mother tongue of the speaker, as well as the linguistic origin of the word, can be beneficial to producing better pronunciation variants [11]. The consistency of cross-lingual pronunciation of proper names was recently studied for four South African languages, namely Afrikaans, English, Setswana and isiZulu [4]. It was confirmed that knowledge of the linguistic origin of each word was an important factor in predicting how it would be pronounced.

The Autonomata Spoken Names Corpus (ASNC) [12] was recently used in state-of-the art work most related to our current investigation [13], [14], [15], [16]. The database contains 3540 unique names of Dutch, French, English, Turkish and Moroccan origin. The corpus contained only names of people (personal names and surnames), street names and city names in a single language (i.e no mixed language names).

In [17] a tandem G2P-P2P approach was used for the G2P conversion of proper names, where an initial transcription generated by the G2P converter is passed to a P2P converter, along with the orthography of the word. The P2P converter applies learned rules (in the form of decision trees or rule networks, automatically learned from the data) that generate alternative pronunciations. In [18] this method was shown to work well for the G2P conversion of proper names, although the linguistic origin of the word was not taken into account. In [13] it was found that ASR accuracy for proper names

increased when pronunciation variants were added to the lexicon. This was true for native speakers speaking foreign names, but not for foreign speakers. Here "native" refers to the target language of the system (e.g. Dutch) and foreign, or "non-native" include English, French and Moroccan.

A study on how mother tongue and the linguistic origin of the word affect ASR performance, was reported in [14]. Language-specific G2P converters were used, both monolingual as well as multilingual acoustic models, and language-specific P2P converters. It was found that native speakers used their own non-native G2P rules when pronouncing unfamiliar words from the non-native language and not knowledge from the G2P rules from their native language. Non-native speakers, however, tended to employ their own non-native G2P rules when pronouncing unfamiliar words from the native language, resulting in substantial error increases. When the speaker's mother tongue was used as basis for selecting variants (from that language) for the recognition of foreign names, performance decreased. Also, names with linguistic origins of languages different from that of the native/target language of the system, were found to be easier to recognise due to the names having less chance of being confused with the pronunciations of the native language. An experiment was also done to investigate whether ASR performance increases if the correct transcription is always added to the lexicon. It is encouraging that improved ASR accuracies were observed for all native/non-native combinations. Better pronunciation prediction algorithms may thus lead to even more improved ASR accuracy as the lexicon will contain even better coverage of the true transcriptions.

The work in [15], [16] can be considered as the current state-of-the-art in the multilingual recognition of proper names using knowledge of the speaker's mother tongue and the linguistic origin of the word. Here it was found that nativised transcriptions [19] are appropriate as target transcriptions for P2P learning. P2P transcriptions improved ASR accuracy of non-native words by a native speaker, but not significantly for native and non-native words by a non-native speaker. Automatically generated P2P transcriptions compete well with typical transcriptions from human experts. For non-native words, speakers will attempt to use the non-native G2P rules of that language; hence, knowledge of the speaker's mother tongue is important for accurate P2P converters. When a P2P converter was trained on foreign names, it outperformed a P2P converter trained exclusively on native words.

From the work above, many unanswered questions remain. For example, it is unclear what benefit task-specific (trained on the same type of data we are trying to predict, taking language of origin into account), rather than language-specific G2P converters would have. It is also important to see how well the results obtained generalise to the South African languages. However, in [1], the names and surnames form a word in which the constituent parts could be of different language origins, making pronunciation analysis difficult – hence the need to create a "split" corpus in order to address these questions.

III. Approach

A. The Multipron "split" corpus

In order to split the first name-surname pairs in Multipron, we started with grapheme-to-phoneme alignment of the dictionary. Dynamic programming was used, with the orthography as the reference string (with a special symbol "=" used to join first names and surnames) and a manual transcription as the observation. (These manual transcriptions were created as an approximate starting point for further development by a first-language Afrikaans speaker, after listening to a few samples of each name.) An automatically trained scoring matrix with no gap extension penalties, based on the Needleman-Wunsch algorithm, was used for alignment [20]. Log-likelihood probabilities were used in the scoring matrix. Next, the aligned sequences were split where "=" was aligned to a gap. This resulted in a separate name and surname. In a few cases, the alignment could not be done (e.g. due to incorrect transcriptions), and these were inspected manually. There were 3 such name-surname combinations, of which 3 individual words could not be used. Hence, from the 10130 entries in the dictionary we generated 20257 individual words.

Word boundary effects were subsequently manually checked and corrected. All double graphemes at word boundaries in the orthography (first name ends in the same grapheme as the first grapheme in the surname) were marked to be checked. All double phonemes in the transcriptions (at the first name/surname boundary) were also marked, but none of these required manual intervention. For all /r/ phonemes that were dropped during the splitting process from the first names, no changes were made. All /l/ phonemes split off from the first name resulted in the phoneme being added to the transcription of the first name (at the end). Finally, double-consonant effects were corrected, as well as nasals. Table I shows a few of these manually corrected examples.

Uncorrected	Corrected
amber__rennie	*{ m b @ r\ E n i*
{ m b @ r\ E n i	{ m b @ r\ E n i
donald__day	*d Q n @ l d @i*
d Q n @ l d @i	d Q n @ l d @i
peaceful__lottering	*p i s f @ l Q t r\ @ N*
p i s f @ l Q t r\ @ N	p i s f @ l l Q t r\ @ N
hellen__nzwakele	*h E l @ n z v a k E l E*
h E l @ n z v a k E l E	h E l @ n n z v a k E l E
markus__stoop	*m a r k @ s t u@ p*
m a r k @ s t u@ p	m a r k @ s s t u@ p
jeanett__taylor	*d Z @ n E t @i l @ r*
d Z @ n E t @i l @ r	d Z @ n E t t @i l @ r

TABLE I
EXAMPLES OF MANUALLY CORRECTED "SPLIT" WORDS.

B. Reference extraction

For pronunciation variation analysis and evaluation, the typical pronunciation of a word by a native speaker is needed, called references here. These can either be obtained from experts, or be extracted automatically. In the work presented here, a semi-automatic process was employed.

In order to create reference pronunciations, the following was done:

1) **Extract references**: References were extracted by first counting the number of occurrences of every observation/transcription for every word (orthography) from a given language origin, per speaker language. The observation (per speaker language) with the most occurrences was taken as the starting reference. If a name was not pronounced by a certain speaker language, then speaker language was ignored and the observation with the maximum occurrence irrespective of speaker language taken as the starting reference for that word-speaker language. A scoring matrix was then trained [20] between the transcriptions/observations and starting references. The average dynamic programming (DP) score between all observations of a word, per speaker language, was then computed. Two methods of reference selection were compared:
   - **OPTMAX**: We take the reference to be the transcription with the highest average DP score per speaker language. If there are ties (unlikely) then the transcription with the highest number of occurrences is taken. If there are still ties, then the first transcription is taken.
   - **MAXOPT**: We take the reference to be the transcription with the highest number of occurrences per speaker language. Ties are resolved by taking the transcription with the highest average DP score (to all observations) as the reference.

   Names that were not pronounced in certain speaker languages were again treated in the appropriate language-independent fashion for the respective reference selection method (i.e. the observation with the maximum number of occurrences or maximum average DP score, irrespective of speaker language was taken as the reference).
   A total of 5176 unique "split" references were extracted in this way. The reference for a name-surname combination is then the reference per speaker language for each part (name or surname) of the entry independently.
2) **Manual correction**: The references where the speaker language and name language were the same ("in-language") were checked and corrected by a human expert. It is assumed here that speakers with the same home language as that of the word origin would know best how to pronounce it.
3) **Create references**: The "in-language" references were used as the reference for every word. If a reference was not available for a word in a specific language, then one was selected from the other languages. Table II shows the preferences given. No attempt was made in this work to investigate how similar languages are.

	Language			
Choice	A	E	Z	S
1	E	A	S	Z
2	Z	Z	E	E
3	S	S	A	A

TABLE II
SELECTION OF ALTERNATIVE "REFERENCES" FROM "IN-LANGUAGE" REFERENCES FROM OTHER LANGUAGES WHEN AN "IN-LANGUAGE" REFERENCE FOR A WORD DOES NOT EXIST.

If a reference could still not be found, an automatic reference from step 1 (for the same speaker language and name language as the word in question) could be used as back-off; however, we did not encounter any such cases. Name and surname references were combined to give references for all entries in the original Multipron dictionary. There were 261 Afrikaans, 517 English, 254 isiZulu and 262 Sesotho "in-language" references, for a total of 1294.

C. Reference prediction

Default&Refine [10] is a rule-based algorithm that can be used to perform grapheme-to-phoneme (G2P) conversion. In our task here, it is used to extract rules from the pronunciation dictionary and then predict the pronunciation of the references from the orthography alone. We explore two cases:

- Generic G2P trained on a variety of texts is used to predict the pronunciations for the names. The G2P rules were language-dependent, based on the Lwazi corpus [21].
- A task-specific G2P is developed, with rules trained on the names corpus developed here, using 10-fold cross-validation.

For both these cases, we compare the prediction results against the references extracted from the MAXOPT, OPTMAX and manually verified references. We consider two cases for each of these "target" reference sets:

- References dependent on speaker language only (we temporarily ignore name language).
- References dependent on speaker language and name language, where the speaker language and name language are the same - the so-called "in-language" references.

D. Variant analysis

The variant analysis we present gives insight into the reasons behind the variation between observed pronunciations and reference pronunciations. We take the manually verified reference pronunciations and extract simple P2P rules (no context) using the observed pronunciations. The resultant rules have the form

$p_r \rightarrow p_o$

where p_r is the phoneme from the reference pronunciation and p_o is the phoneme from the observed pronunciation.

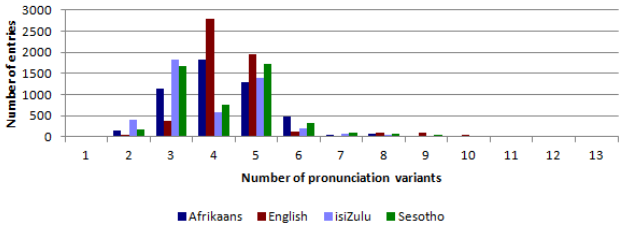


Fig. 1. Relationship between the number of entries (over all speaker and name languages) and the number of variants for an entry.

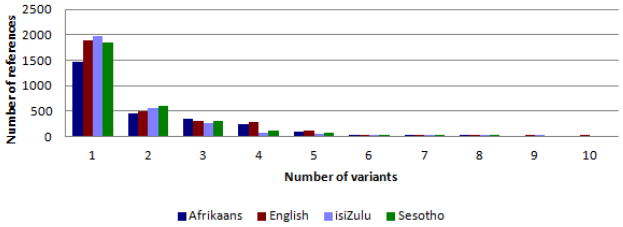


Fig. 2. Relationship between the number of references for each of the "in-language" references and the number of variants for such a reference.

IV. Experiments and Results

A. Corpus analysis

	Speaker language			
Name language	A	E	Z	S
A	1069	1144	912	976
E	2105	2289	1872	2020
Z	961	1063	907	985
S	971	1094	895	994

TABLE III
Number of entries in the Multipron-split corpus based on speaker language (mother tongue) and name language (language origin).

Table III shows the number of entries in the Multipron-split corpus split according to speaker and name language. A total of 20257 entries exist and these are fairly evenly distributed over the speaker and name language pairs, except for English words, which (by design of the Multipron corpus) were more frequent than those from other languages.

Figure 1 shows how many entries exist in the corpus with a given number of variants. The graph gives some insight into the variedness of the pronunciations – we see that most words in the corpus have around 3 to 5 pronunciation variants. This correlates well with Figure 2, from which we deduce that most variants consist of a single pronunciation in each of the speaker language/name language pairs.

Figure 2 shows the relationship between the number of references for each of the "in-language" references and the number of variants for such a reference. Here it can be seen that most of the "in-language" references (dependent on both a speaker and name language) had only a single pronunciation variant which we selected as reference . This is likely to be a typical scenario for multilingual corpora, due to the scarcity of data.

B. Reference extraction

To evaluate how well the reference extraction methods worked, we compared references extracted with those from a manually corrected version of the references. Dynamic programming alignment (Needleman-Wunsch) was performed, where a similarity score of 2 was given if the symbols were identical, -2 for a gap and -1 if they differed. Accuracy (Acc) was calculated as the average accuracy over all of the "in-language" references. The accuracy (percentage) for a single reference was calculated as:

$$Acc = 100 \cdot \frac{Num - Ins - Del - Sub}{Num}$$

(See the Appendix for additional information on the difference between accuracy and correctness, as well as other definitions of terminology.) We also counted the number of references that were perfectly predicted.

Lang	Acc	$Perf/T$	Ins	Del	Sub	Num
A	94.29	200/261	8	16	67	1564
E	94.84	411/517	23	23	93	2711
Z	94.29	178/254	5	22	72	1735
S	90.27	151/262	8	17	145	1816

TABLE IV
Accuracy (Acc), Perfectly predicted ($Perf$) and Total (T) references, Insertions (Ins), Deletions (Del), Substitutions (Sub), and Number (Num) of phonemes for the MAXOPT reference extraction method.

Lang	Acc	$Perf/T$	Ins	Del	Sub	Num
A	91.69	172/261	4	27	98	1560
E	92.18	362/517	19	37	157	2707
Z	92.75	160/254	2	30	89	1732
S	88.69	132/262	6	35	161	1814

TABLE V
Accuracy (Acc), Perfectly predicted ($Perf$) and Total (T) references, Insertions (Ins), Deletions (Del), Substitutions (Sub), and Number (Num) of phonemes for the OPTMAX reference extraction method.

Tables IV and V show the accuracy of the reference extraction methods. Here it can be seen that the MAXOPT method outperforms the OPTMAX method due to the most typical variants occurring more frequently than others. When data is particularly scarce, OPTMAX may still be useful to choose the most "average" pronunciation variant as reference. The percentage of references predicted with 100% accuracy using this method ranges between somewhat less than 60% for Sesotho to almost 80% for English. The relatively low accuracy for Sesotho results from inaccurate initial transcriptions. This was confirmed in that many of the errors encountered in the Multipron corpus, most of which were corrected by hand, were of Sesotho origin. The manually corrected Sesotho references were then quite different as a result.

The result of this analysis shows that these automatically extracted references may be very beneficial as a first-round version of pronunciations. A human expert may then check these transcriptions and do the manual corrections. Such semi-automated processes can save a considerable amount of time [6].

C. Reference prediction

In this section we evaluate how well the references can be predicted from trained rules. The accuracy of the conditional pronunciation rules are evaluated directly, in order to gain insight into the predictability of pronunciations under the various combinations of causal factors, using 10-fold cross-validation. In addition, the accuracy of the conditional pronunciation rules are evaluated against the references extracted from the data. The aim is to understand how well the pronunciation rules are able to produce a base reference from which variants can be generated.

1) G2P per speaker language: Here we consider the accuracy with which we can predict the typical pronunciation, of a person with a specific first language, of a name in any of the four languages.

The experiment was performed as follows:

- The effect of name language is ignored temporarily.
- For each name pair, we obtain a reference pronunciation per speaker language (4 references per name).
- Four different dictionaries are created from these references, one per speaker language.
- Finally, we generate pronunciation rules in two different ways:
  - system A: Extract rules directly from the name data; measure accuracy using 10-fold cross-validation. Data is more closely matched, but the training set is very small.
  - system B: Extract rules from generic data, apply to full data set and measure accuracy. Now data is less closely matched, but the training sets are somewhat larger (5,000 to 100,000 words per language).

Results when extracting rules from name data (system A) and generic data (system B) are shown in Table VI. (In this table, both correctness and accuracy are reported.) From the results we see that when name language is not taken into account, task-specific rules outperform generic rules. This may be due to proper names having a less regular spelling system than other more commonly used words in the same language. The generic G2P rules are then insufficient to predict proper name pronunciations accurately.

2) G2P per name language: In this section we consider the accuracy with which we can predict the typical pronunciation of a person with a specific first language of a name in his/her own first language. This can then serve as the basis for adding variants based on the phonemic substitution rules described in Section IV-D.

Experimental setup:

- Only consider pronunciations where the name language is similar to the speaker language.

		Task-specific		Generic	
Lang	Ref	Corr	Acc	Corr	Acc
A	Manual	82.06	79.13	68.95	61.82
	OPTMAX	79.50	75.92	67.48	59.75
	MAXOPT	81.55	78.54	68.52	61.24
E	Manual	80.11	77.57	72.30	67.61
	OPTMAX	77.35	74.34	71.53	66.32
	MAXOPT	80.22	77.47	71.85	67.03
Z	Manual	82.27	78.87	79.94	69.62
	OPTMAX	79.96	76.59	77.78	66.58
	MAXOPT	81.50	78.09	79.39	68.83
S	Manual	80.91	77.76	76.04	67.80
	OPTMAX	78.91	75.30	74.90	65.52
	MAXOPT	81.43	78.29	77.07	68.76

TABLE VI
RESULTS OF G2P PREDICTION (PER SPEAKER LANGUAGE) OF REFERENCES WITH TASK-SPECIFIC AND GENERIC RULES.

- For each name, obtain a reference pronunciation per name language (1 reference per name).
- Create 4 different dictionaries from these references, one per name language.
- Generate pronunciation rules in two different ways:
  - system A: Extract rules directly from the name data; measure accuracy using 10-fold cross-validation. Data is more closely matched, but the training set is very small.
  - system B: Extract rules from generic data, apply to full data set and measure accuracy. Now data is less closely matched, but the training sets are larger (5,000 to 100,000 words per language).

Results when extracting rules from name data (system A) and generic data (system B) are shown in Table VII. The results show that when the name language and the speaker language are the same, these "in-language" reference pronunciations can be predicted more accurately than when name language is ignored (Table VI). Furthermore, generic G2P rules outperform the task-specific G2P rules. This is a direct consequence of the limited data available for training. Of interest is the lower accuracy observed for English, which is to be expected, given its less regular spelling system.

		Task-specific		Generic	
Lang	Ref	Corr	Acc	Corr	Acc
A	Manual	89.03	87.08	88.30	84.19
	OPTMAX	83.04	80.01	84.41	78.93
	MAXOPT	86.44	83.62	86.24	81.40
E	Manual	82.44	79.32	90.36	87.13
	OPTMAX	78.07	74.06	87.19	82.92
	MAXOPT	80.72	76.77	89.10	85.49
Z	Manual	96.79	96.27	97.17	96.36
	OPTMAX	91.36	89.93	93.83	91.66
	MAXOPT	93.41	92.31	94.92	93.11
S	Manual	86.37	85.20	87.17	86.39
	OPTMAX	86.38	84.29	88.76	86.62
	MAXOPT	88.22	87.06	91.66	90.66

TABLE VII
RESULTS OF G2P PREDICTION (PER NAME LANGUAGE) OF REFERENCES WITH TASK-SPECIFIC AND GENERIC RULES.

D. Variant analysis

Speaker language	Name language: A	E	Z	S
A	r → r\	r\ → r	a → A:	O → u
	a → A:	@ → a	O → u	u → O
	r →	z → s	E → i	a → A:
	a → @	{ → a	i → @	i → E
	i@ → i	Q → O	s → z	E → @
E	r → r\	r\ → r	a → @	a → A:
	a → @	@ → a	a → A:	u → O
	a → A:	{ → a	i → @	a → @
	a → {	@ → 3:	O → @u	O → @u
	x → g	E → {	E → @	E → @
Z	r →	r\ → r	A: → a	O → u
	@ → i	@ → a	a → A:	a → A:
	@ → E	@ → i	g →	E → i
	A: → a	Q → O	E → i	u → O
	@ → a	{ → a	k → g	i → E
S	r →	r\ → r	A: → a	O → u
	@ → E	@ → a	z → s	u → O
	A: → a	@ → i	E → i	E → i
	r → r\	Q → O	g → k	a → A:
	@ → i	{ → a	a → A:	A: → a

TABLE VIII
SOME OF THE TOP PHONE SUBSTITUTIONS MADE BETWEEN PRONUNCIATIONS OBSERVED AND THE AUTOMATICALLY EXTRACTED REFERENCES BASED ON THE MAXOPT METHOD FOR DIFFERENT SPEAKER LANGUAGE AND NAME LANGUAGE COMBINATIONS. PHONES ARE IN XSAMPA FORMAT, AND ARE SELECTED FROM THE LWAZI PHONE SETS [5].

Speaker language	Name language: A	E	Z	S
A	r → r\	r\ → r	a → A:	u → O
	a → @	@ → a	O → u	i → E
	{ → E	z → s	g → k	a → A:
	h →	d →	E → i	O → u
	r →	{ → a	i → @	E →
E	r → r\	d →	a → A:	u → O
	r →	@ → a	a → @	a → A:
	a → @	r\ → r	i → @	i → E
	a → A:	E → {	g → k	a → @
	a → {	→ 3:	E → @	O → @u
Z	r →	r\ → r	a → A:	u → O
	@ → E	@ → a	g →	i → E
	@ → i	@ → i	g → k	a → A:
	j → Z	Q → O	E → i	E → i
	A: → a	{ → a	K → tl_>	O → u
S	r →	r\ → r	a → A:	u → O
	@ → E	@ → a	g → k	i → E
	r → r\	@ → i	z → s	a → A:
	A: → a	Q → O	g →	A: → a
	@ → i	{ → a	E → i	h →

TABLE IX
SOME OF THE TOP PHONE SUBSTITUTIONS MADE BETWEEN PRONUNCIATIONS OBSERVED AND THE MANUALLY CORRECTED REFERENCES FOR DIFFERENT SPEAKER LANGUAGE AND NAME LANGUAGE COMBINATIONS.

In Table VIII we see the top 5 substitutions or deletions made by speakers with different mother tongue languages pronouncing words from different language origins. The insertions are not shown here as they require more context to be meaningful. The results reveal a number of interesting patterns: for example, the approximant /r\/ of English and trilled /r/ of Afrikaans are prone to deletion or interchange in all languages, the voiced/voicing feature in /z/ and /s/ is not stable, etc. It is interesting to note that these "rules" are not the same for the different language combinations, even though some commonalities do exist. In the results here, the automatic references were used to compare the pronunciations from different languages. It is also interesting to observe that when a first language speaker pronounces a name in his/her language, the G2P rules of other languages are sometimes employed, e.g. the "$r \rightarrow r\backslash$" mapping for Afrikaans speakers on Afrikaans names. Clearly, determining the correct linguistic origin of a word, is not an easy task and often ambiguous.

When the manual references are used, the results in Table IX are obtained. Here we see that the rules extracted are very similar to those from the automatic references. This is encouraging as it means that the process of variant generation may not be very sensitive to the accuracy of the references extracted. Consequently, it is possible that good variants may still be generated using the automatically extracted references.

V. CONCLUSION

It was shown that reference pronunciations can be extracted in a semi-automatic process. Although a human expert was required to verify and correct some of the entries, the process was relatively fast and efficient, and the benefit of this process will be even more pronounced when larger dictionaries are being developed.

One of our aims with this research was to determine which is most predictable: cross-lingual reference pronunciations directly, or "in-language" reference pronunciations combined with a number of P2P rules to generate additional variants. (For ASR systems it is not necessary to generate the single-best pronunciation, as long as the most commonly occurring variants can be predicted.) We found that there are numerous P2P effects that occur systematically and that these can be used to generate variants using the "in-language" reference pronunciations, which can be predicted with high accuracy.

When the name language is not taken into account, we found that the task-specific G2P rules outperformed the generic rules, suggesting that proper names pronunciations have a less regular spelling system than generic words. However, for "in-language" prediction the generic rules perform very well, suggesting that "in-language" name pronunciations are quite similar to the pronunciation of generic words. It may be that with more data the task-specific G2P rules will still outperform the generic rules. G2P systems for all languages (name and speaker languages) achieve close to 80% phoneme correctness. The only system that does not achieve this level of accuracy is English, which is not surprising given the general complexity of English G2P. If speaker language and name language overlap, reference pronunciations can be predicted with good accuracy.

Much interesting work remains to be done in order to achieve our goal of accurate pronunciation modelling of South African proper names. Most importantly, the accurate "in-

language" results achieved with generic pronunciation rules (see Table VII), along with the regularities in cross-language pronunciations (Table IX) suggest that significantly improved predictions can be obtained by combining these different knowledge sources – perhaps by using a P2P-based approach similar to that in [17]. Comparing the ASR accuracies that can be achieved with these various approaches on the Multipron corpus will also be of great practical interest.

From a linguistic perspective, it will be interesting to see whether the process of cross-language transfer of pronunciations can be characterized more generically. For example, our four languages are from two different language families; it is reasonable to expect that those family relationships will reveal themselves in the cross-lingual pronunciations. A detailed understanding of this process will be helpful in the development of algorithms that can also be applied to all those language pairs for which cross-lingual data is not available.

ACKNOWLEDGMENT

This corpus is being developed in collaboration with Jean-Pierre Martens from the University of Ghent (Belgium) and Oluwapelumi Giwa from North-West University (South Africa). Corpus development is being sponsored by the Department of Arts and Culture of the government of the Republic of South Africa; their support is gratefully acknowledged.

TERMINOLOGY

Speaker language - This is the first language of a speaker, also called the native language or mother tongue.

Name language - The language of the word's origin is referred to here as the name language or word language.

Correctness and accuracy - These are two closely related measures that can be used to evaluate the performance of a pronunciation prediction system. As the predicted pronunciation and the reference pronunciation may be of different lengths, these two pronunciations are first aligned on a phoneme-to-phoneme basis. When the two pronunciations are aligned, some of the phonemes will match (predicted correctly), others will not (prediction errors). The number of phonemes that match as a percentage of the total number of aligned phonemes is referred to as "phoneme correctness". "Phoneme accuracy" is a stricter measure whereby the total number of incorrectly inserted phonemes are subtracted from the total number of correct phonemes before the percentage is calculated. We use both measures to quantify our ability to predict different reference pronunciations.

In-language reference pronunciation - This is defined as the single pronunciation per name that is produced most often by first language speakers from the language community where the name originated. (For example, the way an isiZulu speaker would produce an isiZulu name, or an Afrikaans speaker an Afrikaans name.)

REFERENCES

[1] O. Giwa, M. H. Davel, and E. Barnard, "A Southern African corpus for multilingual name pronunciation," in *22nd Annual Symposium of the Pattern Recognition Association of South Africa (PRASA 2011)*, Nov. 2011, pp. 49–53.

[2] H. Strik and C. Cucchiarini, "Modeling pronunciation variation for ASR: A survey of the literature," *Speech Communication*, vol. 29, no. 2-4, pp. 225–246, 1999.

[3] M. Adda-Decker and L. Lamel, "Multilingual Dictionaries," in *Multilingual Speech Processing*, T. Schultz and K. Kirchoff, Eds. Berlington, MA, USA: Academic Press, 2006, ch. 5, pp. 123–166.

[4] M. Kgampe and M. H. Davel, "Consistency of cross-lingual pronunciation of South African personal names," in *21st Annual Symposium of the Pattern Recognition Association of South Africa (PRASA 2010)*, Nov. 2010, pp. 123–127.

[5] E. Barnard, M. H. Davel, and G. B. van Huyssteen, "Speech technology for information access: a South African case study," in *Proceedings of the AAAI Spring Symposium on Artificial Intelligence for Development (AI-D)*, Mar. 2010, pp. 8–13.

[6] M. H. Davel and O. Martirosian, "Pronunciation dictionary development in resource-scarce environments," in *Proceedings of the 10th Annual Conference of the International Speech Communication Association (Interspeech)*, Sep. 2009, pp. 2851–2854.

[7] B. Erol, J. Cohen, M. Etoh, H.-W. Hon, J. Luo, and J. Schalkwyk, "Mobile media search," in *ICASSP '09: Proceedings of the 2009 IEEE International Conference on Acoustics, Speech and Signal Processing*. Washington, DC, USA: IEEE Computer Society, 2009, pp. 4897–4900.

[8] F. Bechet, R. De Mori, and G. Subsol, "Very large vocabulary proper name recognition for directory assistance," in *IEEE Workshop on Automatic Speech Recognition and Understanding*, Dec. 2001, pp. 222–225.

[9] F. Bechet, R. De Mori, and G. Subsol, "Dynamic generation of proper name pronunciations for directory assistance," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 1, May 2002, pp. I–745–I–748.

[10] M. H. Davel and E. Barnard, "Pronunciation prediction with Default&Refine," *Computer Speech and Language*, vol. 22, no. 4, pp. 374–393, 2008.

[11] A. F. Llitjos and A. W. Black, "Knowledge of Language Origin Improves Pronunciation Accuracy of Proper Names," in *Eurospeech*, 2001, pp. 1919–1922.

[12] H. van den Heuvel, J.-P. Martens, K. D'hanens, and N. Konings, "The Autonomata Spoken Names Corpus," in *Proceedings LREC*, 2008, pp. 140–143.

[13] H. van den Heuvel, B. Réveil, and J.-P. Martens, "Pronunciation-based ASR for names," in *Interspeech 2009: 10th Annual Conference of the International Speech Communication Association, Vols. 1-5*, 2009, pp. 2959–2962.

[14] B. Réveil, J.-P. Martens, and B. D'Hoore, "How speaker tongue and name source language affect the automatic recognition of spoken names," in *Interspeech 2009: 10th Annual Conference of the International Speech Communication Association, Vols. 1-5*, 2009, pp. 2971–2974.

[15] B. Réveil, J.-P. Martens, and H. van den Heuvel, "Improving proper name recognition by adding automatically learned pronunciation variants to the lexicon," in *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, N. Calzolari, K. Choukri, B. Maegaard, J. Mariani, J. Odijk, S. Piperidis, M. Rosner, and D. Tapias, Eds., 2010, pp. 2149–2154.

[16] B. Réveil, J.-P. Martens, and H. van den Heuvel, "Improving proper name recognition by means of automatically learned pronunciation variants," *Speech Communication*, vol. 54, no. 3, pp. 321–340, 2012.

[17] Q. Yang, J.-P. Martens, N. Konings, and H. van den Heuvel, "Development of a phoneme-to-phoneme (p2p) converter to improve the grapheme-to-phoneme (g2p) conversion of names," in *Proceedings LREC*, 2006, pp. 287–292.

[18] H. van den Heuvel, J.-P. Martens, and N. Konings, "G2P conversion of names : what can we do (better)?" in *Interspeech 2007: 8th Annual Conference of the International Speech Communication Association*, vol. 1-4, 2007, pp. 1181–1184.

[19] F. Stouten and J. Martens, "Dealing with cross-lingual aspects in spoken name recognition," in *IEEE Workshop on Automatic Speech Recognition and Understading, Vols. 1-2*, 2007, pp. 419–424.

[20] M. H. Davel, C. J. van Heerden, and E. Barnard, "Validating smartphone-collected speech corpora (accepted for publication)," in *Proc. Spoken Language Technologies for Under-resourced Languages (SLTU)*, May 2012.

[21] Meraka-Institute, "Lwazi ASR corpus," http://www.meraka.org.za/lwazi, 2009.

Grid Smoothing Based Image Compression

Jenny Bashala
Electrical Engineering
French South African Institute, Tshwane University of technology Private Bag X680, Pretoria 0001, South Africa
Email: jennybashala@gmail.com

Karim Djouani
Electrical Engineering
French South African Institute, Tshwane University of Technology Private bag X680, Pretoria 0001, South Africa
Email: djouani@ieee.org

Yskandar Hamam
Electrical Engineering
French South African Institute, Tshwane University of Technology Private Bag X680, Pretoria 0001, South Africa
Email : hamama@tut.ac.za

Guillaume Noel
Setsebi Consulting, Bagnols, Ceze, 30200, France
Email: noel_gpa@yahoo.com

***Abstract*—The lossy image compression method described in this paper uses a graph-based approach to reduce the image size. The presented method is based on the assumption that any image may be decomposed into a structure and detailed components. The detail part is compressed with a frequency-based scheme (transform coding used in JPEG and JPEG2000 for example) while the structure component is processed with a grid smoothing assisted by a graph decimation technique. The performance of the compression method is demonstrated on few popular images.**

Keywords—Bilateral Mesh filtering, Grid smoothing, Mesh decimation

I. INTRODUCTION

Digital images usually contain a large amount of data. The facility to save, transmit and retrieve digital images efficiently becomes more and more important in this cutting edge technology. In today's world, where exchange of images is part of our daily life, everyone has experienced the benefit of reducing the size of a file containing images. The existing image compression techniques reduce the number of bits representing the image by exploiting the redundancies in the original image while preserving the resolution and the visual quality of the reconstructed image as close to the original image as possible. The compression method can be either lossy or lossless. The well-known lossy compression methods make use of transform coding, vector quantization, image compression by linear splines over adaptive triangulation, fractals, or subband wavelet coding schemes for removing psychovisual and statistical image redundancies [5]. However, as the bit rate is decreased and the compression ratio increased, each compression technique introduces artifact, creating blocky, blurry, patchy or smudgy images [5]. Most of these methods operate on pixels values of the original image and only few methods operate on the graph of the image to reduce its size.

The main idea of our compression technique is to capitalize on the advantages of the pixel-based and graph-based methods. The algorithm uses bilateral mesh filtering to split the input image into structure and detail components. The structure component is the resulting filtered image which contains the large scale features while the detailed component corresponds to the residual image obtained by subtracting the image structure from the input image. In figure 1, it is shown that the grid smoothing is applied on the filtered image I_S in order to extract the non-uniform grid reflecting the image structure. The structure of an image I can be seen as a set of points in which the first two coordinates represent the row x and the column y determining the position (x, y) of a pixel. The third coordinate corresponds to the pixel value $I(x, y)$ at the given position. The neighborhood of a pixel contains either four or eight pixels. Four pixels create four connectivity while eight pixels create eight connectivity. The set of points and the connectivity associated to the image helps to associate an image with a graph. The image is seen as a collection of vertices or nodes where a vertex represents a pixel. The edges are represented by the connectivity of the neighborhood pixels. Uniformly distributed position coordinates (x, y) leads to a uniform mesh or uniform grid. Meshes or graphs with non-uniformly distributed coordinates (x, y) will be named non-uniform grids or meshes. During the grid smoothing process, vertices are moved from small variances regions to large variance regions since the regions with small variance require fewer points than the regions with large variance [9]. The output of the grid smoothing contains a set of coordinates combined together to form the non-uniform grid. Delaunay triangulation is performed on the set of coordinate's points to generate triangular faces. The resulting triangular mesh is decimated through mesh simplification process. The simplification lies in eliminating elements of the mesh such as vertices, edges and faces [4, 2]. The simplification exploited is the mesh decimation [11]. The decimation process removes vertices and faces from a mesh. Since we are working on a triangular mesh, the mesh decimation will reduce the number of triangles (faces) in the mesh without losing the overall structure. The number of vertices of the simplified mesh corresponds to number of pixels of the compressed image. The reconstruction process is based on

mapping the color values associated to the each vertex of the simplified mesh. In our case, we map the associated gray level of each vertex (pixel) by interpolation since we are working in gray scale.

The lines below of this paper will give more details on the components used to implement our lossy image compression algorithm. Section 2 gives the notion of bilateral mesh filtering and grid smoothing in image processing. Section 3 describes the use of mesh simplification to reduce the size of an image. Section 4 illustrates the proposed lossy image compression method. Section 5 shows the results. A conclusion is given in section 6.

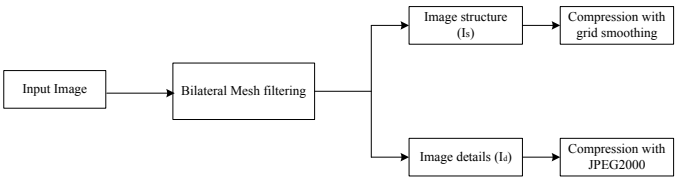


Figure 1. Image Preprocessing

II. BILATERAL MESH FILTERING AND GRID SMOOTHING

A. Bilateral Mesh Filtering

Bilateral mesh filtering corresponds to a bilateral filter implemented using graph-based approach. It imitates the behavior of the classical bilateral and mesh filtering; whilst presenting some properties of mesh smoothing [10]. The graph used in the bilateral mesh filtering process consists of a set of vertices that are correlated with the image pixels values. The link between vertices is identified as edges characterizing the relationship between pixels. This new filtering is implemented via an energy function based on the mesh smoothing model of Hamam and Couprie. The cost function is developed as a graph and minimized. This function is expressed as a sum of data fidelity and smoothing terms based on the node-edge incidence [10]. The filter defines a weight based on the difference in grayscale of the extremities of the connection and makes use of an exponential law. It takes into account the luminance proximity and computes the distance between the luminance of two vertices z_i and z_j as in [10]:

$$d_{i,j}^{l} = \exp\left(-\frac{\left\| z_i - z_j \right\|_2^2}{\sigma} \right) \quad (1)$$

With: - $d_{i,j}^{l}$: represents the distance between the vertices i and j .

- $\left\| z_i - z_j \right\|_2$ represents the L_2 norm between the grey levels.
- σ represents the variance parameter of the Gaussian distribution.

The objective function of the first order bilateral mesh filter is defined as:

$$J_Z = \left(Z - \hat{Z}\right)^t \left(Z - \hat{Z}\right) + \theta Z^t C^t \Omega C Z \quad (2)$$

The optimal solution of the first order is given by:

$$Z = \left(I + \theta C^t \Omega C\right)^{-1} \hat{Z} \quad (3)$$

The optimal solution of the second order is given by

$$Z = \left(I + \theta C^t \Omega C C^t \Omega C\right)^{-1} \hat{Z} \quad (4)$$

The diagonal square matrix $\Omega = diag\left(w_1, ..., w_L\right)$ of size $L \times L$ (L: number of connections in the graph) has its w_l diagonal elements defined by:

$$w_l = \exp\left(-\frac{\left\| z_i^0 - z_j^0 \right\|_2^2}{\sigma} \right) \quad (5)$$

Where i is the sending end of the connection l and j is the receiving end. z_α^0 represents the initial grey level of node α .

The model of the bilateral mesh filtering is defined from equation (1) to (5). From these expressions, it is understood that the performance of the new filter depends on the parameters θ and σ which corresponds to σ_d and σ_r respectively when compared the classical bilateral filter [12].

(a)Original Image (b)Image structure (c)Image details

Figure 2. Result of Bilateral Mesh filtering

B. Grid Smoothing

The grid smoothing is a new graph-based technique for image processing and analysis developed by Guillaume Noel, Karim Djouani, and Yskandar Hamam. This technique presents a general outline analogous to the mesh smoothing in which a cost function is defined and optimized. The method is interpreted as projection of the grey levels of the input image onto the sampling grid; and enhances the edges of the input image while preserving the number of nodes. The Grid smoothing operates on the theory where regions with small variance necessitate fewer points than regions with a large variance. Points with small variance regions are moved to large variance regions. The grid smoothing method changes the coordinates of the points in the grid to match the entities in the image. This graph based technique is formulated as an optimization problem defined in [9] as:

$$\min_{(X,Y)\in R^K \times R^K} J(X,Y) \quad (6)$$

Where $J(X,Y)$ represents the cost function of variables $(X,Y)\in R^K \times R^K$. (X,Y) represents the coordinates of nodes in the mesh, and *K* represents the number of pixels images.

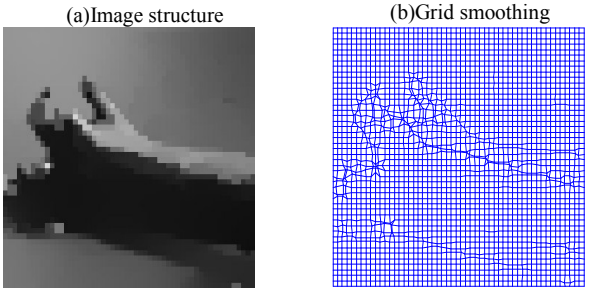


Figure 3. Grid smoothing of a portion of the image structure

III. MESH SIMPLIFICATION TECHNIQUES

Mesh simplification consists of eliminating the elements of a mesh (vertices, edges or faces) while preserving the original shape and appearance [3]. Several mesh simplification algorithms have been developed [2]. Most algorithms reduce the complexity of the mesh by merging elements of the mesh, by resampling the vertices [1, 2]. Depending on the desired output mesh, some algorithms preserve the input mesh while others alter it illogically [4].

One category of mesh simplification referred to as mesh decimation simplifies meshes by removing vertices and faces from a mesh [11]. The main idea is to reduce the number of faces in the mesh by iterative vertex decimation, edge collapse or contraction without losing the overall structure. . Most faces are triangles. The iteration process is terminated when the required percentage of reduction of the mesh is reached or when some decimation criteria are reached. Most mesh decimation approaches are based on iterative edge collapse or edge contraction [8]. An edge collapse is an operation that reduces an edge into a single vertex. When this is done all edges and faces connected to the removed vertices has to be reconnected to the new vertex. Several theories have been developed on how to efficiently collapse edges while preserving the original topology and a good approximation to the original geometry. Some techniques have been more complex than others. The essential difference between these techniques lies in how they choose an edge to contract.

One of the well-known techniques of mesh decimation is the Surface Simplification Using quadratic error metrics developed by Garland and Heckbert. The base operation of their technique is the edge collapse where an edge is reduced into a single vertex by merging the two vertices of the edge. The contraction of the pairs is performed by repositioning the two vertices to a new selected location. The change in vertices location results in deletion of vertices, while all the edges and faces connected to the removed vertices are reconnected to the new vertex. This process might degenerates few faces or edges which will be removed from the mesh. The approximation produced by the algorithm maintains high fidelity to the original mesh [6]. The algorithm of Surface simplification using quadratics error metrics of Garland and Heckbert is implemented based on the norm stating that the validity of the vertex pair (v_1, v_2) chosen for contraction focus on either:

- (v_1, v_2) is an edge or
- $\|v_1 - v_2\| < t$, where t is a threshold parameter.

The choice of the contraction is based on the cost function of contraction. The characteristic of the error at each vertex helps to define the contraction cost. Garland and Heckbert defined the error at a vertex $v = [v_x \quad v_y \quad v_z \quad 1]^T$ using the quadratic form:

$$\Delta(v) = v^T Q v = q_{11}x^2 + 2q_{12}xy + 2q_{13}xz + 2q_{14}z + q_{22}y^2 + 2q_{23}yz + 2q_{24}y + q_{33}z^2 + 2q_{34}z + q_{44} \quad (7)$$

Where Q is a symmetric 4x4 matrix associated with each vertex. A new matrix $\overline{Q}$ must be derived at each vertex pair contraction to approximate the error at new vertex $\bar{v}$. The new matrix $\overline{Q}$ is defined as:

$$\overline{Q} = Q_1 + Q_2 \quad (8)$$

The contracted vertex pair (v_1, v_2) is placed at either v_1, v_2 or $(v_1 + v_2)/2$ depending on the lowest value of the error $\Delta(\bar{v})$ produced by either of the selected location. The ideal location would the one that minimize $\Delta(\bar{v})$. The minimum is found by solving for $\bar{v}$ (homogenous vector):

$$\frac{\partial \Delta}{\partial x} = \frac{\partial \Delta}{\partial y} = \frac{\partial \Delta}{\partial z} = 0 \text{ Or}$$

$$\begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{21} & q_{22} & q_{23} & q_{24} \\ q_{31} & q_{32} & q_{33} & q_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \bar{v} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \quad (9)$$

With

$$\bar{v} = \begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{21} & q_{22} & q_{23} & q_{24} \\ q_{31} & q_{32} & q_{33} & q_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

The performance of decimation method developed by Garland and Heckbert is similar to a MATLAB function reducepatch. The operation of the function consists in reducing the number of faces of the triangular mesh while preserving the overall shape of the original mesh. For details on the reducepatch function see MathWorks.com.

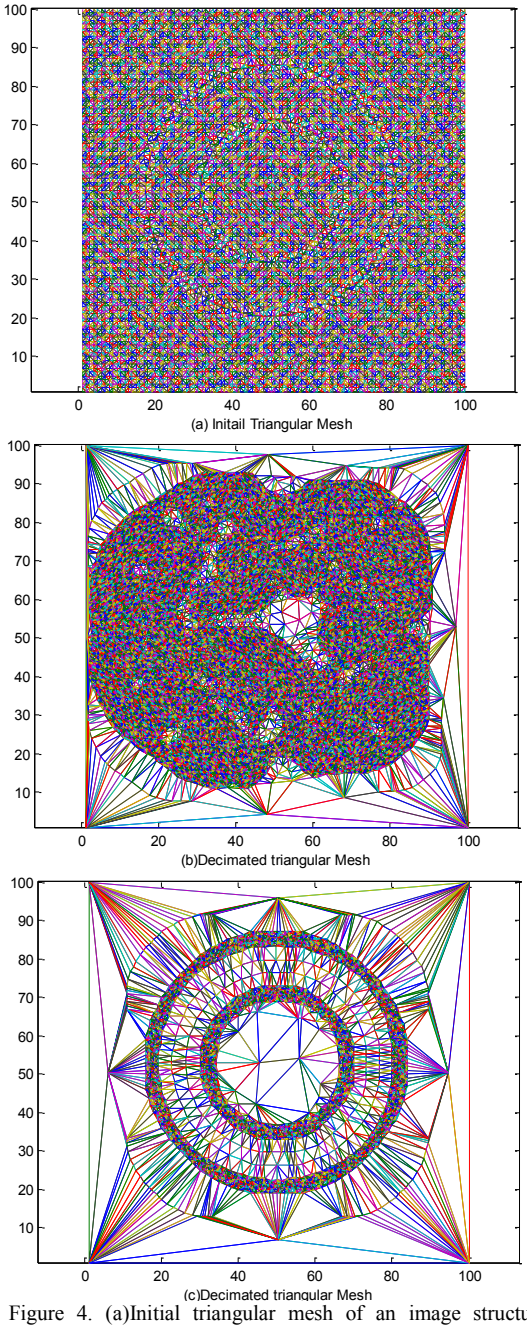


Figure 4. (a)Initial triangular mesh of an image structure; (b)Mesh decimated to 50%; (c) Mesh decimated to 10%

IV. PROPOSED COMPRESSION METHOD

The proposed lossy compression scheme concentrates on the data reduction of the image structure using the grid smoothing to extract the image structure graph. The vertices of resulting image graph are rearranged using Delaunay triangulation to create triangular faces. The resultant mesh with triangular faces is decimated using a triangulated mesh simplification technique. The resulting decimated mesh is used to retrieve the vertices coordinates' and convert the set of coordinates to a matrix of pixels. The number of vertices equals the number of pixels in the image. The reconstruction process is based on mapping the gray values associated to the vertices of the decimated mesh into a set of gray values associated to a uniform grid. Each vertex is associated with a gray level indicating the color of the vertex. The objective of the reconstruction is to allocate gray levels to the pixels. The approach used for the reconstruction is the triangle based interpolation of the gray levels and the resampling of the interpolated surface.

V. RESULTS

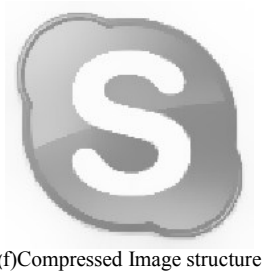
(a)Image structure

(b) Compressed Image structure with 40 % of the mesh decimated (PSNR= 41.9303 dB)

(c)Image structure

(d)Compressed Image structure using Grid smoothing by 50% of mesh decimation PSNR = 29.6049 dB

(e)Image structure



(f)Compressed Image structure using Grid smoothing by 20 % of mesh decimation PSNR = 38.7868 dB

(g)Image structure

(h)Compressed Image structure by 60% of mesh decimation Grid smoothing (PSNR = 39.2764 dB)

Figure 5. Simulation results

VI. CONCLUSION

The lossy image compression scheme presented in this paper proposes a new graph-based approach to compress images. It shows the efficiency of graph-based approach in image compression. The reconstructed image displays a good visual quality with a good peak signal to noise ratio which makes this new technique an alternative lossy image compression scheme. The developed method is centered on image data reduction. A study has to be done on the encoding of the reduced image data.

REFERENCES

[1] Chen, H., Yin, G., & Zhang, J. (2008). A real time mesh simplification algorithm based on half-edge collapse. *Control and Decision Conference, 2008. CCDC2008* (pp. 1896-1899). Chinese: IEEE.

[2] Cignoni, P., Montani, C., & Scopigno, a. R. (1997). A comparison of Mesh Simplification Algorithms. *Computers & Graphics*, 37-54.

[3] Cohen, J., Olano, M., & Manocha, a. D. (1998). Appearance preserving simplification. *SIGGRAPH'98 Proceedings of the 25th annual conference on Computer graphics and interactive techniques* (pp. 115-122). New York: ACM New York.

[4] Erikson, C. (1996). *Polygonal Simplification: An overview.* UNC Chapel Hill Computer Science.

[5] Eskicioglu, A. (2000). Quality measurement for monochrome compressed images in the past 25 years. *Acoustics, Speech, and Signal Processing, 2000, ICASSP'00, Proceedings, 2000 IEEE International Conference* (pp. 1907-1910 vol.4). IEEE.

[6] Garland, M., & Heckbert, P. S. (1997). Surface simplification using quadratic error metric. *SIGGRAPH'97 Proceedings of the 24th annual conference on Computer graphics and interactive techniques* (pp. 209-216). New York: ACM Press/Addison-Wesley Publishing Co.

[7] Hoppe, H. (1996). Progressive Meshes. *SIGGRAPH'96 Proceedings of the annual conference on computer graphics and interactive techniques* (pp. 99-108). New York: ACM New York.

[8] Hoppe, H., DeRose, T., Duchamp, T., McDonal, J., & Stuetzle, a. W. (1993). Mesh optimization. *SIGGRAPH'93 Proceedings of the 20th annual conference on computer graphics and interactive techniques* (pp. 19-26). New York: ACM New York.

[9] Noel, G., Djouani, K., & Hamam, a. Y. (2011). Graph-based Image Sharpening Filter Applied to Image Denoising. *International Journal of Smart Home*, Vol.5, No.2.

[10] Noel, G., Djouani, K., Wyk, B. V., & Hamam, a. Y. (2012, July 1). Bilateral Mesh filtering. *Pattern Recognition Letters*, pp. 1101-1107.

[11] Schroeder, W. J., Zarge, J. A., & Lorensen, a. W. (1992). Decimation of triangle meshes. *SIGGRAPH'92 Proceedings of the 19th annual conference on Computer graphics and interactive techniques* (pp. 65-70). New York: ACM SIGGRAPH Computer Graphics.

[12] Tomasi, C., & Manduchi, R. (1998). Bilateral Filtering for gray and color images. *Computer Vision, 1998, Sixth International Conference* (pp. 839-846). IEEE.

Developing and improving a statistical machine translation system for English to Setswana: a linguistically-motivated approach

Ilana Wilken
North West University
Potchefstroom, South Africa
ilanawilken@gmail.com

Marissa Griesel and Cindy McKellar
CTexT©
North West University
Potchefstroom, South Africa
{Marissa.Griesel; Cindy.McKellar}@nwu.ac.za

***Abstract* — This paper describes the methods that were followed in the development and improvement of a statistical machine translation system for translation from English to Setswana. Setswana is regarded as a resource scarce language and therefore an adequate amount of parallel data is not freely available. The methods created attempt to improve the quality of a machine translation by manipulating the data during processing. The methods include the creation of sentence reordering, term deletion and term replacement rules. The rules were applied to training and testing data in the pre- and post-processing stages of development. The systems were compared to one another to detect whether the quality of the machine translation improved.**

***Keywords*—**statistical machine translation, pre-processing, post-processing, sentence reordering, English, Setswana, term replacement, term deletion

I. Introduction

South Africa is a diverse, multi-lingual country and has eleven official languages [1]. According to the South African Bill of Rights [2], "everyone has the right to use the language…of their choice" as well as "the right of access to any information held by the state."

The South African government strives to provide information in all of the languages, but according to Prinsloo & De Schryver [3], corpora (and even more so parallel corpora) for all eleven official languages of South Africa is not always obtainable. Statistical machine translation (SMT) systems could serve as an additional tool for human translators to simplify, standardise, and expedite the translation process in the South African context.

For this research, it was decided to develop a SMT system for the translation of English to Setswana. Setswana falls into the Southeastern Bantu language group [4] and this research will contribute to the advancement of other closely related languages. These languages include Sesotho and languages in the Nguni, Tsivenda, and Xitsonga groups.

The development of the SMT system was done in two stages: first, a baseline system was developed. A text was translated and results were obtained. For the second stage, six adapted systems were developed. The adapted systems are an extended, a reordering, a replacement, a deletion, a deletion-replacement, and a deletion-reordering system. During the development of the adapted systems, linguistically motived rules were written and applied to the data. A text for each system was translated and individual results were obtained. The results of all the systems were compared to establish if an improvement of the quality of the translation took place.

The rest of this paper is organised as follows: Section 2 describes related work and Section 3 describes the development of the rules as well as the training of the systems. Section 4 gives details on the development of the adapted systems as a whole. Section 5 explains the evaluation of the systems as well as how the quality of the output of the systems improved. The conclusion and an overview of future work can be found at the end of the paper in Section 6.

II. Related Work

Truly automated machine translation of complex text cannot deliver output of the quality human translators would achieve. This project aims at improving the workflow and quality of language services in the government sector. Machine-aided human translation was therefore recognized as a means to achieve this aim. Machine-aided human translation can be explained as a draft translation initially done by a computer, but a human translator still remains responsible for correcting any errors. Such systems have already been developed for numerous international studies as well as for South African language pairs.

A machine translation (MT) system employing a pre-processing step is the English to Swahili, Swahili to English machine translation system [5]. The SAWA Corpus Project developed an English-Swahili parallel corpus and then built a SMT system for the application of the corpus. Swahili is a

strong agglutinative language and so words were first morphologically deconstructed to facilitate the connection between the morphemes and their corresponding English words. This improved the automatic word alignments drawn during the training phase. The very basic SMT system's results were compared to those of the Google Translate [6] system's results for Swahili. The results showed that the SAWA system disappointed in comparison to the Google Translate System for the translation from English to Swahili. The BLEU [7] and NIST [8] scores declined by 0.06 and 1.04 respectively. However, for the Swahili to English system, the SAWA system fared much better and showed improvements in the BLEU and NIST scores, increasing by 0.06 and 0.38 respectively.

A recent high scale machine translation project for South African languages was undertaken in the Autshumato project [9]. Smaller (research) experiments were previously conducted for SA languages, but this was the first project to develop an integrated strategy for the government domain. The project concentrated on translation from English to Afrikaans, Sesotho sa Leboa and isiZulu and a pre-processing method of syntactic reordering of the source language was used to improve on the results of the baseline systems [10]. The experiments showed positive results, resulting in improvements of the BLEU and NIST scores. The English-Afrikaans system's NIST score improved by 0.0274, whereas the English-Sesotho sa Leboa system's BLEU and NIST scores improved by 0.0406 and 0.5321, respectively.

SMT systems require great amounts of data, but large English-Setswana parallel data does not exist, because Setswana is considered a low resource language[1]. Simply gathering more bilingual data is not a practical option when developing SMT systems and so other methods to improve the quality of machine translation output is essential. The Autshumato project set a benchmark for machine translation for South African languages. Accordingly, the purpose of this research is to serve as an extension of the Autshumato project. For the English to Setswana SMT system developed in this research, it was decided to attempt improving the translation quality of the baseline system by applying pre- and post-processing steps. The steps include sentence reordering, as well as linguistically motivated deletion and replacement rules.

III. Rule Development

The data sets used for the development of the linguistic rules consist of 200 randomly selected sentences of each language and was taken from the training data mentioned above. The English data set was first translated with the baseline system to identify areas suitable for potential improvement. By comparing the English and Setswana data sets, it was noted that certain words exist only in one language and not in the other. The word order of the sentences did not align either. The original English data was then annotated with part-of-speech tags and by applying extensive linguistic knowledge [11], the rules were developed.

The numbers of core technologies to draw from are limited for Setswana and therefore limit the amount of processing that we are able to perform on the target language. However, numerous core technologies exist for English and it was decided that merely a part-of-speech tagger for English would be adequate for this project. The Stanford Log-linear Part-of-Speech Tagger (Stanford PoS Tagger) [12] was used to annotate the English training data and the development data set. This tagger was chosen because of the output data's usable quality. All of the reordering, deletion and replacement of words was done based on these tags. The rules were created and implemented using Perl [13] regular expressions.

The reordering and deletion rules are similar to those used by the Autshumato system for English-Sesotho sa Leboa. This is possible because both Setswana and Sesotho sa Leboa belong to the syntactically similar Sotho language family group [4]. However, a different approach was followed in the implementation thereof.

For this project, the rules were implemented individually, as well as in groups of rule sets. The deletion, replacement and reordering rules were implemented each on their own and so formed three of the six systems. These three systems are the deletion, replacement, and reordering systems. The deletion and replacement rules were grouped together, forming the deletion-replacement system; and the deletion and reordering rules were grouped together to form the deletion-reordering system. All the rules were then grouped together to form the extended system.

1. Deletion Rules

The deletion rules remove English words for which no Setswana equivalent exists. There are only three deletion rules and all three rules affect specific determiners in English. The determiners affected are *the*, *an* and *a*.

2. Replacement Rules

The purpose of the replacement rules is to ensure that the English conjunction word is translated with the correct Setswana conjunction word. In Setswana, the conjunction of nouns and the conjunction of verbs differ. When nouns are joined, *and* is translated as *le*, but when verbs are joined, *and* is translated as *mme*. Other conjunctions that are translated with the correct Setswana word is *or, but* and *because*. They are respectively replaced with *kgotsa*, *mme* and *ka gore*.

3. Reordering Rules

The reordering rules address the differences in the word order between English and Setswana. In Setswana, nouns are written first, followed by adjectives, and/or pronouns, and/or cardinal

[1] In 2005 the Pretoria Setswana Corpus consisted of 6 130 557 words, whereas the Pretoria English Corpus consisted of 12 799 623 words [3].

numbers, and/or specific determiners. The reordering rules change the order of the English words.

When the replacement and reordering rules are implemented on their own, determiners must be taken into consideration and the rules must be able to detect determiners when they are not deleted by the deletion rule. The rules can be explained as follows: the sequences of certain words are written in square brackets. *Possible:* means that an adjective, an adverb, or a determiner will be detected, but it will not matter if no adjective, adverb or determiner is present. When a word in a rule is written in bold in square brackets (for example [**or**]), it means that the word must be present for the rule to be applied.

The three basic rule groups were each applied separately and are set out below. An example sentence of the implementation of the rule is also given. The first sentence is the original sentence, as found in the baseline system, followed by the adapted sentence for that particular system.

A. Deletion System

- [determiner: **the** or **an** or **a**] → delete [determiner: **the** or **an** or **a**]

Example:

the status of a person as an only member of a state

→ **status of person as only member of state**

B. Replacement System

- [noun] [**and**] [possible: the or an or a] [possible: adjective] [noun] → translate *and* with *le*
- [verb] [**and**] [possible: the or an or a] [possible: adverb] [verb] → translate *and* with *mme*
- [conjunction **or**] → translate *or* with *kgotsa*
- [conjunction **but**] → translate *but* with *mme*
- [conjunction **because**] → translate *because* with *ka gore*

Example:

we have limited opportunities because we have limited resources and help from volunteers

→ **we have limited opportunities ka gore we have limited resources le help from volunteers**

C. Reordering System

- [specific determiner] [possible: *the* or *an* or *a*] [possible: adjective] [noun] → [possible: *the* or *an* or *a*] [possible: adjective] [noun] [specific determiner]
- [cardinal number] [**to**] [cardinal number] [possible: adjective] [noun] → [possible: adjective] [noun] [cardinal number] [**to**] [cardinal number]
- [cardinal number] [possible: adjective] [noun] → [possible: adjective] [noun] [cardinal number]
- [pronoun] [possible: adjective] [noun] → [possible: adjective] [noun] [pronoun]
- [adjective] [**and**] [adjective] [noun] → [noun] [adjective] [**and**] [adjective]
- [adjective] [noun] → [noun] [adjective]

Example:

the unacceptable misapplication of government power

→ **the misapplication unacceptable of government power**

The basic rules were also combined in three different systems to optimize the rule ordering. The rules for the deletion-replacement and deletion-reordering systems are similar to the separate rules explained above, but for these rules the determiners *the*, *an* and *a* do not need to be detected, since they are deleted before the next steps are reached. The combination of the rules and the changes affecting the rules are listed below:

D. Deletion-Replacement System

- [determiner: **the** or **an** or **a**] →delete [determiner: **the** or **an** or **a**]
- [noun] [**and**] [possible: the or an or a] [possible: adjective] [noun] →translate *and* with *le*
- [verb] [**and**] [possible: the or an or a] [possible: adverb] [verb] →translate *and* with *mme*
- [conjunction **or**] →translate *or* with *kgotsa*
- [conjunction **but**] →translate *but* with *mme*
- [conjunction **because**] →translate *because* with *ka gore*

Example:

having to report and explain to a higher authority

→ **having to report mme explain to higher authority**

E. Deletion-Reordering System

- [determiner: **the** or **an** or **a**] →delete [determiner: **the** or **an** or **a**]
- [specific determiner] [possible: *the* or *an* or *a*] [possible: adjective] [noun] → [possible: *the* or *an* or *a*] [possible: adjective] [noun] [specific determiner]
- [cardinal number] [**to**] [cardinal number] [possible: adjective] [noun] → [possible: adjective] [noun] [cardinal number] [**to**] [cardinal number]
- [cardinal number] [possible: adjective] [noun] → [possible: adjective] [noun] [cardinal number]

- [pronoun] [possible: *the* or *an* or *a*] [possible: adjective] [noun] ⟶ [possible: *the* or *an* or *a*] [possible: adjective] [noun] [pronoun]
- [adjective] [**and**] [adjective] [noun] ⟶ [noun] [adjective] [**and**] [adjective]
- [adjective] [noun] ⟶ [noun] [adjective]

Example:
compulsory enlistment in the armed forces
→ **enlistment compulsory in forces armed**

F. Extended System

All of the abovementioned rules were applied during the training of the extended system. The order of the rules is as follows:

- Deletion rules
- Replacement rules
- Reordering rules

Example:
a branch or subdivision of the public service and the relationship between the state and its citizens
→ **branch kgotsa subdivision of service public mme relationship between state le citizens its**

IV. Training of the Systems

The training data used in this research project consist of a parallel corpus[2] of English-Setswana sentence pairs and a monolingual Setswana corpus for language modelling. The corpora contain data from the South African government domain. The parallel corpus was automatically aligned with an algorithm developed by Robert Moore [14]. The open source statistical machine translation toolkit, Moses [15], was used for the training of both the baseline and adapted systems and the SRILM toolkit [16] was used to train the language model. Table 1 indicates the quantity of data used.

TABLE I. Data Quantity

Corpus	**Number of sentences / -pairs**
Parallel Corpus	34 321 English-Setswana sentence pairs
Monolingual Corpus	50 923 Setswana sentences

[2] For more information on these corpora, please contact CTexT® [12]

V. The Development of the Adapted Systems

The different systems were created to evaluate which linguistic rule – whether on its own or grouped together – provided the best translation quality of a translated text. An example of how the adapted systems were developed can be seen in Fig. 1. This example shows the development of the extended system, where all the rules are grouped together. For the other systems, the applied rules are adapted to suit each system.

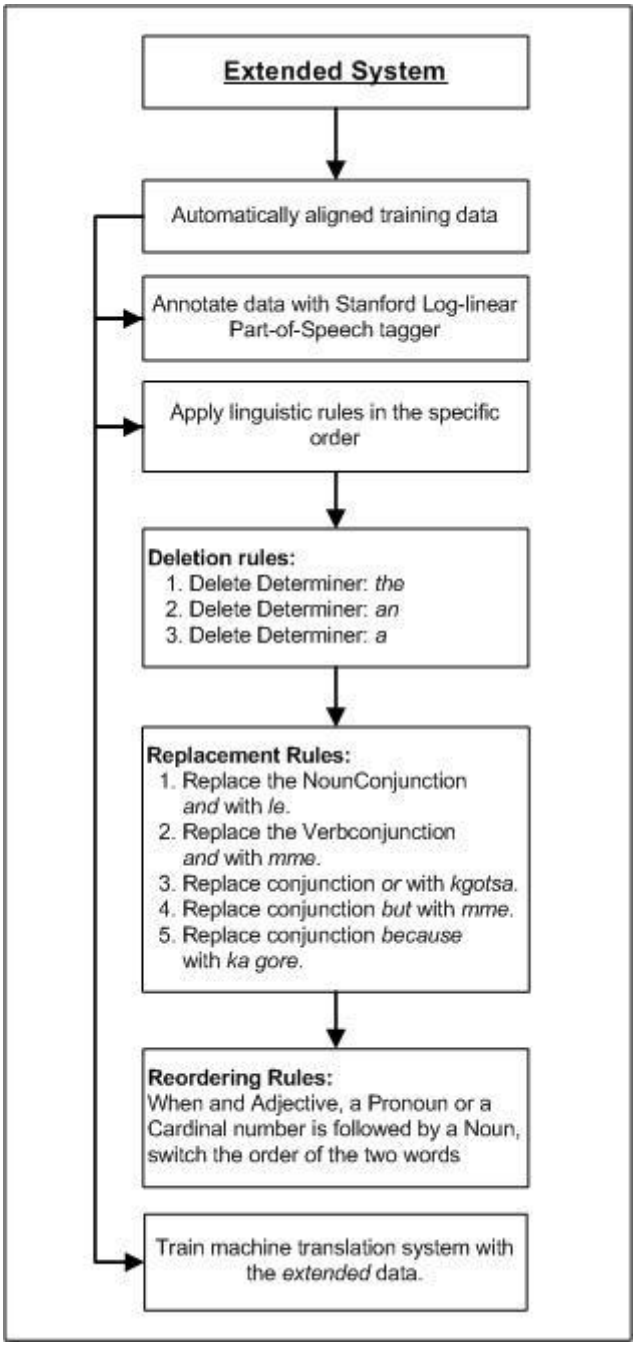


Fig. 1. Development of an adapted system

All the linguistic rules were applied during pre-processing. However, post-processing of the *the*-deletion rule was necessary for the deletion, deletion-reordering, deletion-replacement and extended systems. The reason for this is that the Setswana data contain English words and phrases. Since we have such a small amount of data and because we do not have the means to cleanse the data manually, we decided not to dispose of the sentences containing English words and phrases.

The data containing English words poses a big problem, because when the language model is trained, the English words in the Setswana data are seen as Setswana words. They are therefore included in the Setswana language model and when an English text is then translated, an English word will be translated with a 'Setswana' word, when the word is in fact also an English word.

When the testing data was translated, the *the*-determiner was detected in the translated text. Post-processing was the preferred method of choice because it is quick and effective in the removal of wrongly inserted/translated English words. The removal of these words could also have a positive impact on the word-level evaluations done later.

VI. Evaluation

Testing data consisted of 500 manually aligned English-Setswana sentence pairs. The alignments were done using the CTexT® Alignment Interface [9]. This data set is from the same domain as the training data; however, none of these 500 sentence pairs appear in the training data.

A baseline machine translation system was trained and the original testing data was translated. All six adapted systems received the same testing data set, but the manner in which the linguistic rules were applied differed, as explained in section V. The adapted testing data sets were translated and results were obtained. The results of all six adapted systems were compared to the results of the baseline system.

The output quality of a machine translation system can be evaluated in two ways: human evaluation or automatic evaluation [17]. Only the automatic evaluation was used for this project, because it is a sufficient way of obtaining reliable results immediately. These results determine whether or not the quality of the translation generated by the SMT system improved.

The BLEU and NIST scores were calculated for each machine translation system. A BLEU score measures the closeness between a machine translation and a reference translation. The quality of the machine translation is determined by how identically similar it is to the professional human translation. The BLEU evaluation is done according to a numerical metric, ranging from 0 to 1. When a score of 1 is reached, it means that the translated text is as identically similar to the reference translation as possible.

The machine translation output and the reference translations are compared in terms of the statistics of short sequences of words, also known as word n-grams. The NIST score calculates how informative a particular n-gram is. The translation quality is judged to be at its best when a translation shares as many n-grams as possible with the reference translation. Table 1 indicates the BLEU and NIST scores for the baseline and extended machine translation systems. The BLEU scores are also represented as percentages, as well as the difference between the baseline system and the other systems indicated in brackets.

For each of the adapted systems, the results showed gains in both the BLEU and NIST scores when compared to the results of the baseline system. The biggest gain was for the extended system, where both the BLEU and the NIST scores improved by 0.0136 (1.36%) and 0.0896 respectively.

The system that showed the least improvement of both the BLEU and NIST scores is the replacement system. The reason is that no word reordering or replacement took place – one word is merely translated with another, which might be correctly translated by the system from the start.

A translation is evaluated on different levels when evaluation takes place. They are: an overall scoring, an individual n-gram scoring, and a cumulative n-gram scoring level. The overall scoring level presents the BLEU and NIST scores used to determine if an improvement in a system's translation quality took place. The individual n-gram level does comparisons of the translated and reference data on an isolated n-gram level. The levels range from 1-gram to 9-gram, meaning one word is compared to one word, two words to two words, and so forth until nine words are compared to nine words. The cumulative n-gram level does comparisons of groups of words that occur in the translated and reference data. These levels also range from 1-gram to 9-gram, but now the first word is compared to the first word of the translated text, the first two words of both texts are compared, and so it continues until a group of nine words are compared to a group of nine words of both texts.

When the NIST scores are compared for the individual n-gram scoring level, the Extended system fares the best of all the systems evaluated, showing improvements from 1-gram through to 8-gram. On the other hand, the replacement and deletion-replacement systems indicated the second and third highest improvements of the NIST scores of the individual n-gram scoring level. Both their improvements range from 1-gram through to 7-gram. It would therefore be safe to say that when results seem to be unimportant at first, there could be prospective improvements on a smaller scale. These small-scale improvements might be equally as useful as the overall results.

TABLE II. EVALUATION RESULTS

System	BLEU	BLEU %	NIST
Baseline	0.2744	27.44%	6.0911
Extended	0.2880	28.80% (+1.36%)	6.1807
Reordering	0.2781	27.81% (+0.37%)	6.1155
Replacement	0.2751	27.51% (+0.07%)	6.1049
Deletion	0.2813	28.13% (+0.69%)	6.1488
Deletion-Reordering	0.2861	28.61% (+1.17%)	6.1444
Deletion-Replacement	0.2817	28.17% (+0.73%)	6.1495

To determine whether the difference in the overall BLEU score results is statistically significant for the baseline and extended systems, a two-sample t-test between proportions was performed with a statistical calculator [19]. The following hypothesis was made: the null hypothesis states that the difference between the results of the baseline system and the extended system is statistically significant. For this hypothesis, the p-value calculated must be bigger than the significance level alpha (α), so that the null hypothesis is not rejected. The p-value and α-value were calculated as follows, using the test set of 500 sentences as the samples:

$$p = 0.31625$$
$$\alpha = 0.05$$

Thus:

$$0.31625 > 0.05$$
$$p > \alpha \quad (1)$$

The p-value is bigger than the α-value, therefore the null hypothesis is not rejected, and the difference between the BLEU scores of the baseline and extended systems is statistically significant.

VII. CONCLUSION AND FUTURE WORK

Although the results showed marginal improvements, it indicates that there is potential for a machine translation system for English to Setswana using these pre- and post-processing methods.

This is an initial experiment and only one reference translation was used. However, for an automatic evaluation to be truly successful, a number of reference translations are needed. The need for these extra reference data is because two separate translations of the same text done by the same (or different) translator are seldom identical. Synonyms play a big part in translations and a SMT system does translations based on its language model, which might not always contain all possible synonyms of a target language. A machine translation system will only be an effective tool to human translators if the translator does not spend more time adjusting the output than doing a translation from scratch. A human evaluation will certainly give information as to how good the quality of the machine translation really is and how useful it would be in an everyday working environment. For future experiments, more reference translations and human evaluations will be used.

Also included in future work, is the assessment of the linguistic rules in isolation. This will determine the effectiveness of the rules' application. The rearrangement of the rules before implementation might have a positive effect on the success of other rules, by ensuring that one rule doesn't overwrite another in the processing stage. The rules can also be extended to include the correct translation of the time forms of the verbs as well as the direct relative verb construction.

As the results obtained indicate, SMT systems with these particular pre- and post-processing methods show that by developing SMT systems for a resource scarce language like Setswana, improvements in the translation quality can be achieved. However, because the development of machine translation systems is never-ending and because there is still room for improvement of the system as a whole, continuous effort will be made to achieve the highest translation quality possible.

REFERENCES

[1] South African Government information: South Africa at a glance, http://www.info.gov.za/aboutsa/glance.htm, 2012.

[2] Republic of South Africa, "Constitution of the Republic of South Africa: Chapter 2 - Bill of Rights", http://www.info.gov.za/documents/constitution/1996/96cons2.htm, 1996.

[3] D.J. Prinsloo and G-M. de Schryver, "Managing eleven parallel corpora and the extraction of data in all official South African languages," In Multilingualism and Electronic Language management. Proceedings of the 4th international MIDP Colloquium, 22-23 September 2003, Bloemfontein, South Africa. W. Daelemans, T. du Plessis, C. Snyman and L. Teck (eds.), pp. 100-122, Pretoria, Van Schaik Publishers, 2005.

[4] D. Joffe, "African Languages: Setswana (Tswana)," http://africanlanguages.com/setswana/, 2012.

[5] G. De Pauw, P.W. Wagacha and G.M. De Schryver, "Towards English-Swahili machine translation," Proceedings of Machine translation and morphologically rich languages: Research workshop of the Israel Science Foundation, pp. 1-2, University of Haifa, Israel, 2011.

[6] See http://translate.google.com/ for more information on Google Translate.

[7] K. Papineni, S. Roukos, T. Ward and W.J. Zhu, "BLEU: A method for automatic evaluation of machine translation," Proceedings of the 40th Annual Meeting of the Association for

Computational Linguistics, pp. 311-318, Philadelphia, USA, 2002.

[8] G. Doddington, "Automatic evaluation of machine translation quality using n-gram co-occurrence statistics," Proceedings of the 2nd International Conference on Human Language Technology Research, pp. 138-145, San Diego, California, 2002.

[9] H.J. Groenewald and L. Du Plooy, "Processing parallel text corpora for three South African language pairs in the Autshumato project," Proceedings of the 2nd Workshop on African Language Technology, pp.27-30, Valetta, Malta, 2012.

[10] M. Griesel, C.A. McKellar and D. Prinsloo, "Syntactic reordering as preprocessing step in statistical machine translation from English to Sesotho sa Leboa and Afrikaans," Proceedings of the 21st annual Symposium of the Pattern Recognition Association of South Africa (PRASA), pp. 205-110, Stellenbosch, South African, 2010.

[11] A.S. Berg and R.S. Pretorius, "Tswana: taalkunde, leeswerk en stelwerk," Study guide ATSW 114 A & 124 A, North-West University, Potchefstroom Campus, South Africa, 2009.

[12] K. Toutanova, D. Klein, C.D. Manning and Y. Singer, "Feature-rich part-of-speech tagging with a Cyclic Dependency Network, Proceedings of HLT-NAACL, pp. 252-259, Edmonton, Canada, 2003.

[13] See http://www.perl.org/ for more information about the programming language.

[14] R. Moore, "Fast and accurate sentence alignment of a bilingual corpus," Proceedings of the 5th conference of machine Translation in the Americas, pp. 135-144, Tiburon, CA, 2002.

[15] P. Koehn et al., "Moses: Open source toolkit for statistical machine translation," Proceedings of the ACL demo an poster sessions, pp. 177-180, Prague, Czech Republic, 2007.

[16] A. Stolcke, "SRILM – an extensible language modeling toolkit," Proceedings of the 7th International Conference on Spoken Language Processing, pp. 901-904, Denver, Colorado, 2002.

[17] D. Jurafsky and J.H. Martin, "Speech and language processing: an introduction to natural language processing, computational linguistics and speech recognition," 2nd ed. pp. 897-931, New Jersey: Pearson Education, 2009.

[18] See http://www.nwu.ac.za/ctext for more information on the Centre for Text Technology (CTexT®).

[19] StatPac, "The Statistics Calculator: statistical analysis at your fingertips," http://www.statpac.com/statistics-calculator/percents.htm, 2012.

Classification of Noun-Noun Compound Semantics in Dutch and Afrikaans

Ben Verhoeven, Walter Daelemans
CLiPS – Computational Linguistics Group
University of Antwerp
Antwerp, Belgium
{Ben.Verhoeven;Walter.Daelemans}@ua.ac.be

Gerhard B van Huyssteen
CTexT – Centre for Text Technology
North-West University
Potchefstroom, South Africa
Gerhard.Vanhuyssteen@nwu.ac.za

***Abstract*—This article presents initial results on a supervised machine learning approach to determine the semantics of noun compounds in Dutch and Afrikaans. After a discussion of previous research on the topic, we present our annotation methods used to provide a training set of compounds with the appropriate semantic class. The support vector machine method used for this classification experiment utilizes a distributional lexical semantics representation of the compound's constituents to make its classification decision. The collection of words that occur in the near context of the constituent are considered an implicit representation of the semantics of this constituent. F-scores were reached of 47.8% for Dutch and 51.1% for Afrikaans.**

Keywords—compound semantics; Afrikaans; Dutch; machine learning; distributional methods

I. INTRODUCTION

Computational language understanding can be seen as one of the major goals of research in computational linguistics and natural language processing (NLP). However, many issues need to be resolved before we can even approximate human level language understanding. A notable obstacle, for example, is the productivity that a language exhibits in creating new words. An important and very productive word formation process, in at least Germanic languages, is compounding [1:141]. Since these new words are not available in a computational dictionary and their meanings are hence not explicated, a computational system will have trouble interpreting the meaning of these words. Existing NLP applications, such as question answering, information extraction and machine translation systems, will benefit from better compound understanding. This paper presents initial results on first-generation semantic analyzers for Dutch and Afrikaans noun-noun compounds.

This research builds to a great extent on techniques previously used and discussed by Ó Séaghdha [2] for English and Verhoeven [3] for Dutch. Some results of the latter are revisited in this article.

The structure of this paper will be as follows. First, a summary of related research on the topic will be presented. This summary will focus on the techniques used in our own research. We then describe our annotation scheme and process for the Dutch and Afrikaans noun-noun compounds. The classification experiments are then discussed, after which we present our results and propose some directions for further research.

II. RELATED RESEARCH

Past research on semantic analysis of noun-noun compounds has focused almost exclusively on English. The problem of semantically analyzing these compounds was mostly considered a supervised machine learning problem. Different approaches were proposed considering two main characteristics of the research: the scheme of categories being used for the semantic classification of the compounds, and the features that the machine learning algorithm uses to classify the compounds.

A. Classification Schemes

Several attempts have been made in the past to come up with appropriate classification schemes for noun-noun compound semantics. These schemes are mainly inventory-based in that they present a limited list of predefined possible classes of semantic relations a compound can have. Early work in computational research is due to Warren [4], Finin [5] and Lauer [6].

In some cases, proposed classes are abstractly represented by a paraphrasing preposition as in [6], [7] and [8]. For example, all compounds that can be paraphrased by putting the preposition 'of' between the constituents belong to the class OF, e.g. a 'car door' is the 'door of a car'. Another possibility is using predicate-based classes where the relations between the constituents are not merely described by a preposition but by definitions or paraphrasing predicates for each class. The class AGENT would contain compounds that could be paraphrased as 'X is performed by Y' [9], e.g. *enemy activity* can be paraphrased as 'activity is performed by the enemy'. Different schemes vary from 9 to 43 classes with kappa scores for inter-annotator agreement ranging from 52% to 62% [2][4][7] [10][11][12][13][14].

B. Features

With regard to the information used by the classifier to assign the classes to the compounds, two main roads are

available, *viz.* taxonomy-based methods, or corpus-based methods.

Taxonomy-based methods (also called semantic network similarity [15]) base their features on a word's location in a taxonomy or hierarchy of terms. Most of the taxonomy-based techniques use WordNet [16] for these purposes; especially the hyponym information in the hierarchy is used. A bag of words is created of all hyponyms and the instance vector contains binary values for each feature (the feature being whether the considered word from the bag of words is a hyponym of the constituent or not). Kim and Baldwin reached an accuracy of 53.3% using only WordNet [9]. Other research was based on Wikipedia as a semantic network [17] or the MeSH hierarchy of medical terms [18].

Corpus-based methods use co-occurrence information of the constituents of the selected compounds in a corpus. The underlying idea – the distributional hypothesis – is that the set of contexts in which a word occurs, is an implicit representation of the semantics of this word [17]. This information can be used in different ways. Ó Séaghdha [2] describes measures of lexical similarity and relational similarity.

The lexical similarity measure assumes that compounds are semantically similar when their respective constituents are semantically similar. The co-occurrences of both constituents will be combined to calculate a measure of similarity for the entire compound. Accuracies[1] of 54.98% [12][17] and 61% have been reached [2][20].

The relational similarity measure assumes two pairs of constituents "to be similar if the contexts in which the members of one pair co-occur are similar to the contexts in which the members of the other pair co-occur" [2:118]. Ó Séaghdha and Copestake [17] report an initial accuracy of 42.34%. This result was improved to 52.6% in [2]. Lapata and Keller [8] report an accuracy of 55.71% with web-based relational similarity. Their corpus-based similarity's accuracy was only 27.85%.

Nastase *et al.* [21] extract grammatical collocations of the constituents from a corpus and use it as features for the classifier. This collocation includes words that appear with the target word in a grammatical relation, e.g. subject, object, etc.

Corpus-based and taxonomy-based methods have also been combined by several researchers. Accuracies of 58.35% [19], 79.3% [12] and even 82.47% [21] were reported.

III. Annotation

In order to perform a supervised machine learning experiment, we need semantic information of compounds that machine learning algorithms can learn from. There is thus a need for examples with an explicit description of the compound semantics, as is created through manually annotating data.

The compounds considered for manual annotation are only those noun-noun compounds that do not occur in a dictionary – otherwise a semantic classification is both unnecessary and unwanted: unnecessary because there is already a gloss for the compound present (the meaning is thus already known), and unwanted because we want to train our classifier on the systematics that will be found in the semantics of newly produced compounds. However, the constituents of these compounds are required to appear in a dictionary. If the constituents would not be present in a dictionary, their individual meanings would not be known to us and semantically relating an unknown word to some other word seems pointless. Hence, compounds with proper nouns (e.g. *Beneluxland* 'Benelux country') will be excluded from our dataset.

A. Scheme and Guidelines

For our research, we adopted the annotation scheme and guidelines created by Ó Séaghdha [2], which were by and large based on Levi's set of categories from 1987 [2]. The guidelines were developed for semantic annotation of English noun-noun compounds, so some adaptations were in order. These adaptations mainly existed of supplementing the guidelines with Dutch and Afrikaans examples. More details on other changes can be found in [3].

The annotation tag of each compound consists of three parts: the category, the annotation rule by which the category is determined, and the direction in which the rule applies. The annotation scheme will be summarized here; the complete guidelines can be found on the project website[2].

Ó Séaghdha [2] describes eleven classes of compounds; six of these classes are semantically specific. These classes include:

- BE: The compound can be rewritten as 'N2 which is (like) (a) N1' with N1 and N2 being the two constituents nouns. Example: *woman doctor*
- HAVE: The compound denotes some sort of possession. Part-whole compounds, typical one-to-many possession, compounds expressing conditions or properties and meronymic compounds belong here. Example: *car door*
- IN: The compound denotes a location in time or place. Example: *garden party*
- ACTOR: The compound denotes a characteristic event or situation and one of the constituents is a salient entity. Example: *enemy activity*
- INST: The compound denotes a characteristic event and there is no salient entity present. Example: *cheese knife*
- ABOUT: The compound describes a topical relation between its constituents. Example: *film character*

1 The accuracies presented in the related research section are mentioned as an indication of those systems' performance. Comparison with our own results is not in order due to the use of different data, methods, etc.

2 http://tinyurl.com/aucopro

The other five categories are less specific. The MISTAG and NONCOMPOUND categories serve to classify compounds that do not belong in the dataset. MISTAG refers to the fact that one or both of the constituents is not a common noun (e.g. *London Town*, where N1 is a proper noun). NONCOMPOUNDs are not two-noun compounds (e.g. 'a salt and *pepper beard*'). The REL class describes compounds with a clear meaning that does not belong to any of the other classes, but of which the relation between the constituents seems productive (e.g. *sodium chloride*). The LEX category is almost the same as REL, but the relation does not seem to be productive (e.g. *monkey business*). The UNKNOWN category is for correct noun-noun compounds of which the meaning is not clear enough to annotate.

B. Dutch

The Dutch noun-noun compounds were taken from a compound list that was extracted from the e-Lex Dutch lexicon[3]. This compound list was already split into constituents and the POS tags of the constituents were available. The WNT (Woordenlijst Nederlandse Taal) lexicon [22] was used to check the occurrence of the compounds and constituents in a dictionary. The eventual compound list contained 1802 Dutch noun-noun compounds.

The Dutch compound set was annotated by a student in linguistics that played no role in the development of the annotation guidelines. One of the authors of this paper annotated a subset of 500 compounds to be able to calculate an inter-annotator agreement (IAA). Both annotators are native speakers of Dutch. The reported IAA was 60.2% (Kappa = 0.60) [3].

C. Afrikaans

The Afrikaans noun-noun compounds were taken from the CKarma list of splitted compounds [23]. Since there were no POS tags available, these compounds were manually selected from the list. These compounds and their constituents were not crosschecked with a dictionary; this will be the case in future research. The compound list contained 1500 Afrikaans noun-noun compounds.

The complete Afrikaans compound set was annotated by three bachelor students in language, all native speakers of Afrikaans. The pair-wise average IAA was 53.4% (Kappa = 0.53). This IAA is a bit lower than our IAA for Dutch, possibly due to the fact that lexicalized compounds were not removed from the annotation list. They might be harder to annotate because their lexicalized meaning is not always a logical semantic relation between their constituents and may not fit into one of our categories then. Take the Afrikaans *naaldenkoker* as example; this compound has 'needle case' as literal meaning, but it also has a lexicalized meaning: 'dragonfly'. It is clear that lexicalized compounds may cause annotation difficulties.

3 This compound list was created by Lieve Macken of the LT3 research group at University College Ghent.

IV. Experiment

The conducted experiments were based on those conducted by Ó Séaghdha [2]. We will provide a description of our own experimental setup here. An in-depth discussion of the methodology and more extensive experimentation on the Dutch data can be found in [3].

Our classification experiment is based on a combination of the distributional hypothesis (as proposed above) with the idea of analogical reasoning. It is assumed that the semantic category of a compound can be predicted by comparing compounds with similar meanings [2].

A. Lexical Similarity

The lexical similarity measure is a corpus-based method of feature selection. As described above, this measure will compare the semantic similarities of the constituents of the considered compounds. The modifiers of the compounds (normally the left-hand members of the compound) will be compared with each other and the compound heads (normally the right-hand members of the compound) will be compared with each other. Two compounds, for example 'flour can ' and 'corn bag' will be considered similar if they have similar modifying constituents ('flour' and 'corn') and similar head constituents ('can' and 'bag'). In this example, the similarity would be rather high because the compounds both denote a container with its content.

B. Vector Creation

In order to perform a classification experiment, one needs the information for each instance (in this case: each compound) to be stored in a vector. This section will describe the creation of these vectors.

1) Bag of words (BOW)

For every compound constituent, the co-occurrence context was calculated. For this purpose, for each instance of the constituents in the corpus, the surrounding *n* words (that belong to the 10,000 most frequent words of the corpus) were held in memory. The number of context words was 3 or 5 to both the left and right hand side of the constituent in the two variants of the experiment. The relative frequencies of these context words (the number of times the word appeared in the context of the constituent, divided by the frequency of the constituent in the corpus) for each constituent were stored.

For Dutch, the Twente News Corpus [24] was used. This is a 340 million word corpus of newspaper articles. For Afrikaans, we used the Taalkommissie corpus [25], a 60 million word corpus that consists of a variety of text genres.

A concatenation of the constituent data is used to create the instance vector features. Each instance vector contains the compound it represents, its category, direction and annotation rule, and the relative frequencies for the 1000 most frequent words for each constituent (hence 2000 per compound). However, for purposes of training data in our experiment, the vectors are stripped from their compound, direction and rule, leaving only the category and the features. Compounds of which one or both of the constituents did not appear in the corpus were excluded from the data.

The classification experiment dealt with those compounds that are annotated with a semantically specific category. This means that only compounds with the category tags BE, HAVE, IN, INST, ACTOR and ABOUT were used for the experiments. The final vector set for Afrikaans contains 1439 compounds, while the final vector set for Dutch has 1447 compounds. The class distributions for Dutch and Afrikaans are presented in Table 1.

TABLE I. CLASS DISTRIBUTIONS FOR DUTCH AND AFRIKAANS

	Dutch		**Afrikaans**	
BE	105	7.3%	359	25.0%
HAVE	233	16.1%	140	9.7%
IN	428	29.5%	299	20.8%
ACTOR	62	4.3%	126	8.8%
INST	235	16.2%	108	7.5%
ABOUT	384	26.6%	407	28.2%
Total	1447		1439	

2) Principal Component Analysis

The BOW approach that was described so far takes the occurrence of each word as one attribute in the vector. Our vectors thus have 2000 attributes and one class (the category) each. This makes our experimentation computationally rather expensive. Principal component analysis (PCA) was used to reduce the dimensionality of our vectors to improve the performance of our system.

Performing PCA on a matrix or vector of data transforms this data by mathematically optimizing the variance between the instances. The vectors will reduce in size because correlated attributes will be fused into new attributes that are called principal components (PCs) [3:42].

The 'PCA Module for Python', as implemented by Risvik [26] was used to perform these mathematical transformations on our data. Apart from our BOW vectors, we now also have a PCA vector for both context variants.

C. Machine Learning

For the actual machine learning experiments on the four sets of vectors (BOW and PCA, each with 3 or 5 context words), we used the SMO algorithm, which is WEKA's [27] support vector machines (SVM) implementation. Automatic optimization of the parameters was performed by the CVParameterSelection function.

We used 10-fold cross-validation; the classifier was trained and tested ten times on a different train and test set. The ten folds cover the whole data set maximally. The average results and standard variation of these ten runs are a representation of the performance of this classifier.

V. RESULTS

Since this is the first research on both Dutch and Afrikaans, we will assume the most frequent class probability in the datasets as baselines for these classifiers. This baseline is calculated by dividing the count of the most frequent class by the total number of compounds in the dataset. This number represents the accuracy that can be obtained by always guessing this most frequent class as the output class. For Dutch, this baseline is 29.5% (428 instances of class IN on a total of 1447 compounds) [3]. For Afrikaans, this baseline is 28.2% (407 instances of class ABOUT on a total of 1439 instances).

TABLE II. RESULTS OF SMO CLASSIFIER ON DUTCH COMPOUND SEMANTICS

	Precision	**Recall**	**F-Score**
BOW 3	47.6	48.0	47.8
PCA 3	41.7	46.2	41.7
BOW 5	47.7	48.0	47.8
PCA 5	43.0	47.6	43.6

All results in Table 2 of the classification experiment with Dutch compounds show a significant improvement over the most frequent class baseline (29.5%). The BOW approach seems to do better than the PCA results with an F-score of 47.8% for both the 3 and 5 word variant. The results for the PCA approach (41.7% and 43.6%) are somewhat lower, but still significantly higher than the baseline.

TABLE III. RESULTS OF SMO CLASSIFIER ON AFRIKAANS COMPOUND SEMANTICS

	Precision	**Recall**	**F-Score**
BOW 3	50.8	51.6	51.1
PCA 3	47.7	50.5	47.5
BOW 5	50.3	50.8	50.5
PCA 5	49.3	51.3	48.5

Table 3 shows that the classification experiment with Afrikaans compounds also performs significantly better than its most frequent class baseline of 28.2%. The highest F-score reached was 51.1% for the BOW approach with 3 context words. These results are even slightly better than our results for Dutch.

This 3% improvement of the Afrikaans over the Dutch performance may be ascribed to the final annotation list for Afrikaans being a combination of the semantic annotations of three persons. In taking the most agreed upon class for each compound, we may have reached a better approximation of the actual compound semantics than when using the annotation list of just one person, as we did for Dutch. However, this hypothesis remains a subject for further research.

VI. CONCLUSION AND FURTHER WORK

This paper presented, for the first time, exploratory research on the semantic classification of noun-noun compounds in Dutch and Afrikaans. The results show that a first approach, based on corpus-based semantic representations, already provides promising results for both Afrikaans (highest F-score of 51.1%) and Dutch (highest F-score of 47.8%). Although a full comparison with earlier systems for English is not appropriate, we can note that the results of our initial classifiers already compare favorably to previous results for English; for example, Ó Séaghdha reaching an F-score of 58.8% (accuracy

of 61%) also using only lexical similarity with a training set of 1443 compounds [2].

The performance of the classifiers significantly outperforms the most frequent class baselines. The BOW approach turns out to provide better results than the PCA approach, because it seems that some of the information in the vectors is lost during PCA calculation. It is nevertheless our intention to further explore the PCA approach and variants in future research, because the computational performance of the approach is important in practical applications. We will also investigate alternative methods for constructing corpus-based lexical semantic representations, explore the use of lexical databases (a lexical semantic network such as WordNet is also available for Dutch, while a small-scale WordNet of Afrikaans is also available), and experiment with context-based representations.

We will try and test other machine learning algorithms, such as memory-based learning. An attempt will be made to improve the IAA's as well.

The semantics of other compounds than noun-noun compounds, such as verb-noun and adjective-noun compounds, will be investigated from a linguistic perspective, in order to determine the viability to model such semantic relations computationally.

ACKNOWLEDGMENT

The current paper fits in a broader research on automatic compound processing. Automatic Compound Processing (AuCoPro) is a mutual project by research groups of the North-West University (Potchefstroom, South Africa), the University of Antwerp (Belgium) and Tilburg University (The Netherlands). The University of Antwerp deals mainly with the compound semantics subproject, Tilburg University deals mainly with compound splitting. North-West University works on the Afrikaans aspects of both subprojects.

This research was co-funded by a joint research grant of the Nederlandse Taalunie (Dutch Language Union) and the Department of Arts and Culture (DAC) of South Africa and a grant of the National Research Foundation (NRF) (grant number 81794).

We also want to acknowledge the work of the bachelor students of the North-West University, Potchefstroom Campus (Carli de Wet, Nadia Schultz, Benito Trollip and Joanie Liversage) that annotated the Afrikaans compounds as part of their bachelor dissertation.

REFERENCES

[1] G. Booij, The Morphology of Dutch, Oxford: Oxford University Press, 2002.

[2] D. Ó Séaghdha, "Learning compound noun semantics," Ph.D. thesis, University of Cambridge, UK, 2008.

[3] B. Verhoeven, "A computational semantic analysis of noun compounds in Dutch," M.A. thesis, University of Antwerp, Belgium, 2012.

[4] B. Rosario, and M. Hearst, "Classifying the semantic relations in noun compounds via a domain-specific lexical hierarchy," in Proc. EMNLP, 2001, 82-90.

[5] T. W. Finin, "The semantic interpretation of compound nominal," in Proc. AAAI, 1980.

[6] M. Lauer, "Designing statistical language learners," Ph.D. thesis, Macquarie University, Australia, 1995.

[7] R. Girju, D. Moldovan, M. Tatu, and D. Antohe, "On the semantics of noun compounds," in Computer Speech and Language, vol. 19, 2005, pp.479-496.

[8] M. Lapata, and F. Keller, "The web as a baseline: evaluating the performance of unsupervised web-based models for a range of NLP tasks," in Proc. NAACL-HLT, 2004, pp. 121-128.

[9] S. N. Kim, and T. Baldwin, "Automatic interpretation of noun compounds usin WordNet similarity," in Proc. IJCNLP, 2005, pp. 945-956.

[10] P. Nakov, "Noun compound interpretation using paraphrasing verbs: feasability study," in Proc. AIMSA, 2008.

[11] D. Moldovan, A. Badulescu, M. Tatu, D. Antohe, and R. Girju, "Models for the semantic classification of noun compounds," in Proc. NAACL-HLT Workshop on Computational Lexical Semantics, 2004, pp. 60-67.

[12] S. Tratz, and E. Hovy, "A taxonomy, dataset, and classifier for automatic noun compound interpretation," in Proc. ACL, 2010, pp. 678-687.

[13] K. Barker, and S. Szpakowicz, "Semi-automatic recognition of noun-modifier relationships," in Proc. ICCL, 1998, pp. 96-102.

[14] D. T. Wijaya, and P. Gianfortoni, "'Nut-case: what does it mean?': Understanding semantic relationship between nouns in noun compounds through paraphrasing and ranking the paraphrases," in Proc. SMER, 2011.

[15] D. Ó Séaghdha, "Semantic classification with WordNet kernels," in Proc. NAACL-HLT Short Papers, 2009, pp. 237-240.

[16] G. A. Miller, "WordNet: a lexical database for English," Communication of the ACM, vol. 38, 1995, pp. 39-41.

[17] D. Ó Séaghdha, and A. Copestake, "Co-occurrence contexts for noun compound interpretation," in Proc. Workshop on a Broader Perspective on Multiword Expressions, 2007, pp. 57-64.

[18] Z. Harris, Mathematical Structures of Language. New York: Interscience, 1968.

[19] D. Ó Séaghdha, "Annotating and learning compound noun semantics," in Proc. ACL Student Research Workshop, 2007, pp.73-78.

[20] D. Ó Séaghdha, and A. Copestake, "Semantic classification with distributional kernels," in Proc. COLING, 2008, pp. 649- 656.

[21] V. Nastase, J. Sayyad-Shirabad, M. Sokolova, and S. Szpakowicz, "Learning noun-modifier semantic relation with corpus-based and WordNet-based features," in Proc. AAAI, 2006, pp. 781-787.

[22] Nederlandse Taalunie, "Bronbestand woordenlijst Nederlandse taal," Internet: http://www.inl.nl/tst-centrale/nl/producten, 2005 [18/09/2012].

[23] CTexT, CKARMA ("C5 KompositumAnaliseerder vir Robuuste Morfologiese Analise") [C5 Compound Analyser for Robust Morphological Analysis]. Potchefstroom: Centre for Text Technology (CTexT), North-West University, 2005.

[24] R. Ordelman, F. de Jong, A. Van Hessen, and H. Hondorp, "TwNC: a multifaceted Dutch news corpus," ELRA Newsletter, vol. 12, pp. 3-4, 2007.

[25] Taalkommissie van die Suid-Afrikaanse Akademie vir Wetenskap en Kuns, Taalkommissiekorpus 1.1. Potchefstroom: Centre for Text Technology (CTexT), North-West University, 2011.

[26] H. Risvik, "PCA module for Python," Internet: http://folk.uio.no/henninri/pca_module, 2008 [27/05/2012].

[27] I. H. Witten, E. Frank, and M. A. Hall, Data Mining: Practical Machine Learning Tools and Techniques (Third Edition). Burlington, MA: Morgan Kaufmann, 2011.

Handwritten Symbol Recognition using an Ensemble of SVM Classifiers

Ronald Clark*, Quik Kung*, M.A. Van Wyk†
University of the Witwatersrand
* {ronald.clark, quik.kung}@students.wits.ac.za
† ma.vanwyk@wits.ac.za

***Abstract*—Support Vector Machines (SVM) have proven to be highly accurate in classifying handwritten mathematical symbols - especially when a diverse range of features is used. This paper investigates the classification of handwritten mathematical symbols using an SVM method and an ensemble of three different feature sets in order to minimise the number of training samples required and achieve accurate classification rates. The architecture of the system consists of pre-processing, symbol segmentation and classification. Three SVMs are used, each operating on different feature sets: sample point coordinates (SVM 1) turning angles and their derivatives (SVM 2) and global features (SVM 3). The symbol classifications are combined using various decision fusing techniques. The system was tested using a small set of 252 samples consisting of 41 classes or 6 samples per class. The results yielded a 97.20% correct classification rate using feature set 1 while a rate of only 90.91% was obtained using a single high-dimensional SVM combining the three feature sets. The ensemble configuration further improved the classification rate to 98.601% using a simple average-based decision fusion scheme. As such, the proposed SVM ensemble considerably increases the classification accuracy when only a few training samples are available.**

NOMENCLATURE

Ω	Set of all symbol classes.
h_i	Possible symbol classes (ie. the elements of Ω)
p_{ji}	Confidence that the input of the j'th classifier belongs to class i.
$m_k(h_i)$	Proposition that the sample belongs to class i. In this paper it is assumed to be identical to the confidence output p_{ji}.
$P(h_j)$	Probability that the j'th classifier labels a sample with the class proposition h_j.
$P(\mathbf{h}\|c_k)$	Likelihood of the propositions given input class c_k.
$P(c_k\|\mathbf{h})$	Posterior probability of the class label given the propositions $\mathbf{h}$.
$\mathbf{x_i}$	Feature vector belonging to the ith sample.
$\mathbf{p_j}$	Vector representing the soft-decision output of an SVM.
L	Total number of classifiers.
c	Total number of symbol classes.
$\mathbf{p_i}$	Coordinate vector of sample point i.
$\phi(\mathbf{x})$	Map to higher-dimensional feature space.
$K(\mathbf{x}, \mathbf{x_i})$	Kernel function defining the inner-product space.
$\mathbf{w}$	Normal vector defining the optimal hyperplane.
b	Offset defining the optimal hyperplane.
α_i	A Lagrange multiplier.
ϵ	Error term used in finding the SVM decision boundary.

I. INTRODUCTION

Handwriting recognition is the process of converting characters drawn as a series of graphical marks into their symbolic representation which can be further processed by a computer system. This process may be carried out in an *online* or *offline* manner. In the *online* case (which is considered in this paper) recognition is performed at the same time as the writing process which means that information related to the writing dynamics and stroke ordering are available. The complete symbol recognition process involves three steps:

1) Pre-pocessing (de-skewing, de-hooking, conversion to equidistant samples, smoothing etc.)
2) Segmentation to isolate symbols
3) Classification of symbols

The classification of symbols can be done using either a parametric or non-parametric classifier. Parametric classifiers operate on a number of specially-selected features that have been extracted from the symbol to perform classification while non-parametric classifiers simply operate on the entire input data set. Parametric classifiers often achieve better recognition rates and classification times compared to non-parametric classifiers and are therefore the preferred method in many systems [1].

As such, this paper describes a parametric character recognition process which involves two primary steps: feature extraction and classification [2]. Most modern systems make use of either Support Vector Machines (SVM's) or Artificial Neural Networks (ANNs) as these techniques have proven very effective for online symbol recognition [3]–[5]. SVMs have been successfully used in face detection, handwritten digit recognition and data mining [6].

The accuracy of the SVM can be improved in two ways: by increasing the number of features or increasing the number of samples used for training. The training process, however, is very computationally-expensive which leads to a tradeoff between the desired accuracy and acceptable training time. Furthermore, this problem is exacerbated by the fact that the number of samples required to achieved a certain classification accuracy as well as the training time per sample increases with the dimensionality of the feature vectors. This can render the use of even a moderately-sized training set unfeasible.

Numerous studies have investigated the idea of combining multiple SVMs to improve classification accuracy through techniques such as boosting and bagging. For example, Kim et al. obtained an improvement in performance of 1.81% using a boosted combination of 10 multi-class SVMs to achieve a classification accuracy of 97.83% [6]. Even so, a relatively large number of samples was still needed to achieve this accuracy with a training set of 3828 and a test set of 1797 symbols.

This paper proposes the use of an ensemble of three SVMs, each operating on a unique low-dimensional feature set with the goal of minimising the number of training samples required while maximising the classification accuracy. The recognition of handwritten mathematical symbols to illustrate the performance gains achieved by the proposed ensemble of classifiers.

The paper is structured as follows. In Section II and III, a brief background is given of the SVM classifier and techniques for combining the output of multiple classifiers. In Section IV the pre-processing and symbol segmentation techniques are explored and the feature extraction is described. Lastly, in Section VI and Section VIII, the classification rates for the individual SVMs, a single optimised SVM as well as the ensemble configuration are analysed and discussed and relevant conclusions are drawn.

II. Support Vector Machines

SVMs work by finding a boundary in the feature space which maximises the distance between feature vectors belonging to two distinct input classes [7]. The decision boundary usually takes the form of a linear function which separates the two classes. In linearly inseparable problems, a non-linear decision surface is created by lifting the feature space into a higher dimensional space which allows a linear separating hyperplane to be found [8]. This hyperplane corresponds to a non-linear decision surface in the original feature space. The mapping is denoted by $\phi(\mathbf{x})$ which represents the map to the higher dimensional space where the data are linearly separable.

By using the kernel function $\mathbf{K}(\mathbf{x}, \mathbf{x}_i) = \phi(\mathbf{x_i})^{\mathbf{T}}\phi(\mathbf{x})$, the decision function of the SVM can be represented by:

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i y_i \mathbf{K}(\mathbf{x}, \mathbf{x_i}) + b \tag{1}$$

where $f(x)$ is the decision output, y_i is the label of the training symbol $\mathbf{x_i}$ and $\mathbf{x}$ is the symbol to be classified.

The parameters α_i and b are found during training which is performed by solving the following optimisation problem:

$$\min_{w,b} \frac{1}{2}||\mathbf{w}||^2 + C\sum_{\mathbf{i=1}}^{\mathbf{n}} \epsilon_\mathbf{i} \tag{2}$$
$$\text{subject to } y_i(\mathbf{w}^{\mathbf{T}}\phi(\mathbf{x}_i) + b) \geq 1 - \epsilon_i$$

Many kernel functions exist but a well-performing kernel, used in many optical character recognition (OCR) systems, is the radial basis function (RBF):

$$\mathbf{K}(\mathbf{x}, \mathbf{x}_i) = \exp(-\gamma||\mathbf{x} - \mathbf{x_i}||^2),\ \gamma > 0 \tag{3}$$

The constant C in Equation 2 is the penalty parameter of the error term and the constant γ in Equation 3 is a kernel parameter. Both of these parameters have a significant effect on the accuracy of the trained system and need to be carefully set prior to the training process. The method used for selecting these parameters is mentioned in Section VI.

Although SVMs are binary classifiers, multi-class classification is easily achieved by combining SVMs in a one-against-others or one-against-one scheme [9]. Although SVM training time is proportional to the square of the number of samples and thus relatively slow, actual classification is very fast and can be performed in real-time [9].

A. Parameter Selection

As described in Section II, there are two parameters that need to be set when using an RBF kernel: C and γ which need to be carefully selected prior to the training process.

The selection of these parameters can be automated by using the grid-search method described in [7]. This method selects the combination of parameters which give the best cross-validation accuracy during training by carrying out a brute force search of all the possible parameter combinations. This optimisation process only needs to be carried out once for a particular training set.

III. Combining Classifiers

Throughout this paper it is assumed that the output of the j'th SVM classifier is a vector of scores $\mathbf{p_j} = [p_{j0}p_{j1}...p_{jn}]$ which approximate the posterior probabilities of the input sample belonging to a certain symbol class c_i given the observed feature vector, ie. $p_{ji} = P(c_i|\mathbf{x})$. A decision rule is thus needed to determine the final symbol class based on these probability values.

A. Dempster's Rule of Combination

Dempster's rule considers a number of mutually exclusive and exhaustive propositions h_i, $i = \{1...n\}$ which form part of a universal set Ω. Each classifier indicates its opinion by producing a mass of belief function $m_k(h_i)$ over Ω which is an independent indication of the classifiers belief that the proposition is correct.

The combined masses of belief can then be found using Dempster's rule of combination:

$$m_{12}(A) = \frac{1}{1-K} \sum_{B \cap C = H} m_1(B) m_2(C) \tag{4}$$
$$K = \sum_{B \cap C = \phi} m_1(B) m_2(C)$$

The output quantity $m_{12}(A)$ represents a third mass function which combines pieces of evidence from the individual classifiers to produce stronger support for the most likely propositions.

The quantity $1 - K$ in Equation 4 is a normalisation coefficient which gives a measure of the conflict between the sources. If this quantity is near zero, the classifiers are in total disagreement and the Dempster rule is no longer valid.

B. Naive Bayes

The Bayes scheme also assumes that the individual classifiers produce independent predictions for each class type. If $P(h_j)$ denotes the probability that the j'th classifier labels a sample $\mathbf{x}$ (of class c_k) with the class proposition h_j, the likelihoods of the proposed classes can be calculated as follows [10]:

$$P(\mathbf{h}|c_k) = P(h_1, h_2, ..., h_L|c_k) = \prod_{i=1}^{L} P(h_i|c_k) \tag{5}$$

The posterior probabilities are then obtained by calculating [10]:

$$P(c_k|\mathbf{h}) = \frac{P(c_k)P(\mathbf{h}|c_k)}{P(\mathbf{h})} = \frac{P(c_k)\prod_{i=1}^{L} P(s_i|c_k)}{P(\mathbf{h})} \tag{6}$$

which can be used to classify the input $\mathbf{x}$. As the quantity $P(\mathbf{s})$ does not depend on the class type, the decision can be made using the quantity [10]:

$$\mu_k(\mathbf{x}) \propto P(c_k) \prod_{i=1}^{L} P(s_i|c_k) \tag{7}$$

C. Majority Vote

The majority vote ensemble technique chooses as the final decision the class that appears most often in the selections made by the component classifiers. By denoting the decision of j'th classifier as $d_{j,k} \in \{0, 1\}$, $j = 1, ..., L$ and $m = 1, ..., c$ where L is the number of classifiers and c is the number of symbol classes, the decision output will be class k if [11]:

$$\sum_{j=1}^{L} d_{j,k} = \max_{m=1}^{c} \sum_{j=1}^{L} d_{j,m} \tag{8}$$

If any "ties" result, the output of the classifier with the highest measure is taken as the final decision.

D. Average

The average method simply finds the average of the confidence outputs of the individual classifiers and assigns the class label with the highest average confidence.

E. Product

In this scheme, the maximum of the product of the classifier outputs for each class is used as the class label.

IV. System Architecture

The system uses a modular architecture. The input module captures strokes as an ordered series of (x_i, y_i) data point coordinates. These data points are preprocessed to reduce noise and decrease the number of points per stroke. The symbol segmentor then groups strokes based on a simple distance threshold to form symbols.

The classifier module operates on these symbols by extracting chosen features and sending the resulting feature vectors to the appropriate SVM for classification as illustrated in Figure 1.

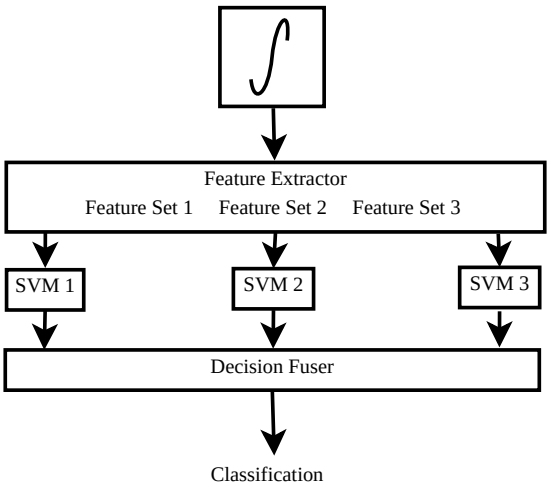


Figure 1. Ensemble SVM classifier

A. Pre-processing

The pre-processing stage involves filtering and re-sampling of the data points. Filtering is carried out by replacing the raw coordinates by a weighted sum of the neighbouring points. As in [12], three coefficient values are used:

$$\mathbf{p}_i^* = 0.25\mathbf{p}_{i-1} + 0.5\mathbf{p}_i + 0.25\mathbf{p}_{i+1} \tag{9}$$

This smoothing technique is computationally inexpensive and has proven to be very effective [12]–[14]. After the stroke has been smoothed, new samples are obtained by generating points that are equidistant with respect to arc length.

B. Symbol Segmentation

Individual strokes are grouped into symbols using a method similar to that of Ernesto [12]. In order to determine whether two strokes belong to the same symbol, the minimum distance between the sample points of the two strokes is compared to a threshold value d_{th}. The threshold value is dynamically adjusted according to the height of the second stroke to accomodate for symbols of various sizes (for example superscripts and subscripts). The threshold is determined as follows:

$$d_{th} = \frac{1}{10}\max(width,\ height) \tag{10}$$

If the distance between the end points of the strokes is lower than the threshold value, the strokes are concatenated to form a single continuous stroke.

C. Classification

Many methods exist for creating an ensemble of classifiers, however, the most important consideration is to ensure that the classification performance of the individual SVMs are independent and differ as much as possible from each other [15]. This is usually done by using different training sets for different SVMs which are obtained using techniques such as bagging, boosting or randomisation [6].

In this system, instead of varying the training sets, each SVM is trained using a unique set of features. That is, each vector contains different information about the symbol and not a different training set.

D. Feature Extraction

The features used for classification are similar to those proposed in [12]. The first two feature sets (FS) are derived from the local features of the stroke (s) of points $(p_1, \ldots, p_n)$ and the third FS is obtained by including the total number of points. The points are used in the order in which they are written, making them sensitive to writing direction. The feature sets have been chosen to capture the greatest variation and are as follows:

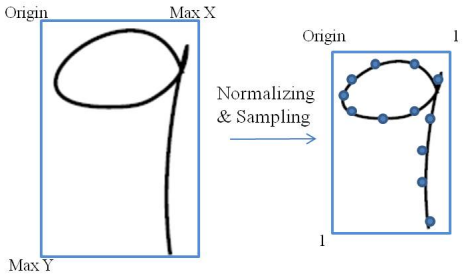


Figure 2. Normalizing and Sampling of a Symbol for FS 1.

- FS 1
  - 20 co-ordinate points: (x_i, y_i) of p_i
- FS 2
  - 19 turning angles: $\frac{\theta_i}{2\pi}$ where the turning angle is $\theta_i = \angle p_{i-1}\bar{p_i}p_{i+1}$
  - 18 derivatives of the turning angle: $\frac{(\theta_{i+1}-\theta_i)}{2\pi}$ and $\frac{(\theta_{i-1}-\theta_i)}{2\pi}$
- FS 3
  - Center of gravity: $x_g = \sum_{i=1}^{n} x_i/n$ and $y_g = \sum_{i=1}^{n} y_i/n$
  - Total length: l
  - Accumulated angle: $\theta_a = \sum_{i=1}^{n} \theta_i/2\pi$

FS 1 is chosen for its simplicity and direct approach of co-ordinate comparisons to be performed by the SVM. It takes the x and y co-ordinates of the strokes, with the origin at the top left corner, scales and normalizes the values, depending on the size of the written symbol, and stores these points as its feature vector or FS 1. The normalisation of the symbol correctly factors the symbol size in order to match to the training set provided. This feature can be seen in Figure 2 where a written symbol is sampled, preprocessed (smoothed) and then normalised before re-sampling.

FS 2 calculates the turning angles by comparing to a point to the one before and after it. The turning angle is obtained by the cosine rule and thus calculates the interior angle. The derivative of the angle is then calculated with respect to the neighbouring points. This FS is aimed at differentiating between symbols with similar structural characteristics in terms of point co-ordinates but which differ in angular integrity, such as sharp corners as opposed to gradual changes in direction. The process of determining the angle between three points in a stroke is shown in Figure 3.

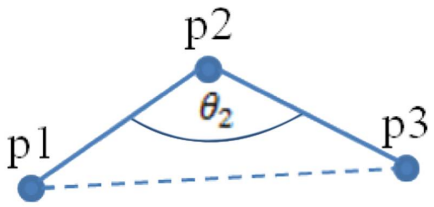


Figure 3. Calculation of Angle at a point for FS 2.

FS 3 consists of the center of gravity, the total length and the sum of all the angles of each stroke in a symbol. The centre of mass is found by summing all the x and y co-ordinates and dividing by the total number of points. The total length of the strokes is taken as the length from point to point after the symbol has been smoothed and the total angle is the summation of the constituent angles. FS 3 easily distinguishes between symbols with similar structure but with points clustered in a certain region. The total length and accumulated angles of the symbols are global properties to further strengthen this feature for SVM classification.

V. Testing Procedure

The system was trained on a small set of 252 samples consisting of the following 41 classes:

- Digits: 0-9
- Symbols: infinity (∞)
- Letters: a-d, i, m, k, s, x-z
- Greek letters: α, β, θ, ∂, ϵ, μ, σ, ω and π
- Operators: addition (+), subtraction (-), division (/), parenthesis, square root ($\sqrt{}$), summation ($\sum$), and integral ($\int$)
- Relations: equality (=), less than (<), greater than (>)

The samples for both the testing and training data sets were written by the same user on a Wacom Intuous3 6×8 tablet. The symbols were chosen randomly from the Aster database of mathematical expressions [16].

An SVM with an RBF kernel was used as it achieved better cross-validation rates during training compared to the linear, Gaussian and polynomial kernels. The classifiers were implemented and trained using the libSVM software library [17] which provides excellent tools for tuning the kernel and training parameters. The grid-search method was used to find the optimal SVM parameters. The output of the grid-search is shown in Figure 4 where it can be seen that the the optimal values of C and γ were 2^{10} and 2^{-5}, respectively.

VI. Results and Discussion

A complete set of correctly recognised symbols, extracted from the testing results, is shown in Figure 5.

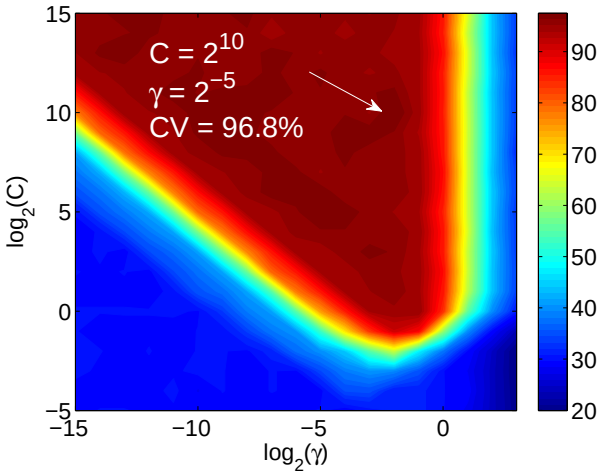


Figure 4. Output of the grid-search used for tuning the SVM parameters. A maximum cross validation accuracy of 96.8% is obtained when $C = 2^{10}$ and $\gamma = 2^{-5}$.

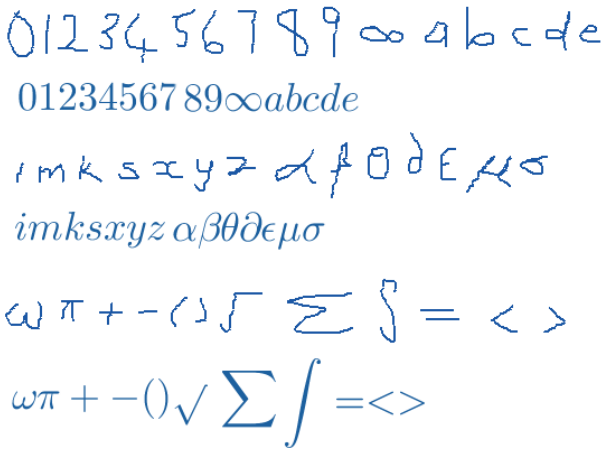


Figure 5. A set of correctly recognised symbols obtained during testing. The symbols were sourced from the Aster database of mathematical expressions. The symbols shown in this image are representative of the raw input to the system, ie. the results of the preprocessing stage are not shown. The typeset symbols have been rendered using LaTeX.

The aim of using an SVM ensemble is to extend the set of correctly recognised symbols beyond that of a single FS while still keeping the required number of training samples as low as possible. The different features enable the system to distinguish key characteristics in certain symbols that the other feature may not. A pertinent example is the digit '2' and the letter 'z'. Considering only the co-ordinate features, it can easily mistake the letter for the digit or vice verse. However, when considering the turning angle and the derivative, the top right corner of the letter 'z' will create a vast difference for the SVM classifier compared to the co-ordinate vector. This result is confirmed by the confusion matrices in Figure 6 which shows the probability outputs obtained from the SVM for each testing sample. The confidence level for the digit "2"

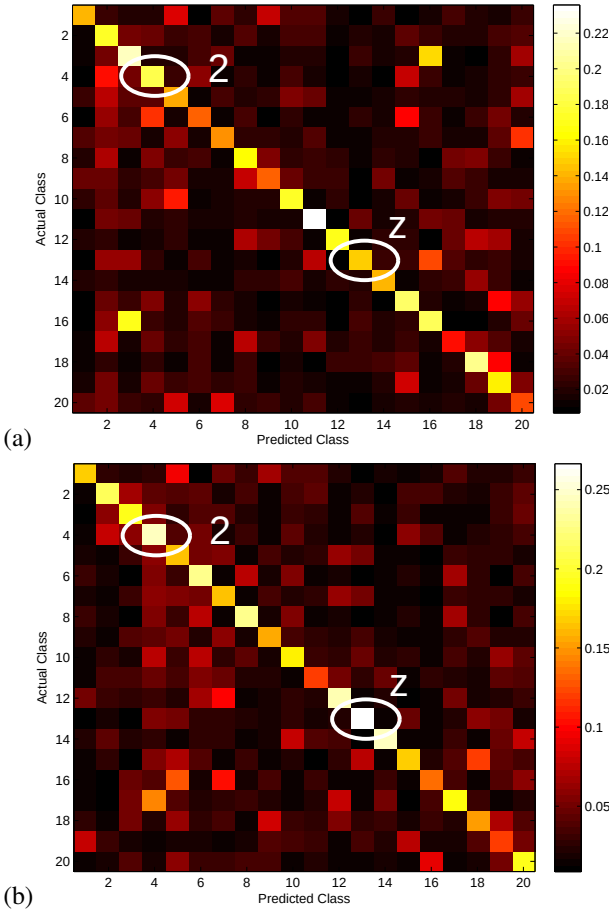


Figure 6. Confusion matrix of FS1 (a) and FS2 (b) for a subset of 20 testing symbols obtained from the probability output of the SVM.

is twice as high (≈ 0.22) for FS2 as it is for FS1 (≈ 0.11).

Table I
CLASSIFICATION RATES ACHIEVED BY COMBINING 3 SVMS WITH DIFFERENT FEATURE SETS USING A TRAINING SET AND TESTING SET EACH CONTAINING 252 SYMBOLS.

Decision Fuser	Correct Classification Rate (%)
FS1	97.203
FS2	70.63
FS3	88.11
Single SVM	90.91
Product	97.902
Average	98.601
Bayes	96.5
Dempster	97.202
Majority Vote	84.61

The recognition rates for the different feature sets are shown in Table I. The first four rows report the rates achieved by using the different feature sets independently and as a single classifier using all the features. The remaining rows show the result of using the three SVMs in an ensemble configuration with different decision-fusing schemes.

Table I shows that the classification rate achieved using only FS1 is higher than that of the single optimised SVM. This is most probably due to the severely limited size of the set of symbols used during training which leads to poor generalization performance for the high-dimensional classifier. As FS1 has significantly less features (40 compared to 79) it requires fewer training samples to achieve the same (and even better) classification accuracy which is a key advantage of using an ensemble of SVMs operating on small feature sets.

Out of the five decision fusion methods evaluated in this paper, the majority vote clearly performs the worst with a classification accuracy of only 84.61%. This is similar to the result achieved by Gorgevik et al., who attribute the poor performance of voting cooperation schemes to the limited information that is used about the member classifiers as no consideration is given to confidence outputs or second choices [6].

In this case, the "simple" average and product cooperation schemes outperform the more complicated naive Bayes method and Dempster rule. This is contrary to the result obtained in [6] where the simple cooperation schemes only achieve average recognition rates. Again, this is mainly due to the small training set size which limits the accuracy of the prior probabilities used in the Bayes scheme as well as the the output posterior probabilities generated by the SVM which are used by both the Dempster rule and the Bayes scheme. For example, Gorgevik et al. use a minimum training set size of 10000 samples to derive these values [2].

VII. FUTURE WORK

An important factor affecting the performance of the ensemble technique is the independence of the features and feature sets used by the classifiers. The independence of the features was, however, not verified in this paper as an adequate method for quantifying the independence could not be found. Three techniques that could possibly be used include principal component analysis, factor analysis and linear discriminant analysis which are commonly used for dimensionality reduction.

Furthermore, an intelligently-weighted sum ensemble could improve on the accuracy achieved by the average ensemble technique. In this case the output of the ensemble would be $w_1 \times$FS1 + $w_2 \times$FS2 + $w_3 \times$FS3 with $w_1 + w_2 + w_3 = 1$. Optimal values for w_1, w_2 and w_3 could then be found on the training set, possibly giving better results than the average.

VIII. CONCLUSION

Although SVMs have been shown to achieve high classification rates for handwritten symbols, they generally require a large number of training samples to achieve satisfactory performance. Because of the writer dependent nature of online handwritten symbol recognition, these samples need to be generated by the end user of the system which can be a time-consuming and tedious process.

In this paper, the use of an ensemble of SVM classifiers is investigated as the symbol recognition component of a handwritten mathematical expression recognition system. A number of decision fusion methods are considered to combine the outputs of the individual classifiers.

The results show that, for a small training set, a classifier using a feature vector with fewer components can increase the classification accuracy. This accuracy can be further enhanced by combining the output of multiple SVMs, each performing classification on a different set of features extracted from the same symbol. In this case, the optimal performance is achieved by a product-based combination of the individual SVM outputs.

IX. Acknowledgments

The authors would like to thank their colleagues Merelda Wu and Matthew Looi for their valuable suggestions and comments regarding this paper.

References

[1] D. Blostein. *Recognition of Mathematical Notation*, chapter 22. World Scientifc Publishing Company, 1996.

[2] Dejan Gorgevika, Dusan Cakmakov, and Vladimir Radevski. Handwritten digit recognition by combining support vector machines using rule-based reasoning. In *Proceedings of the 23rd International Conference on Information Technology Inferfaces*, pages 139–144, 2001.

[3] Y. LeCun. Energy-based models in document recognition and computer vision. In *Proceedings of the Ninth International Conference on Document Analysis and Recognition*, 2007.

[4] L. Bottou. Comparison of classifer methods: a case study in handwritten digit recognition. In *Proceedings of the 12th IAPR International Conference on Pattern Recognition*, 1994.

[5] T. M. English. comparison of neural network and nearest-neighbor classifers of handwritten lower-case letters. In *Proceedings of the Fifth IEEE International Conference on Neural Networks*, 1993.

[6] Hyun-Chul Kim, Shaoning Pang, Hong-Mo Je, Daijin Kim, and Sung Yang Bang. Pattern classification using support vector machine ensemble. *Journal of Pattern Recognition*, pages 160–163, 2002.

[7] A. Ben-Hur and J. Weston. A users guide to support vector machines. Technical report, Department of Computer Science. Colorado State University, 2010.

[8] B. Keshari and S.M. Watt. Hybrid mathematical symbol recognition using support vector machines. In *Proceedings of the Ninth International Conference on Document Analysis and Recognition*, 2007.

[9] I.-S. Oh and C. Suen. A class-modular feed-forward neural network for handwriting recognition. *Pattern Recognition*, 35:229–244, 2002.

[10] Ludmila I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms*. John Wiley & Sons, 2004.

[11] Nicholas Stepenosky, Deborah Green, John Kounios, and Christopher M. Clarkand Robi Polikar. Majority vote and decision template based ensemble classifiers trained on event related potentials for early diagnosis of alzheimer's disease. In *ICASSP*, 2006.

[12] Ernesto Tapia. Understanding mathematics: A system for the recognition of on-line handwritten mathematical expressions. Master's thesis, Freie Universität Berlin, 2005.

[13] R. Zanibbi, D. Blostein, and J. R. Cordy. Baseline structure analysis of handwritten mathematics notation. In *Proceedings of the IEEE ICDAR conference*, 2001.

[14] X.-D. Tian, H.-Y. Li, X.-F. Li, and L.-P. Zhang. Research on symbol recognition for mathematical expressions. In *Proceedings of the First International Conference on Innovative Computing, Information and Control*, 2006.

[15] Hyun-Chul Kim, Shaoning Pang, Hong-Mo Je, Daijin Kim, and Sung Yang Bang. Constructing support vector machine ensemble. *Pattern Recognition*, 36:2757–2767, 2003.

[16] T.V. Raman. Audio system for technical readings. Technical report, Cornell University, 1994.

[17] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:2701–2707, 2011.

Cross-Lingual Genre Classification for Closely Related Languages

Dirk Snyman[1], Gerhard B. van Huyssteen[1] & Walter Daelemans[2]
[1]Centre for Text Technology (CTexT®), North-West University, Potchefstroom, South-Africa
[2]CLiPS-CL, University of Antwerp, Antwerp, Belgium
[1]{Dirk.Snyman;Gerhard.Vanhuyssteen}@nwu.ac.za
[2]Walter.Daelemans@ua.ac.be

Abstract— **Resource-scarcity is a topic that is continually researched by the HLT community, especially for the South-African context. We explore the possibility of leveraging existing resources to help facilitate the development of new resources for under-resourced languages by using cross-lingual classification methods. We investigate the application of an Afrikaans genre classification system on Dutch texts and see encouraging results of 63.1% when classifying raw Dutch texts. We attempt to optimise the performance by employing a machine translation pre-processing step, boosting performance of the Afrikaans system on Dutch data to 67.2%. Further investigation is required as we conclude that the robustness of the Afrikaans genre classification system needs improvement.**

Keywords – cross-lingual; genre classification; resource scarce languages; closely related languages; Afrikaans; Dutch

I. INTRODUCTION

When working with the indigenous South-African languages, one is always faced with resource scarcity. In [4] we describe the automatic classification of genre in a resource scarce environment, where experiments were done for six of the indigenous South-African languages. We concluded that the sparseness of available training data, data due to the resource scarceness of the languages in question, causes erratic results (due to overfitting) when using machine learning techniques to classify the genre of a text and that techniques to alleviate these symptoms should be investigated [4]. Therefore, this article investigates the application of technology recycling for the use in genre classification systems.

By adapting existing technologies for closely related languages, the development of resources for resource-scarce languages can be fast-tracked. This process is known as technology recycling [1]. Given a technology, created for a well sourced language *L1*, which is needed in another language *L2* which is resource-scarce, it would be faster and cheaper to adapt the *L1* technology for *L2* than to redevelop the *L2* technology from the ground up [1].

We investigate the effect of the language differences on genre classification and investigate methods by which existing technologies for a well resourced language could be leveraged for a resource-scarce language. We evaluate a genre classification system when classifying a strange language and then implement approaches to enhance its performance. Dutch and Afrikaans have been used successfully in technology recycling experiments because these two languages are similar enough [1][5] and as a result thereof, Dutch and Afrikaans will be used as the languages in question for this article.

We first give an overview of related research pertaining to cross-lingual genre classification in Section 2, after which we describe the experimental setup in Section 3 and the results of the experiments are shown and discussed in Section 4. We conclude this article in Section 5 and we give view to future work.

II. RELATED WORK

Relatively little research is available for the evaluation of a genre classification system that is based on one language, on data that is written in another language. The first research on actual "Cross-Lingual Genre Classification" was made available by Petrenz [2] although cross-lingual methods have been explored for other text classification tasks (other than genre, that is) as will be described later on. Petrenz [2] states that a lot of research aims to develop language independent approaches to text classification rather than cross-lingual approaches, but are seldom able to give definitive empirical proof that these approaches are actually viable. As an example, Petrenz [2] recalls one of the few research reports on genre classification on more than one language (English and Russian) done by Sharoff [3],[3] which suggests that encoding part of speech (POS) data and combining that with variation of common words as feature sets, will be a viable language independent approach. According to Petrenz [2], the claim of this approach being a language independent one is false as the construction of these features are based on the language they are constructed for, although constructed in the same manner for each language. Language neutrality of said approaches can thus be described as a "holy grail"-type pursuit as language specific information will always be implicitly included when constructing these kinds of feature sets. The experiments are also conducted on a per language basis, i.e. the English genre classification system is only evaluated on unseen English data and the Russian system on Russian data so "real" language independent performance is not evaluated. Petrenz [2] chooses to call these features, "stable" features for cross-lingual experiments as they are easily extracted for any language without any prior language knowledge or expertise and do not rely on existing technologies like POS-taggers. How can cross lingual genre classification then be done, if capable language

independent approaches for direct cross-lingual classification do not exist?

To bridge the gap between languages, cross-lingual methods often rely on target language adaptation [2]. Target language adaptation can be achieved by making use of techniques like syntactic reordering [1], morphological adaptations [1], lexical transfer [1] and full- or partial translation [6] to name but a few. Translation is the method which is favoured by Bel *et al*. [6] and by being one of the earliest reports on cross-lingual text classification (for English and Spanish), has set the tone for subsequent research to follow and has had a great influence on the direction that cross-lingual text classification experiments have taken [2], i.e. using machine translation as a pre-processing step when classifying another language. Bel *et al*. [6] state that, when attempting to classify an *L2* text with a *L1* classifier, the discrepancy between the source and target language vocabularies causes incompatibility between the classifier model and the test cases, resulting in very low classifier performance. This discrepancy can be (at least partially) solved with translation by using one of the following translation strategies [6]:

- Terminology translation: Terminology lists are compiled in the classifier language on a per class basis and only the terms which are deemed relevant (by some or other measure, e.g. information gain) to the classification of the specific class are translated in the target language (*L1*).
- Profile based translation: Only the words that occur in the training data for the classifier are translated in the target language (*L1*).
- Full text translation: The entire text is translated in the target language.

Bel *et al*. [6] however criticise the full text translation approach due to the high financial costs and time consuming nature of translation and the questionable translations rendered by machine translation. Petrenz [2] however reports good results on full text evaluations with machine translation systems, as the target language only has to be adapted and does not need to be translated in its entirety. This also compares to the findings of Pilon *et al*. [1] where simple lexical conversion is used in the same manner for POS-tagging experiments with Afrikaans and Dutch, yielding good results. Machine translation should therefore be more than sufficient to bridge the gap in vocabularies for the purpose of this research.

A prerequisite for cross-lingual genre classification using machine translation is that there is a certain set of minimum resources that have to be available for both *L1* and *L2*:

- An *L1* text classification system (i.e. a classifier model trained with genre-specific information);
- A compatible L2 test corpus (i.e. a corpus that is annotated with the same genre specific information as the *L1* classifier model, or which has genre annotations which can be adapted to match *L1*); and
- A machine translation or similar system for target language adaptation.

The next section describes the experimental setup for testing the abovementioned combination of resources for cross-lingual genre classification.

III. Experimental Setup

A. Genre classification system

For the purposes of this article we will use the Afrikaans genre classification system based on the Multinomial Naive Bayes (MNB) algorithm as described in [4] to classify previously unseen Dutch texts according to their genre. The roles of the two languages for traditional technology recycling experiments are reversed in such a way that Afrikaans acts as the well resourced language and Dutch acts as the resource scarce counterpart. This is because a Dutch genre classification system that matches the scope of the Afrikaans classifier could not be found to be used experimentally. A genre classification system with competitive results is already readily available and because Dutch corpora are generally genre annotated in some way and it would be easier to map the genre annotations to the Afrikaans classifier. Petrenz [2] shows the results for cross-lingual genre classification experiments for Spanish and English. From the results reported for these two languages, it can be seen that the directionality of such experiments do not affect the outcome thereof as the reported results for both directions are quite similar.

WEKA [10] is a suite of machine learning algorithms offered as an experimental environment. It holds the benefit of providing access to pre-processing scripts for text to vector conversion with a range of feature extraction options. The Dutch data pre-processing will be done in WEKA as well as the evaluation of the Afrikaans genre classification system, classifying Dutch data.

B. Data

The Afrikaans genre classifier is based on texts that have been extracted from public domain government websites as described in [4]. The classes for the genre classification system mentioned in [4] have been compacted to three classes in order to deal with the sparseness of class representations due to resource scarcity discussed in [4]. Afrikaans showed a good coverage of all the previous classes but for compatibility with the other indigenous languages in planned future work, the shift to a three class genre classification scheme will be used with Afrikaans already.

Class name	**# Training instances**
Expressive	229
Appellative	439
Informative	536
Total	1204

Table 1. Genre classes and instances per class: Afrikaans

These three classes have been adopted from Wachsmuth and Bujna [8] which identify the three classes as follows:

- Personal (expressive). Text that aims to express the personal attitude of an individual towards a product of interest.
- Commercial (appellative). Text that follows commercial purposes with respect to a product of interest.
- Informational (informative). Text that reports on a product of interest in an objective and journalistic manner.

The resulting Afrikaans training data is composed as shown in Table 1. The number of available training instances for each class differs, but the best results for the Afrikaans genre classifier are seen when using all of the available data, when compared to balancing the classes. The best results noted for the Afrikaans genre classifier, based on cross validation experiments, are a precision of 0.931, a recall of 0.930 and a resulting *f*-score of 0.929.

For the Dutch test corpus an excerpt from the original LASSY corpus [7] is used. An official extract from the corpus which is known as LASSY Small is a million word corpus, annotated with syntactic information, as well as POS-tags and lemmas. Genre annotations are also present, but are a little harder to come by. The genre annotations are not explicitly mentioned in the corpus or corpus meta data, but there is mention of the genres in LASSY in the project documentation[1]. The genre classes can be identified by matching the classes mentioned in the documentation to the file names of the corpus' .xml files. The corresponding files are then mapped to the abovementioned genre classes. The initial composition for the Dutch testing corpus is shown in Table 2. There are some of the LASSY corpus files for which a genre could not be identified from the corpus documentation and these files were therefore excluded when compiling the Dutch test instances.

Class name	**# Training instances**
Expressive	75
Appellative	546
Informative	107
Total	728

Table 2. *Genre classes and instances per class: Dutch*

The abovementioned datasets will be encoded in standard binary word occurrence vectors, also known as a bag of words approach (BOW). BOW is one of the stable features for cross-lingual genre classification as described by Petrenz [2].

C. Machine Translation System

For the machine translation component the "Dutch to Afrikaans Converter" (D2AC) by Van Huyssteen and Pilon [5] will be used. D2AC is a rule-based machine translation system based on the orthographic, morphosyntactic and lexical differences between Afrikaans and Dutch. D2AC is not a complete machine translation system as it only applies lexical transfer because it was developed with technology recycling as motivation. They report a precision of 71% for word-level evaluation and a BLEU score of 0.2519 for D2AC [5]. The experiments will be repeated with the Dutch-Afrikaans Google Translate (GT) as machine translation system to verify the results obtained for D2AC

D. Evaluation

The evaluation method used is *n*-fold cross validation (*n*=10), with 90% of the data used for training and 10% of the data used for testing. The standard information retrieval measures, Precision, Recall and F-measure are used to evaluate the effectiveness of classification for the system [9].

Class C_i		**Actual Class**	
		Yes	No
Classifier class	Yes	TP	FP
	No	FN	TN

Table 3. Standard information retrieval methods[9]

The formulas for Recall, Precision, and F-Measure of Ci (see Table 3) are shown in the following three equations (1)(2)(3), Where TP = True Positive, TN = True Negative, FN = False Negative and FP = False Positive classifications.

$$\textbf{R (Recall)} = \textbf{TPi / TPi + FNi}, \quad (1)$$

$$\textbf{P (Precision)} = \textbf{TPi / TPi + FPi}, \quad (2)$$

$$\boldsymbol{f_1}\ \textbf{(}\boldsymbol{f}\textbf{-Measure) = 2(R*P) / (R+P)} \quad (3)$$

E. Baseline System

As a baseline for the experiments a random class baseline (representing a one out of three chance of guessing the correct class) is used. This would result in a 33.33% chance of choosing the correct class. This does, however, not reflect the class distributions. When taking into account the difference in the training instances available to each class, the random baseline can be adjusted to 36.7%. A most frequent class baseline of 44.52% (obtained by dividing the number of instances for the most frequent class by the total number of instances. i.e. always selecting the "Informative" class) is also used.

In the next section the results for the following set of experiments will be discussed:

- Classifying unseen Dutch instances with an Afrikaans genre classifier;
- Translating the Dutch instances to Afrikaans with D2AC and GT and reclassifying the now Afrikaans(-like) instances; and
- Compare the results of these two experiments with each other and with the baselines set above.

[1] http://www.let.rug.nl/~vannoord/Lassy/deliverable1-1.pdf

IV. RESULTS

A. Classifying Dutch instances with an Afrikaans genre classifier

When classifying the unseen Dutch test instances (where the genre annotations extracted from the LASSY project documentation) with the Afrikaans genre classification system, we see some rather disappointing results where the classification precision of 42.3% (Table 4) exceeds the random baseline of 36.7%, but doesn't exceed the most frequent class baseline of 44.52%. But, Bel *et al.* [6] states a precision of 10.75% when evaluating English and Spanish in a pure cross-lingual text classification situation, which puts the performance of pure cross lingual systems in some perspective. They attribute the overlap in the two languages causing the 10.75% precision, to proper nouns and acronyms which are shared between the training and test sets. One would, however, expect a much larger overlap between languages which are said to be closely related, and would therefore expect a somewhat higher score, taking into account we already see an improvement of 28.45% over the English-Spanish results. When translating the Dutch to Afrikaans (as in the next section) only a 3.1% increase in precision was noted. This however didn't hold to Bel *et al.*'s [6] findings of accuracies ranging from 53.8% to 84.5% for translated cross-lingual classifications. These discrepancies prompted a review of all the variables which have an impact on the results.

Language	Precision	Recall	*f*-Measure
Dutch	0.392	0.281	0.277

Table 4. Initial results for Afrikaans classifier and Dutch data

When taking a closer look at the Dutch texts, it was noted that some of the texts which were annotated with the extracted genre classes, weren't supposed to be annotated as such. It was noted that the classes were very noisy and it would need to be remapped to ensure the class representations were indeed representative of the said class. When the genre classes were extracted from the LASSY documentation, there was no indication of how the classes in LASSY were defined, seeing as the genre annotations for LASSY aren't an explicit part of the corpus, it wouldn't be needed to include this kind of descriptions. It is suspected that the interpretation of what a specific genre class constituted differed from what the class in LASSY actually constituted. The Dutch training set was therefore reclassified by hand, making sure the instances were attributed to the correct class. The reclassified test set is presented in Table 5

Class name	# Training instances
Expressive	321
Appellative	391
Informative	16
Total	728

Table 5. Genre classes and instances per class: Dutch reclassified

The cross-lingual Dutch-Afrikaans experiment was repeated, this time with encouraging results (see Table 6). We now see a precision of 63.1%, which exceeds both the random baseline of 36.7% as well as the most frequent class baseline of 44.52% and also satisfies Bel *et al.*'s [6] findings for translated cross-lingual classification, even without being translated yet. In the following section, the results for the translated cross-lingual classification are presented.

Language	Precision	Recall	*f*-Measure
Dutch	0.631	0.284	0.318

Table 6. *Results for Afrikaans classifier and reclassified Dutch data*

B. Classifying translated Dutch instances with an Afrikaans genre classifier

When translating the data with both D2AC and GT we see an increase in the performance, which is above the baselines that were set and even further approximates the highest result of 84.5% as reported by Bel *et al.*'s [6] for translated cross-lingual experiments. The results are shown in Table 7.

Language	Precision	Recall	*f*-Measure
D2AC: Dutch	0.660	0.385	0.438
GT: Dutch	0. 672	0. 429	0. 485

Table 7. Results for Afrikaans classifier and translated Dutch data

Table 8 shows the confusion matrix for the best results seen in Table **7**, i.e. the Dutch test set, translated with GT and classifier with the Afrikaans genre classification system. The classes seem to be confused across the board with the highest confusion noted between Expressive texts being classified as Appellative and Informative texts being classified as Expressive texts. This could be due to erroneous translations or the choice of words for a translation which could be non-prototypical of the class representation of the classifier, which could lead to a misclassification.

		Classified class		
		a	**b**	**c**
Actual class	**a = Appellative**	324	45	22
	b = Expressive	173	50	98
	c = Informative	4	8	4

Table 8.Confusion matrix for GT: Dutch and Afrikaans classifier

The gain in precision which is seen from translating the text is still substantially lower than the gain seen by Bel *et al.* [6]. The results obtained for D2AC and GT seem to be consistent, with only a small variation in the performance being noted. One possible explanation for this occurrence could once again be found in the differences and similarities of the vocabularies of the Afrikaans training data and the Dutch test instances. Using WEKA [10] the words were analysed to ascertain their contribution to classification or in other words, how informative each word is with respect to the classification task. This was done by ranking the words according to their information gain (IG). The top 10 Dutch

words with the highest IG are listed in *Table 9*. These words all have counterparts in Afrikaans which effectively means that these words do not have to be translated because they exist in both vocabularies. The translation of the text therefore only improves the vocabulary compatibility on words which do not contribute very much to the classification and because of that, the gain seen when translating the text before classification is minimal.

Dutch	IG
nog	0.302910
is	0.295780
maar	0.291490
dit	0.283370
die	0.280110
van	0.244970
al	0.239600
dat	0.239070
was	0.235000
wat	0.228150

Table 9. Information gain of Dutch words

V. Conclusion and Future Work

In this article we investigated the application of an Afrikaans genre classification system on Dutch data. We reported on a precision of 63.1% on the aforementioned. We then experimented with machine translations of the Dutch data as a pre-processing step, by using a Dutch to Afrikaans lexical convertor (D2AC) and the Dutch-Afrikaans Google translate, obtaining accuracies of 66% and 67.2% respectively. This kind of technology recycling could be used to help in bootstrapping training data for an under-resourced language, but to be used as a core technology in real world systems, further development is needed to improve the performance. Machine translation systems for some of the indigenous languages have already been developed in the Autsumato project [11] and further development is taking place to further the development for more of the indigenous languages. As these resources become available, the approach described in this research could be tested for these languages. We note however that there are a range of problems that arose from applying cross-lingual genre classification between Afrikaans and Dutch. The compatibility of the training set in the well resourced language and the test set in the underrepresented language is of cardinal importance. By ensuring compatibility for the Afrikaans and Dutch data sets, we noted an increase in performance of 26.8%. The Dutch data was classified by hand which translates to a time consuming, as well as costly process. We therefore propose further research in the compatibility of genre classified corpora, with special regard to automatic methods.

We noted only a small improvement of the performance when using machine translation as a pre-processing step which seems to be contrary to the findings of Bel *et al.* [6] and Petrenz [2]. The reason why only a small increase in performance was seen was noted to be due to an overlap in the vocabularies of Dutch and Afrikaans. This however should intuitively mean a better compatibility but seems to hamper the possibility for growth, rather than improve it. This brings the robustness of the genre classification system into question. Most of the words that overlap are function words that do not necessarily contribute to knowledge about a specific class and is falsely deemed informative. We would like to investigate the use of stop word lists (i.e. lists of words to exclude from training data) and other approaches in an attempt to improve the robustness of the system and eliminated the system's reliance on falsely informative features. Experiments with other machine learning approaches (like support vector machines) could also be performed to determine the suitability of MNB for this task. Initial experiments could also be performed for indigenous language pairs, implementing human translations (where machine translation is not yet available) and one of the less intensive translation strategies as mentioned in Section II.

VI. Acknowledgements

All fallacies remain our own.

References

[1] Pilon, S., Van Huyssteen, G.B., Augustinus, L., "Converting Afrikaans to Dutch for Technology Recycling," in *Proceedings of the 21st Annual Symposium of the Pattern Recognition Association of South Africa*, Stellenbosch, South Africa, pp. 219–224 , 2010.

[2] Petrenz, P., "Cross-Lingual Genre Classification," in *Proceedings of the EACL 2012 Student Research Workshop*, Avignon, France, pp. 11-21, 2012.

[3] Sharoff, S., "Classifying web corpora into domain and genre using automatic feature identification," in *Proceedings of Web as Corpus Workshop*, Louvain-la-Neuve, 2007.

[4] Snyman, D.P., Van Huyssteen G.B., Daelemans, W. "Automatic Genre Classification for Resource Scarce Languages". in *Proceedings of the 22nd Annual Symposium of the Pattern Recognition Association of South Africa*, Vanderbijlpark, South Africa, pp. 132-137, 2011.

[5] Van Huyssteen, G.B. & Pilon, S., "Rule-based conversion of closely-related languages: A Dutch to Afrikaans convertor", in *Proceedings of the 20th Annual Symposium of the Pattern Recognition Association of South Africa*, Stellenbosch, South Africa, pp. 23-28, 2009.

[6] Bel, N., Koster, C., Villegas, M., "Cross-lingual text categorization," Research and Advanced Technology for Digital Libraries, 2769, Lecture Notes in Computer Science, Springer Berlin/Heidelberg, pp. 126–139, 2003.

[7] Van Noord, G., "Huge Parsed Corpora in LASSY," in *Proceedings of the Seventh International Workshop on Treebanks and Linguistic Theories (TLT 7)*, Utrecht, The Netherlands, pp. 115–126, 2009.

[8] Wachsmuth, B., Bunja, K., "Back to the roots of Genres: Text Classification by Language Function," in *Proceedings of the 5th International Joint Conference on Natural Language Processing*, Chiang Mai, Thailand, pp. 632–640, 2011.

[9] Yi-Hsing, C., Hsiu-Yi, H., "An Automatic Document Classifier System based on Naïve Bayes Classifier and Ontology," *in Proceedings of the Seventh International Conference on Machine Learning and Cybernetics*. Kunming, 2008.

[10] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H., "The WEKA Data Mining Software: An Update," SIGKDD Explorations, 11,1, 2009.

[11] Groenewald, H.J., Du Plooy, L., "Processing Parallel Text Corpora for Three South African Language Pairs in the Autshumato Project," AfLaT, pp. 27-30, 2010.

Towards Lecture Transcription in Resource-Scarce Environments

Pieter de Villiers, Petri Jooste, Charl J. van Heerden, Etienne Barnard

Multilingual Speech Technologies Group
North-West University
Vanderbijlpark 1900, South Africa
Email: {pieterdevill, petri.jooste, cvheerden, etienne.barnard}@gmail.com

***Abstract*—We present progress towards automated Lecture Transcription (LT) in resource scarce environments. Our development has focused on the transcription of lectures in Afrikaans from two faculties at North-West University. A bootstrapping procedure is followed to filter and select well-aligned segments of speech. These segments are then used to train acoustic models. Initial work towards language modeling for LT in a resource-scarce environment is also presented; manual lecture transcriptions are combined with text mined from other sources such as study guides to train language models. Interpolation results indicate that study guides are a useful resource for language modeling, whereas general text (obtained from a publisher of Afrikaans books) is less useful in this context. Our findings are confirmed by the reduced word error rates (WERs) obtained from our off-line speech-recognition system for Lecture Transcription.**

***Index Terms*—Lecture Transcription, Afrikaans, Kaldi, Dynamic Programming, Language Model, Resource-scarce.**

I. Introduction

The availability of lecture transcriptions is understood to be very rewarding – most obviously for students with hearing disabilities, but also for the larger student population. Students with hearing disabilities use these transcriptions as a supportive learning medium, while students without such disabilities use them to better understand the lecturer or to supplement their class notes [1]. The multilingual environment of countries such as South Africa offers additional motivation for the development of lecture transcriptions, since students often attend lectures in languages other than their first language, and can therefore obtain significant benefit from transcriptions (either in real-time or off-line).

Hence, Kawahara et al. [2] report that some universities use student volunteers to create notes of classes, since professional stenographers are too costly. However, real-time transcription of lectures is infeasible for humans, and it was found that with 2 volunteers only 20-30% of the spoken lecture utterances could be transcribed in real-time. Another drawback is that these volunteers have to be familiar with the field of the lecture to be able to recognize domain-specific technical words. As a consequence, automated systems for lecture transcription, even with limited accuracy and topic coverage, hold great promise in multilingual universities.

Previous work on lecture transcription for Afrikaans [3] focused on different approaches to alignment, in order to harvest enough data from approximately transcribed lectures to retrain acoustic models using both a well trained target-language (Afrikaans) acoustic model as well as an acoustic model from another language. It was found that the target-language acoustic model performs significantly better for this task.

Given these results, as well as the availability of audio data collection applications such as Woefzela [4] and smart phones, we do not consider obtaining sufficient target-language audio for acoustic modeling as big an obstacle as it was in the recent past,[1] although the optimal approach to combining general and speaker-specific audio data in this context remains an interesting topic for investigation.

The current main challenge therefore with lecture transcription systems in resource-scarce environments is language modeling: lecturers tend to use domain specific words, spontaneous speech containing many false starts, hesitations, filled pauses, non-lexical artifacts such as coughs and laughs, and many other phenomena present in daily human communication [1], [5]. All these affect the accuracy of the speech recognition system. This is even more challenging in resource-scarce environments where very little text data is typically available for accurately modeling these artifacts with language models.

In this paper, we present results from our Afrikaans Lecture Transcription system. Our acoustic modeling approach is described in Section III-C2, and is similar to the approach described in [3], relying heavily on the Dynamic Programming-based audio harvesting procedure described in [6]. We employ a significantly expanded Afrikaans Lecture Transcription (ALT) corpus compared to that in [3], however, enabling us to work with a larger corpus, experiment more thoroughly with speaker-adaptive training, language modeling and also perform actual lecture transcription.

In Section III-F2, we present initial promising results when interpolating language models trained on text resources one can typically expect to exist even in resource-scarce environments: a small amount of transcribed lecture text and a much larger collection of text obtained from study guides. The effect

[1]Woefzela is a freely available Android application and can be used as a medium for collecting data in typical developing-world environments. It provides the user with a reliable and cost effective way of collecting target-language data even in remote locations.

of speaker adaptive training is investigated in Section III-E.

II. BACKGROUND

Various lecture transcription systems, such as the MIT Spoken Lecture Processing project [5], have been implemented in well-resourced environments. For that system, the developers had collected over 500 hours of recordings, of which over 200 hours had been transcribed. For the purposes of speaker adaptation, that corpus contained between 1 to 30 hours of speech per speaker, and the language models were trained on more than 6 million English words.

According to Munteanu et al. [7], current lecture transcription systems obtain word error rates (WER) between 40% and 45% whilst a minimum WER of 25% is acceptable by users. Even though recognition accuracies as high as 98% have been reported in certain Automatic Speech Recognition (ASR) systems, such high accuracies invariably require extremely favorable conditions, such as reading selected materials (from a limited context) aloud [1].

Glass et al. [5] found a greater improvement in WER from acoustic modeling than language modeling. However, they found that performing acoustic modeling on four 50 minute lectures from a single lecturer, while also performing language model adaptation using two related textbooks and 40 related lectures, still resulted in a high WER (30.7%). Using 29 hours of previous lectures for acoustic modeling decreased the WER noticeably(17%). Similar results where found by Trancoso et al. [8].

Unsupervised training is currently also receiving significant attention. Here, term discovery algorithms are used to identify words or phrases of different speakers and genders by identifying repetitions in the data. Jansen and Church [9] demonstrated that unsupervised training of acoustic models is possible with strong speaker independent properties.

III. APPROACH

Throughout our experiments we made use of two speech corpora: the NCHLT corpus [4] and the Afrikaans Lecture Transcription corpus, which was developed to support the current research.

A. NCHLT corpus

The NCHLT corpus consists of speech from 206 Afrikaans speakers (approximately equal numbers of males and females), with approximately 500 3-5 word utterances of read speech per speaker, recorded in a controlled environment. This amounts to approximately 100 hours of speech data. The vocabulary of this corpus consists of 9375 distinct words, drawn from a variety of subjects, as would be appropriate (for example) for a Web-search application.

B. Afrikaans Lecture Transcription corpus

The Afrikaans Lecture Transcription (ALT) Corpus consists of 20 hours of Afrikaans lecture data from two broad subject areas; law and science/chemistry. Male lecturers account for 14 hours of speaker data and females 6 hours. All audio data has been manually segmented into 5 minute segments, mainly to increase the speed of the alignment and decoding [3].

A single first-language Afrikaans speaker produced orthographic transcriptions of the ALT corpus; the transcriber was given the following instructions:

- Transcribe exactly what was said (do not correct for grammar, hesitations, etc)
- Use punctuation (,.?!) only to indicate sentence structure (no quotation marks or brackets)
- Write out numbers in words instead of using digits 0-9
- Mark foreign words with #

All speakers are listed in Table I with their associated subjects, gender and amount of training and testing data in minutes. The test set consists of one lecture from each of those lecturers who has multiple lectures in the ALT corpus.

TABLE I
ALT SPEAKER INFORMATION WITH TRAINING AND TESTING DATA IN MINUTES

SPKR ID	Gender	Subject	Train	Test	Total
m001	male	sci	17	0	17
m002	male	sci	42	37	79
m003	male	sci	84	37.5	121.5
m004	male	sci	31	0	31
m005	male	sci	44	0	44
m006	male	sci	46	37	83
m007	male	sci	43	0	43
m008	male	sci	37	0	37
m009	male	law	26	23	49
m010	male	law	36	0	36
m011	male	law	35	35.5	70.5
m012	male	law	62.5	37.5	100
m013	male	law	57.5	0	57.5
m014	male	law	47	0	47
m015	male	law	27	0	27
f001	female	sci	39.5	23	62.5
f002	female	sci	46.5	43	89.5
f003	female	sci	25	0	25
f004	female	law	32.5	30.5	63
f005	female	law	61.5	36	97.5
f006	female	law	40.5	0	40.5

C. Baseline systems for alignment

Four baseline systems were created for the purposes of alignment and subsequent harvesting of well-transcribed portions of the ALT corpus. This was done by employing the iterative DP scoring and filtering technique described in [6]. Before we describe the four systems in more detail, we will first elaborate on the experimental setup followed for the alignment systems.

1) Pronunciation modeling: Pronunciation dictionaries were created for all systems by (1) using a dictionary lookup for known Afrikaans words (443 words), (2) identifying English words with a dictionary lookup (840 words) and (3) using the Default & Refine [10] algorithm to automatically generate pronunciations for the remaining 6735 words.

English words occur fairly frequently in the ALT corpus; they were automatically identified by a dictionary lookup and the pronunciation mapped to similar Afrikaans phones was the same as in [3]. These mappings are shown in II. All names and

foreign words (which were marked with # by the transcriber) were then manually verified.

TABLE II
ENGLISH TO AFRIKAANS PHONE MAPPINGS

Eng	Afr	Eng	Afr
3:	@	Q	O
e@	E	r\	r
ai	a i	tS	t S
au	a u	u:	u
d_0Z	d Z	U	u
i:	i	T	f
O:	O	D	v
Oi	O i		

2) Acoustic modeling: The acoustic models for alignment were trained on 39 dimensional Mel frequency cepstral coefficients (13 static, 13 deltas and 13 double deltas). Offline cepstral mean and variance normalization was applied per speaker (that is, the same normalization constants were applied to all the speech from one speaker, and these constants were computed so that all speakers have the same cepstral means and variances after normalization). The hidden Markov models (HMMs), trained with HTK [11], were standard 3-state left to right tied-state triphone models, with 8 mixtures per state and semi-tied transforms. A garbage model [6] was then trained and combined with the initial model.

The following acoustic models were trained:

- **NCHLT baseline**. An acoustic model was trained on the NCHLT corpus described in Section III-A. This model was trained without a garbage model.
- **ALT (5-minute segments)**. We trained acoustic models using the entire manually segmented ALT corpus. Segments were approximately 5 minutes in duration. This acoustic model resulted in a phone accuracy of 45.14%. Based on our earlier experience with this corpus, this was good starting accuracy for a baseline system; we nevertheless decided to make use of DP scoring to automatically further segment the ALT data into smaller – but more reliable – segments, that could be used for further training.
- **DP filtered ALT**. The ALT corpus was automatically segmented into 10 second or smaller chunks as done by [5]. Our process employed the dynamic-programming phone string alignment procedure used by [3] with a flat phone matrix. As described in [3], this approach compares the result of a forced alignment with that of a free decode using a variable cost matrix, identifies the accurately transcribed sections of audio and use these results to segment the audio as well as the transcriptions. Using this technique we segmented the 5 minute ALT data into small chunks of accurately transcribed data using the NCHLT model. These well-aligned portions were then used to train a new improved ALT model.
- **NCHLT MAP**. The NCHLT model was then also MAP adapted using the entire ALT training set and used to automatically segment the ALT data into small chunks of accurately transcribed data using the dynamic-programming technique.

D. Alignment results

The phone accuracies of these 4 baseline systems, tested on the same ALT data are shown in Table III. Here the reference phone strings were generated by using the pronunciations as described in Section III-C1 since it was infeasible to obtain manual phone transcripts. The improvements in various measures of alignment accuracy (see [6] for motivations) after model refinement are shown in Table IV[2].

TABLE III
PHONE-RECOGNITION ACCURACIES OF BASELINE SYSTEMS TESTED ON ALT

NCHLT	LT(5 min)	LT(DP scoring)	NCHLT(MAP all)
19.28%	45.14%	49.70%	16.50%

As seen in Table III, domain-specific training data is very beneficial for the development of a baseline system. A further significant increase in phone accuracy (nearly 5%) is achieved by segmenting the ALT(5min) training data using the dynamic-programming technique.

TABLE IV
MEASURES OF ALIGNMENT ACCURACY ACHIEVED AFTER MODEL REFINEMENT ON THE TEST SET

Model	Avg DP Score	Log P	Time
NCHLT	-0.176	-52.10	4:05/5:39
NCHLT-MAP/all (ALT)	-0.217	-50.82	3:53/5:39
NCHLT MAP/spk	-0.202	-51.03	3:35/5:39
ALT	0.114	-45.60	3:14/5:39

E. Speaker Adaptation

Speaker adaptation was performed on multiple speakers for which we had data from more than one lecture. One or more lectures were used for speaker adaptation (Train column, Table I), and one lecture was held out for testing purposes (Test column, Table I). The NCHLT corpus acoustic model was also adapted to these speakers to see how important the use of speaker-specific data is to the overall system used for alignment. Table V summarizes the phone accuracies for different speakers on ALT without MAP adaptation, ALT with MAP adaptation, NCHLT without MAP adaptation and the NCHLT model with MAP adaptation. These results were obtained using the same techniques as in [8] where 3 iterations of speaker adaptation is performed using the same adaptation data.

We see that some speakers achieve only small gains in phone accuracy, and reduced accuracies after adaptation are even seen in many cases. These disappointing results are probably a consequence of the small amount of adaptation data

[2]It would have been preferable to make these measurements on a held out development set, but because of data scarcity and since these measures do not influence our decision on which model to use for the final data segmentation, we measured the model refinement on the test set

TABLE V
PHONE ACCURACY PER SPEAKER WITH AND WITHOUT MAP ADAPTATION, USING DIFFERENT BASELINE ACOUSTIC MODELS

SPKR ID	ALT	ALT + MAP	NCHLT	NCHLT + MAP
m002	59.69	62.12	27.48	29.01
m003	66.82	67.42	33.95	38.03
m006	48.84	49.59	12.38	12.20
m009	50.73	52.21	19.48	18.66
m011	59.39	61.92	19.97	18.72
m012	55.29	49.02	18.60	15.94
f001	39.24	39.43	16.99	15.45
f002	39.91	39.04	13.53	13.77
f004	40.93	34.97	17.84	15.68
f005	28.14	28.14	12.29	8.50
Avg	48.9	48.39	19.25	18.6

available to us – thus, the risk of overtraining is significant, and MAP adaptation is not able to compensate for the differences in recording conditions between the two corpora.

F. Baseline systems for lecture transcription

The process described in Section III-C is useful in a resource-scarce environment where only a few hours of lecture transcription data is available. The resulting well-aligned portions of our ALT corpus were used to train state of the art acoustic models for offline lecture transcription.

1) Acoustic modeling: The Kaldi toolkit [12] was used to train our best acoustic models which were used for transcribing lectures. Standard MFCCs with cepstral mean normalization were again used, with LDA, MLLT, speaker adaptive training (MLLR) and boosted mmi.

2) Language modeling: The most basic language model for this task is a simple word trigram model, built from the transcribed lectures. The transcriptions were separated into the two groups (law and natural sciences), and each sub-corpus was used for a specialized language model (in addition to the basic language model covering both topics). Because of the small corpus size of these transcriptions, we wanted to investigate whether recognition accuracy could be improved by using larger language models from a more general source of Afrikaans text. The Puk-Protea-Boekhuis corpus was used for this purpose. It contains Afrikaans text from published works and contains substantial quantities of proofread material on various topics. (In addition to prose and instruction documents, it also includes poetry.) In an effort to improve on the anticipated out-of-vocabulary words in the university lectures, a general set of study guides of the North West University was obtained. Since the available lectures chosen to be transcribed were from two different faculties, law and natural sciences, the study guides of these two faculties were used as corpus for the experiments to be described here.

The following three groups of corpora were therefore used: 1) Transcriptions of lectures, 2) University study guides, 3) General text. The details of these corpora are as follows:

- *Transcriptions of lectures in law (1A-tr-law) and natural sciences (1B-tr-sci).* Only the transcriptions of the training data was used so that the test data for measuring the recognition performance would not be in the language models. These were the smallest corpora 60K and 55K words respectively.
- *Study guides for subjects in law (2A-sg-law) and natural sciences (2B-sg-sci).* Only the Afrikaans versions of the study guides were used. For some subjects the study guides were bilingual combining Afrikaans and English in the same document, but these were not used. After text normalisation, these corpora were 1.4 million and 2 million words respectively.
- *The Puk-Protea-Boekhuis (protea) corpus* was used as a source for general proof-read text. After normalisation described below this corpus consisted of 6 million words.

Several steps were taken when pre-processing these text corpora for building language models. These issues are addressed in the discussion on text normalisation below. Initial experiments with 2-, 3-, 4-, 5-, and 6-grams showed lowest perplexity in most cases with 3-grams. In a few cases 4-grams were marginally better, but the benefits were never sufficiently large to justify the cost of the larger language models. All language models reported below are therefore trigram word models, which were built using the Modified Kneser-Ney smoothing algorithm [13]. In all models, the markers for beginning and end of sentences are included as tokens.

To get a basic measure of perplexity and out-of-vocabulary rate for each language model, the text from the lecture transcription test data was used for testing. The test set was also divided into *law* and *natural sciences* transcriptions; language-modeling results when testing on these two sub-corpora are reported in Table VI.

TABLE VI
PPL (PERPLEXITY) AND OOV (OUT-OF-VOCABULARY) RATE FOR BASELINE LANGUAGE MODELS.

Corpus	#2-grams	#3-grams	Test set	PPL	OOV rate
1A-tr-law	28725	5524	law	171.94	7.92%
1B-tr-sci	25285	5169	law	160.08	16.21%
2A-sg-law	255162	117280	law	404.96	7.01%
2B-sg-sci	421448	231741	law	647.88	9.33%
protea	1261554	446814	law	443.23	6.20%
1A-tr-law	28725	5524	sci	174.36	15.39%
1B-tr-sci	25285	5169	sci	151.53	7.54%
2A-sg-law	255162	117280	sci	664.32	15.22%
2B-sg-sci	421448	231741	sci	673.49	6.84%
protea	1261554	446814	sci	498.24	8.86%

As expected, the in-domain transcriptions provide the best match for the respective test sets, with relatively low perplexities and OOV rates. Interestingly, the cross-domain perplexities are comparable to the in-domain perplexities, with *tr-sci* actually achieving a somewhat lower perplexity on the *law* test set (although at a substantially higher OOV rate). Also, the OOV rates from the study guides and *Protea* corpus are encouragingly low, suggesting that corpus combination may be a profitable strategy. However, the disparate sizes of the corpora indicates that combination through weighted interpolation (rather than pooled resources) should be the strategy of choice; we therefore investigated the characteristics

of various interpolated language models involving these five sub-corpora.

3) Interpolated language models: To investigate the potential benefits of language-model interpolation, we created test sets from each of the five sub-corpora mentioned above. Modified Kneser-Ney smoothing was employed to estimate language models based on the training sets extracted from the same corpora, and the SRILM toolkit [14] was then used to find the optimal interpolation weights for combining these language models.

Because these various corpora have widely different characteristics, it is not meaningful to compare the perplexities and OOV rates across different test sets. We therefore focus on the interpolation weights that yield the lowest perplexity for each test set, as shown in Table VII.

TABLE VII
INTERPOLATION WEIGHTS THAT MINIMIZE THE PERPLEXITIES ON FIVE SUB-CORPORA.

Test Corpus	Weight: 1A-tr-law	Weight: 1B-tr-sci	Weight: 2A-sg-law	Weight: 2B-sg-sci	Weight: Protea
1A-tr-law	0.415	0.137	0.266	0.016	0.165
1B-tr-sci	0.074	0.606	0.004	0.170	0.145
2A-sg-law	0.001	0.000	0.933	0.042	0.024
2B-sg-sci	0.000	0.000	0.024	0.960	0.015
Protea	0.003	0.002	0.008	0.007	0.979

As expected, the diagonal entries in Table VII dominate; that is, the training set of each corpus makes the largest contribution to the lowest-perplexity language models of the corresponding test set. However, for the two LT test sets (the first two rows of the table), the other corpora also contribute substantially to the optimal language models. Interestingly, it is the study guide from the same domain as the test set which makes the largest contribution in each case; this is evidence that the non-transcription corpora are contributing to these language models in a predictable way.

4) Recognition results with interpolated corpora: Based on the analysis in Section III-F2 and Section III-F3, it was decided to use the transcriptions of lectures (LT) and University study guides (SG) to train language models to be used for off-line lecture transcription. Our goal here is to investigate whether improved recognition performance can be achieved with language-model interpolation, and to simplify the presentation we focus on the within-topic interpolation of the study guides and transcriptions. That is, we build two sets of language models, one for the *law* domain and the other for the *sciences* domain. In each set, we investigate the effect of ranging the interpolation weight between 0.0 (at which value the study guides dominate) to 1.0 (where the model is dominated by the transcriptions). For consistency, we report all results at a LM weight of 14, which is a reasonable value for our configuration, but not optimized for a particular language model.

As Figures 1, 2 and 3 show, we find in all cases, and for both the *law* and *sciences* test sets, that optimal performance is achieved at an interpolation value somewhere between the extremes, thus showing that language-model interpolation

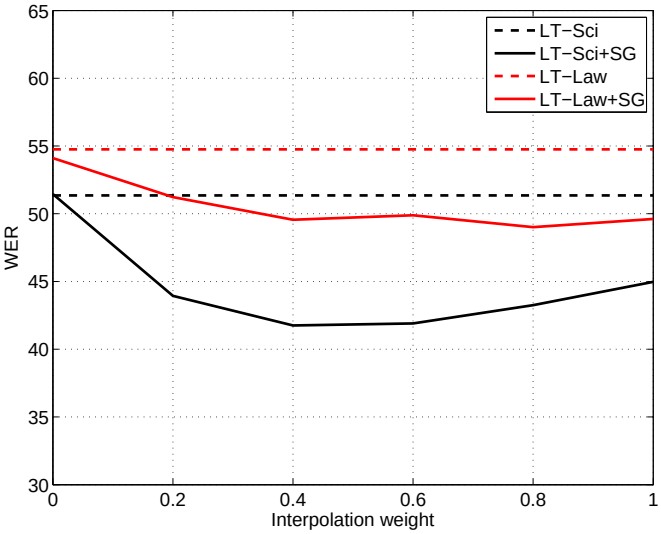


Fig. 1. WER for off-line lecture transcription when trained on *sci* and *law* sources respectively and evaluated on the combined *sci* and *law* LT test set. The dotted lines correspond to language models trained only on the LT training data, and the solid lines represent interpolated results, with the interpolation weight on the horizontal axis.

indeed is beneficial in all cases. As could be expected, the in-domain language models perform best on both test sets; these differences in word error rates are quite large, confirming the importance of language modeling for this task.

Note that our approach to interpolation requires a fixed vocabulary for all settings of the interpolation weight. Therefore, all words from both training sets are included in all interpolated models, albeit with only the unigram back-off probabilities in some cases. In Figures 1, 2 and 3 we can see that even these unigrams make a useful contribution to recognition accuracy, since the WERs with only the lecture-transcription language models (dotted lines in Figs.1, 2 and 3) are notably higher than the corresponding interpolated models (right-most points of the solid lines).

IV. CONCLUSION

Whereas the use of target-language acoustic data has previously shown to be beneficial [3], we have additionally demonstrated that domain-specific training data significantly contributes to the accuracy of lecture transcription systems. This is true even if the amount of available in-domain data is severely limited. However, under these constraints, the additional value of speaker adaptation is minimal.

We have also shown that additional target-language text, such as study guides, can lead to a substantial reduction in word error rates. Since such sources are likely to be available in the type of educational environment which is expected to represent the most important use case of this technology, this is a practically important result.

The error rates that we have achieved are still somewhat higher than those that are considered usable in lecture-transcription applications [7]; hence, the need for further

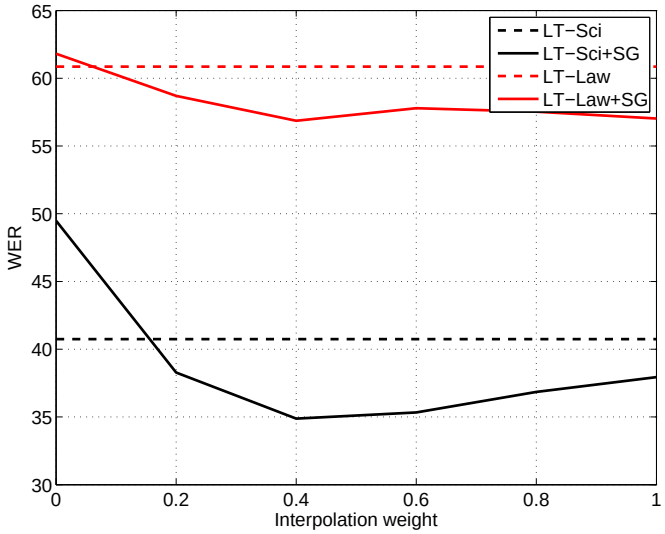


Fig. 2. WER for off-line lecture transcription when trained on *sci* and *law* sources respectively and evaluated on the *sci* LT test set. The dotted lines correspond to language models trained only on the LT training data, and the solid lines represent interpolated results, with the interpolation weight on the horizontal axis.

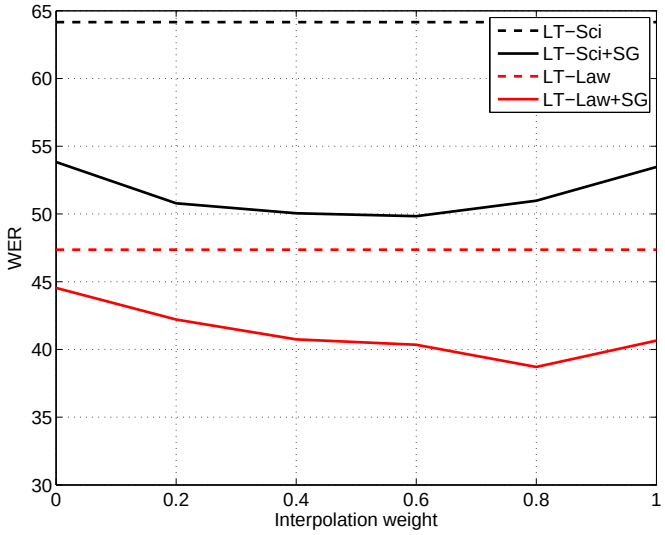


Fig. 3. WER for off-line lecture transcription when trained on *sci* and *law* sources respectively and evaluated on the *law* LT test set. The dotted lines correspond to language models trained only on the LT training data, and the solid lines represent interpolated results, with the interpolation weight on the horizontal axis.

improvement is clear. The most likely sources of such improvement are methods that use the limited acoustic and textual information more efficiently; we therefore believe that the development of such methods should be a priority for further research.

V. Acknowledgment

We would like to thank the NRF (National Research Foundation) for the bursary funding provided throughout the year.

References

[1] K. Bain, S. H. Basson, and M. Wald, "Speech Recognition in University Classrooms : Liberated Learning Project," in *Proceedings of the fifth international ACM conference on Assistive technologies - Assets '02*. New York, New York, USA: ACM Press, 2002, pp. 192–196.

[2] T. Kawahara, "Automatic transcription of parliamentary meetings and classroom lectures - A sustainable approach and real system evaluations -," in *7th International Symposium on Chinese Spoken Language Processing (ISCSLP*, 2010, pp. 1–6.

[3] C. J. van Heerden, P. de Villiers, E. Barnard, and M. H. Davel, "Processing Spoken Lectures in Resource-Scarce Environments," in *PRASA2011 - Proceedings of the 22nd Annual Symposium of the Pattern Recognition Association of South Africa*, P. Robinson and A. Nel, Eds., Vanderbijlpark, South Africa, 2011, pp. 138–143.

[4] N. J. de Vries, J. Badenhorst, M. H. Davel, E. Barnard, and A. de Waal, "Woefzela - An open-source platform for ASR data collection in the developing world," in *Proceedings Interspeech*, Florence, Italy, August 2011, pp. 3176–3179.

[5] J. Glass, T. J. Hazen, S. Cyphers, I. Malioutov, D. Huynh, and R. Barzilay, "Recent Progress in the MIT Spoken Lecture Processing Project," in *Interspeech 2007 (8th Annual Conference of the International Speech Communication Association)*. Antwerp, Belgium: ISCA, 2007, pp. 2553–2556.

[6] M. H. Davel, C. van Heerden, N. Kleynhans, and E. Barnard, "Efficient harvesting of Internet audio for resource-scarce ASR," in *Proceedings Interspeech*, Florence, Italy, August 2011, pp. 3153–3156.

[7] C. Munteanu, G. Penn, and R. Baecker, "Web-Based Language Modelling for Automatic Lecture Transcription," in *In Proceedings of the Tenth European Conference on Speech Communication and Technology - EuroSpeech / Eighth INTERSPEECH*, Antwerp, Belgium, 2007, pp. 2353–2356.

[8] I. Trancoso, R. Nunes, H. Moniz, D. Caseiro, and A. I. Mata, "Recognition of Classroom Lectures in European Portuguese," in *INTERSPEECH 2006 - ICSLP (9th International Conference on Spoken Language Processing)*, no. 1. Pittsburgh, PA, USA: ISCA, 2006, pp. 281–284.

[9] A. Jansen and K. Church, "Towards Unsupervised Training of Speaker Independent Acoustic Models," in *Interspeech 2011 (ICSLP 12th Annual Conference of the International Speech Communication Association)*, August 2011, pp. 1693–1692.

[10] M. Davel and E. Barnard, "Pronunciation predication with Default&Refine," *Computer Speech and Language*, vol. 22, pp. 374–393, October 2008.

[11] S. J. Young, G. Evermann, M. J. F. Gales, T. Hain, D. Kershaw, X. Liu, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. C. Woodland, *The HTK Book (for HTK Version 3.4)*, march ed. University of Cambridge, 2009, no. July 2000.

[12] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society, December 2011, iEEE Catalog No.: CFP11SRW-USB.

[13] S. F. Chen and J. Goodman, "An Empirical Study of Smoothing Techniques for Language Modeling," Computer Science Group, Harvard University, Cambridge, Massachusetts, Tech. Rep. TR-10-98, 1998.

[14] A. Stolcke, "SRILM – an extensible language modeling toolkit," in *Proceedings of ICSLP*, vol. 2, Denver, USA, September 2002, pp. 901–904.

Comparing grapheme-based and phoneme-based speech recognition for Afrikaans

Willem D. Basson[1,2]
[1]Human Language Technology Competency Area
CSIR Meraka Institute
[2]Multilingual Speech Technologies
North-West University,
Vanderbijlpark, South Africa
Email: wlmbasson@gmail.com

Marelie H. Davel
Multilingual Speech Technologies
North-West University,
Vanderbijlpark, South Africa
Email: marelie.davel@gmail.com

***Abstract*—This paper compares the recognition accuracy of a phoneme-based automatic speech recognition system with that of a grapheme-based system, using Afrikaans as case study. The first system is developed using a conventional pronunciation dictionary, while the latter system uses the letters of each word directly as the acoustic units to be modelled. We ensure that the pronunciation dictionary we use is highly accurate and then investigate the extent to which ASR performance degrades when the dictionary is removed. We analyse this effect at different data set sizes and classify the causes of performance degradation. With grapheme-based ASR outperforming phoneme-based ASR in certain word categories, we find that relative error rates are highly dependent on word category, which points towards strategies for compensating for grapheme-based inaccuracies.**

I. Introduction

In an automatic speech recognition (ASR) system, words are traditionally represented as a sequence of acoustic sub-word units such as phonemes [1]. The mapping from these sub-word units to words are usually contained in some sort of lexicon, that is, a pronunciation dictionary. The overall performance of ASR systems is strongly dependent on the accuracy of the pronunciation dictionary and best results are usually obtained with hand-crafted dictionaries, which often requires expert knowledge. Development of these dictionaries is a time-consuming, costly and labour-intensive process. If expert knowledge is either unavailable or too costly, manually developed or statistical grapheme-to-phoneme (g2p) rules can be used to generalise from small data sets [1]. However, these methods typically produce less accurate results.

Earlier work in grapheme-based systems has shown that for regular languages – languages that exhibit a close relationship between graphemes and phonemes – phone-based dictionary development may be unnecessary [1], [2], [3]. Using grapheme-based sub-word units eliminates the need for expert knowledge and saves time and cost. Other advantages include simplified lexicon definition and relatively noise-free pronunciation models [4].

The regularity of a language can be measured based on g2p consistency: using the average accuracy that is obtained at a specific dictionary size when extracting g2p rules. According to this measure, languages vary considerably, from highly irregular languages such as English, to highly regular languages such as Flemish, with Afrikaans being somewhere in between [5].

Some of the earliest work done on grapheme-based speech recognition proposes using polygraphs i.e. letter based units constructed from the orthographic word form with arbitrary length left and right contexts as sub-word units [3]. More recent work include context-dependent grapheme-based recognisers [1] as well as using a decision tree based on graphemic acoustic sub-word units together with phonetic questions [2].

For this paper we developed a grapheme-based ASR system alongside a phoneme-based ASR system using the same standardised approach in both, in the one case using tied-state triphones and the other, tied-state trigrams. With the only variable between the systems being their respective pronunciation dictionaries, this allows for a fairly direct comparison of strengths and weaknesses.

The remainder of this paper is structured as follows: Section II describes the approach followed, both to construct the gold standard phonemic dictionary and to compare grapheme-based and phoneme-based performance. The data used is presented in section III. The various experiments are described and results presented in section IV. Finally, the paper is ended by a summary of our main observations in section V.

II. Approach

We develop comparable grapheme-based and phoneme-based ASR systems for different training data sizes ranging from 5 to 40 hours, and compare word error rate (WER) using independent test sets and 4-fold cross validation. For the comparison to be fair, we need to ensure the pronunciation dictionary is as accurate as possible. The most comprehensive Afrikaans dictionary currently available is the *Resources for Closely Related Languages Afrikaans pronunciation dictionary (rcrl_apd)* [6]. This dictionary however does not include all the words in the data set we are modelling. The process to develop and verify a more comprehensive dictionary is of interest and results relating to this process are included in this paper.

A. Pronunciation Dictionaries

We develop 3 different pronunciation dictionaries. Firstly, we develop a manually verified pronunciation dictionary which serves as a gold standard. It should be noted that this dictionary contains pronunciation variants where appropriate. The total effort in verifying all the sub-word units is lessened by utilising methods such as:

- known word extraction: accepting known pronunciations from existing dictionaries;
- decompounding unknown words and matching these to known components in existing dictionaries;
- short word extraction: analysing short words – which are often non-standard words such as abbreviations or acronyms – separately; and
- the classification of word types to be pre-processed by appropriate g2p methods.

All automated methods used to produce pronunciations were manually verified, which allow us to report on the success rates of each of the automated methods. Since Afrikaans contains many compound words, we focused our effort on identifying known compounds from existing dictionaries, using both a form of longest string matching (LSM) and automated morphological decomposition to achieve this aim.

Secondly, the best possible rule set available to date – rules extracted from the *rcrl_apd* pronunciation dictionary [6] – was used to create an automated (state-of-the-art g2p) pronunciation dictionary. Finally, a minimal effort grapheme-based dictionary was developed by simply splitting the orthographical form of words into space-separated single letters.

Given the gold standard dictionary, the relative accuracy of the g2p dictionary is calculated by measuring the difference between pronunciations. Calculating the accuracy of the grapheme-based dictionary is done by converting every grapheme to its default phoneme based on g2p rules and measuring pronunciation similarity relative to the gold standard dictionary. The relationship between differences in dictionaries and resulting WER is investigated.

B. ASR accuracy

ASR systems are analysed and compared in terms of WER. All test sets are recognised using the same flat language model containing all the words in the entire data set. While better recognition accuracy can be obtained using a statistical language model, we specifically want to evaluate the effect of the acoustic models without recognition being guided by a language model. This means that the systems are evaluated and compared in terms of WER with the only difference between systems being their pronunciation dictionaries. (For the later category-based analysis, it is particularly important that categories are not influenced by the language model used.)

C. Error classification

ASR recognition errors are classified according to word type and compared across systems. Word types include (1) abbreviations, (2) acronyms, (3) foreign words, (4) generic Afrikaans words, (5) partial words, (6) proper names, (7) concatenated words, (8) spelling errors, (9) spelled out words, (10) single spelled out characters and (11) unknown words. Word type categories were determined during the development of the manually verified pronunciation dictionary. Words that belong to more than one category (due to pronunciation variants or context) are classified as multi-category words. Pronunciation variation caused all but one abbreviation to be classified as multi-category words.

III. Data selection

Afrikaans was selected as the experimental language due to its g2p regularity (fairly regular without being fully regular) and the authors' inherent familiarity with the language. The dataset used is a subset of the NCHLT corpus [7] and has a total length of approximately 64 and a 1/2 hours, consisting of 75 150 utterances from 167 speakers with a male to female ratio of 48.5/51.5. Every utterance in this dataset passed basic quality control checks namely: clipping detection, volume detection and speech cutting detection [8]. Also, to ensure a well balanced dataset every speaker contributes exactly 450 utterances. From this dataset a development set of approximately 2 hours and 45 minutes was held out. The remaining utterances were split into 4 folds with 4 mutually exclusive test sets. Each fold's train set is roughly 46 hours long and contains 54 000 utterances from 120 different gender balanced speakers. All 4 the training sets were then individually subdivided into 46 total random, non-sequential incremental segments. In effect each segment contains approximately one hour more data than the previous one. Finally, to study the effect of phone-based and grapheme-based ASR on varying sizes of training data, segments 5, 10, 20 and 40 were selected for training.

F	# utt trn	# hr trn	# spkr trn	# utt tst	# hr tst	# spkr tst
1	54000	46:18:56	120	18000	15:25:9	40
2	54000	46:51:34	120	18000	14:52:31	40
3	54000	45:51:57	120	18000	15:52:8	40
4	54000	46:9:50	120	18000	15:34:15	40

TABLE I
Data selection: Number of utterances (utt), hours (hr) of audio data and number of speakers (spkr) in train (trn) and test (tst) sets across folds (F)

F	seg 5	seg 10	seg 20	seg 40
1	05:05:24	10:05:53	20:07:59	40:14:12
2	05:06:05	10:11:15	20:24:25	40:45:14
3	05:02:28	10:00:23	19:55:38	39:53:24
4	05:02:50	10:03:34	20:05:03	40:07:01
# utt	5870	11740	23479	46957

TABLE II
Training segments: Hours of audio data and number of utterances per segment (seg) across folds (F)

IV. Experiments and Results

Experiments relating to the development of the gold standard pronunciation dictionary are described in sections IV-A to IV-C, while sections IV-D and IV-E compare the ASR results obtained using the three different dictionaries (the gold standard phoneme-based dictionary, the g2p-predicted dictionary and the grapheme-based dictionary).

A. Identifying known constituents in compounds

As discussed in section II, we experimented with two different approaches to decompounding. Note that the primary purpose was to lessen the total effort in creating a pronunciation dictionary: not to find linguistic compounds as such, but only to find known constituents from existing dictionaries (i.e. where pronunciations are known.) Since Afrikaans contains many compounds, many words in a word list would be flagged as unknown when measured against existing dictionaries, while the constituents are actually known and pronounced in an identical manner.

In the remainder of this section we describe the two approaches used (variants of Morfessor-based decompounding and Longest String Matching), the post-processing that is required (which is similar for both approaches), and the results achieved.

1) Morfessor: Morphological decomposition was performed using a modified version of Morfessor 1.0 [9], a popular language independent tool for performing unsupervised morphological decomposition. We changed the tool to only use existing words as 'morphemes' and not to create smaller linguistic components, in effect changing it into a decompounding tool. All other settings were left at their default values.

Given as input is a combination of unique words from an existing dictionary and all words with unknown pronunciations, Morfessor then suggests segmentations for all words, based on identified segments that exist as individual words in an existing dictionary. Words that can be segmented are flagged as candidate compounds, new pronunciations are generated based on the pronunciations of the individual words and prepared for review.

2) LSM: An imperfect version of Longest String Matching algorithm similar to that of [10] was used. The difference being that the longest left hand match is performed at the same time as the longest right hand match, possibly causing overlap and missing some compounds. A limited valence morpheme list is used containing only two valance morphemes, namely *s* and *en*. Using a lexicon of known words as a reference, the largest left- and right hand matching strings of each candidate compound is determined. Words are then flagged as possible compounds if: (a) after subtraction of the left and right match, there is no remainder and the length of the compound is equal to the combined length of the largest left and right match, or (b) the remainder of the compound is either a valid word from the lexicon, or (c) the remainder is a valid valence morph from the limited list.

3) Post-processing: After each decompounding method the pronunciations of compound constituents are extracted from existing dictionaries, residual consonant doubling caused by constituent concatenation is removed, and finally, flagged compounds and their accompanying phone strings are manually verified.

4) Results: After verification, we found 1 492 compounds in the data set (containing 3 225 unique words) of which 1 416 had correct pronunciations. A breakdown of our results are shown in Table III. Morfessor decomposition was applied first, then LSM-based decomposition. Note that LSM-based decomposition was only performed on words that Morfessor was not able to decompound, resulting in 179 additional compounds. Since we are not interested in finding linguistically accurate compound boundaries some of the words identified are not actual compounds, yet they still produce correct phone strings. Table IV summarises the effect of decomposition on pronunciation. Most pronunciation errors relate to a few small morphemes ('ver', 'end', 'bes') that were incorrectly predicted as /E/ rather than /@/ (using SAMPA notation).

	Total flagged	Correctly identified	Incorrectly identified
LSM	203	179	24
Morfessor	1419	1313	106

TABLE III
Breakdown of LSM and Morfessor based decomposition showing the number of correctly identified and incorrectly identified compounds

	Pronunciation		
	correct	error	% correct
Correctly decomposed	1 416	76	94.6
Incorrectly decomposed	130	119	8.5

TABLE IV
Effect of decomposition on pronunciations

B. Developing a gold standard dictionary

As described earlier (in section II-A), in order to lessen the total effort of classifying, predicting pronunciations for and verifying 9 375 unique words, we employed various strategies. Initially, all known words from existing dictionaries were extracted: this comprised nearly two thirds of the dictionary. Remaining words were then checked against known word lists and classified as either valid Afrikaans words, valid English words or unknowns words. All valid English words were then removed, their pronunciations predicted with English g2p rules and these were manually verified. The remaining words were then processed concurrently by the two different decompounding methods described in section IV-A1.

Short word extraction was then performed on the remaining words by extracting all words with a length of 1-4 characters. The vast majority of these words fell into the category of spelled out Afrikaans words. High numbers of partials, abbreviations and acronyms were also present. Words were then categorised and pronunciations were generated with appropriate g2p methods after which all words were reviewed manually. A hand made list was crafted for all spelled out single characters. For the remaining 1 351 words pronunciations were predicted and manually verified. All manual verification was performed by two verifiers.

Results for each step in this process is given in Table V.

Process	Words identified	Valid categories	Valid pron
extr known Afr words	5 925	5 925	5 925
g2p valid Eng	225	189	163
id comps (morfessor)	1 419	1 313	1 265
extract short words	253	196	-
id comps (LSM)	203	179	151
review remaining	1 351	-	-

TABLE V
Per step of the dictionary development process: the number of words correctly identified and the number of valid pronunciations prior to manual correction

C. Dictionary analysis

Using the gold standard dictionary as a reference the phoneme accuracy of the g2p dictionary measured 96.31% with 85.33% of words being identical. This indicates that there is a strong similarity between the two dictionaries. A relative phoneme accuracy of 63.27% was obtained by comparing the grapheme dictionary to the gold standard dictionary. The categorisation of specific differences still requires further investigation. Our findings are presented in Table VI.

Dictionary	Total words	Total phones	Words correct	Phone accuracy
phone	9 374	78 621	-	-
graph	9 374	86 883	6.37%	63.27%
g2p	9 374	78 063	85.33%	96.31%

TABLE VI
Relative phoneme accuracy and percentage of correct words for the g2p dictionary and grapheme dictionary using the gold standard dictionary as reference

D. Effect of dictionary on WER

To evaluate the effect of the dictionaries, we develop three different ASR systems using a relatively standard approach. We use the hidden Markov model toolkit (HTK) [11] and develop context-dependant tied-state acoustic models. Feature extraction on the speech audio data realised 13 Mel Frequency Cepstral Coefficients (MFCCs) with their first and second order derivatives as 39 dimensional feature vectors. MFCC window size was set at 25ms with a frame rate of 10ms. Cepstral mean normalisation was applied at speaker level. With regard to modelling structure, each triphone or trigraph has three emitting states with eight Gaussian mixtures per state and a diagonal covariance matrix. Where parameters are optimised, the development set is used.

Figure 1 shows the effect of different dictionaries on WER at four different training sizes of 5, 10, 20 and 40 hours. At the smallest

data set size (5 hours) the gold standard dictionary outperforms the other approaches, with the g2p-based system also outperforming the grapheme-based system. At the largest data set size (40 hours) the grapheme-based system had a WER of 41.13%, the g2p-based system a WER of 39.82% and the phoneme-based system a WER of 38.03%. As is evident in the convergence of WER between the phoneme-based and grapheme-based ASR systems, the more training data that is available the less the degradation in performance is of the grapheme-based ASR system.

Figure 2 shows the difference in relative percentage of WER between (1) grapheme-based and g2p-based ASR, (2) grapheme-based and phoneme-based ASR and (3) g2p-based and phoneme-based ASR. The highest inter-system difference measured 8.25% between grapheme-based and phoneme-based ASR at 5 hours of training. As then expected, the highest total gain in performance of 5.15% is also measured between grapheme-based and phoneme-based ASR. As training hours increase, g2p-based ASR consistently performs approximately 1.93% worse than phoneme-based ASR. This indicates that even with an increase in training size g2p-based ASR is unlikely to outperform phoneme-based ASR. The lowest inter-system difference measured a very promising 1.31% between g2p-based ASR and grapheme-based ASR.

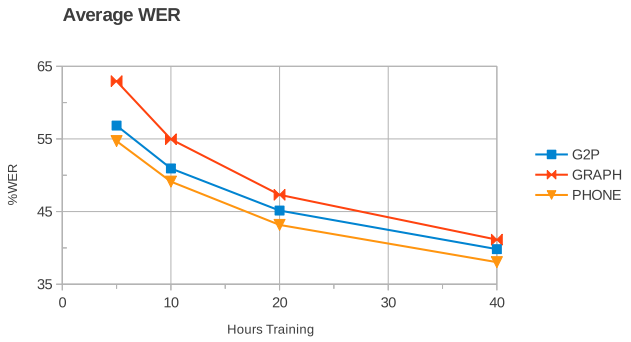


Fig. 1. Average WER of *grapheme-based*, *g2p-based* and *phoneme-based* ASR for training sizes of 5, 10, 20 and 40 hours across 4 folds

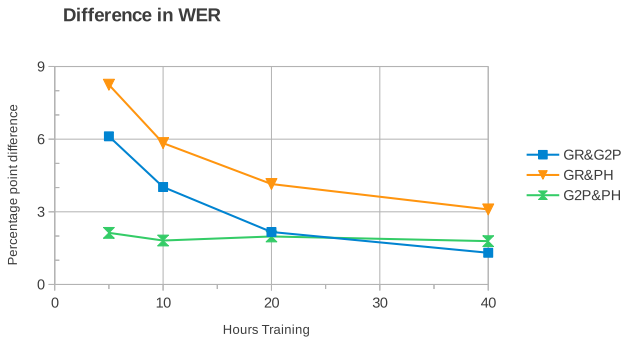


Fig. 2. Average difference in relative percentage of WER between *grapheme-based* and *g2p-based* ASR, *grapheme-based* and *phoneme-based* ASR, and *g2p-based* and *phoneme-based* ASR for training sizes of 5, 10, 20 and 40 hours across 4 folds

E. Error analysis

With the difference in WER being the most pronounced at 5 hours, we analyse the errors made according to word category. As mentioned in section II-C, the abbreviation category contains only one word namely *mej*, and since it doesn't occur in every fold's test set the abbreviation category is ignored during error analysis, leaving a total of 11 categories. Also, it has to be pointed out that words in the spelling error category can only be correctly recognised in their erroneous form. The data set has a fairly low saturation of spelling errors but their effect on recognition accuracy requires further investigation. Ideally (if data containing spelling errors are not to be discarded), spelling errors should either be corrected prior to system development, or the correct and incorrect spellings should be considered the same word during scoring. Both these approaches require that the word actually produced by the speaker should be identified. As this information was not available for the current analysis, spelling errors were handled as if they were standard words.

Table VII gives a detailed view of our findings. Scores are given as a percentage of how many times words from a specific category are miss-recognised as other words out of the total number of words from that category in all 4 test sets. Each cell is coloured green, yellow or red to indicate whether the relevant system performed best, second-best or worst. Not surprisingly grapheme-based ASR performed worse than phoneme-based ASR in 10 of the 11 categories It did however outperform g2p-based ASR in 5 categories namely spelled out words, proper names, spelling errors, partial words and multi-category words. The high WER of spelled out characters can be attributed to the language model used: with a flat language model the insertion penalty (the cost of adding an extra word during decoding) must be very high in order to produce sensible results. This causes short words to be miss-recognised very frequently.

Category	g-based WER	g2p WER	gold-dict WER
Spelled out char	73.73%	68.31%	63.65%
Multi-category	38.53%	40.54%	29.36%
Acronyms	32.03%	28.91%	26.95%
Unknown words	28.65%	25.15%	28.65%
Spelled out word	27.96%	30.53%	15.27%
Foreign	16.04%	14.92%	13.84%
Proper names	10.44%	11.00%	9.48%
Spelling errors	10.40%	11.42%	9.68%
Concatenation	7.48%	5.79%	5.67%
Partial words	6.62%	7.31%	6.13%
Generic Afr words	2.81%	2.49%	2.68%

TABLE VII
Word categories of errors observed at 5 hours of training data

Similarly, with the difference in WER being least at 40 hours, we again split errors based on word categories. Our findings are presented in Table VIII. Comparative to the error analysis of the smallest data set size (5 hours), grapheme-based ASR now outperforms g2p-based ASR in 4 out of the 11 categories, tying for an additional 2 categories. With increased training data, grapheme-based ASR managed to outperform phoneme-based ASR in 5 of the 11 categories. Interestingly, one of the categories includes generic Afrikaans words: the largest category of words in the test set. This might be attributed to noise-free pronunciation models or increased language regularity but this also requires further investigation. The biggest disparity in performance occurs in the spelled out words category between g2p-based and phoneme-based ASR, with g2p-based ASR miss-recognising twice as many words as phoneme-based ASR.

V. Conclusion

In this paper, the recognition accuracy of phoneme-based ASR and grapheme-based ASR was compared, using Afrikaans ASR as a case

Category	g-based WER	g2p WER	gold-dict WER
Spelled out char	62.65%	66.90%	63.89%
Multi-category	37.57%	35.87%	27.52%
Acronyms	31.50%	20.47%	25.98%
Unknown words	25.07%	25.07%	25.66%
Spelled out word	23.24%	28.47%	10.89%
Foreign	13.61%	12.81%	10.00%
Proper names	10.26%	11.83%	9.65%
Spelling errors	10.37%	11.38%	9.22%
Concatenation	5.24%	5.12%	6.33%
Partial words	6.20%	6.20%	8.27%
Generic Afr words	1.85%	1.76%	2.15%

TABLE VIII
Word categories of errors observed at 40 hours of training data

study. It was shown that at a context-level of three (using triphones or trigrams), a minimal effort grapheme-based ASR performs nearly on par with g2p-based ASR and converges quickly to the performance of manually verified phoneme-based ASR as the training set size increases.

Grapheme-based systems do not reach the same level of performance as that of a system developed using a manually verified dictionary, but this degradation in word accuracy is primarily caused by very specific word types, namely: spelled out words, acronyms, proper names and foreign words. All these categories (except for acronyms) tend to have highly irregular relationships between graphemes and phonemes confusing both the g2p-based and grapheme-based systems.

Spelled out words, acronyms and foreign words are typically easy to identify: spelled out words and acronyms tend to be short (and generic short words – which are not acronyms or spelled out words – tend to be known), and foreign words can mostly be identified using known word lists in relevant languages. Proper names tend to be more difficult to identify from text (unless capital letters are accurately retained during pre-processing). Luckily, once identified, these categories tend to be small in comparison with the total number of words to be modelled.

In future work, we will investigate an approach whereby the problematic categories are identified automatically and 'ideal pronunciations' are created for these. We propose that these ideal pronunciations then be converted to grapheme strings (by training phoneme-to-grapheme rules) in order for the pronunciations to be incorporated in a grapheme-based system. Given sufficient data, it may even be possible to train grapheme-to-grapheme rules: transliterating the original orthography of idiosyncratic words to an 'idealised' orthography, more amenable to incorporation in a grapheme-based system. This could possibly combine the best of both worlds: the ability of a dictionary to capture idiosyncratic pronunciations, the minimal effort associated with the development of a grapheme-based system, and the ability of a grapheme-based system to remain 'noise-free', modelling almost all pronunciation variation at the acoustic level. However, in such a process, care should be taken that the additional variability improves the system, and does not introduce the same dictionary inconsistencies found in phoneme-based systems.

REFERENCES

[1] M. Killer, S. Stuker, and T. Schultz, "Grapheme based speech recognition," in *Proc. Eurospeech*, 2003, pp. 3141–3144.

[2] S. Kanthak and H. Ney, "Context-dependent acoustic modeling using graphemes for large vocabulary speech recognition," in *Proc. ICASSP*, 2002, pp. 845–848.

[3] E. G. Schukat-Talamazzini, H. Niemann, W. Eckert, T. Kuhn, and S. Rieck, "Automatic speech recognition without phonemes," in *Proc. Eurospeech*, 1993, pp. 129–132.

[4] J. Dines and M. M. Doss, "A study of phoneme and grapheme based context-dependent asr systems," in *Proceedings of the 4th International Conference on Machine Learning for Multimodal Interaction (MLMI)*, 2008, pp. 215–226.

[5] M. Davel and E. Barnard, "Pronunciation prediction with Default&Refine," *Computer Speech and Language*, vol. 22, pp. 374–393, 2008.

[6] M. Davel and F. de Wet, "Verifying pronunciation dictionaries using conflict analysis," in *Proc. Interspeech*, Tokyo, Japan, 2010, pp. 1898–1901.

[7] N. J. de Vries, J. Badenhorst, M. H. Davel, E. Barnard, and A. de Waal, "Woefzela - an open-source platform for asr data collection in the developing world," in *Proc. Interspeech*, August 2011, pp. 3176–3179.

[8] J. Badenhorst, A. de Waal, and F. de Wet, "Quality measurements for mobile data collection in the developing world," in *Proc. Workshop on Spoken Languages Technologies for Under-Resourced Languages (SLTU)*, Cape Town, South Africa, 2012, pp. 139–145.

[9] M. Creutz and K. Lagus, "Unsupervised morpheme segmentation and morphology induction from text corpora using morfessor," *Publications in Computer and Information Science, Report A*, vol. 81, 2005.

[10] G. B. van Huyssteen and M. M. van Zaanen, "Learning compound boundaries for Afrikaans spelling checking," in *Pre-Proc. Workshop on International Proofing Tools and Language Technologies*, July 2004, pp. 101–108.

[11] S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, G. Moore, J. Odell, D. Ollason, D. Povey, V. Veltchev, and P. Woodland, *The HTK Book*. http://htk.eng.cam.ac.uk/: Cambridge University Engineering Department, 2005.

The Application of Iterated Conditional Modes to Feature Vectors of the Discrete Pulse Transform of Images

Inger Fabris-Rotelli and Jean-Francois Greeff
Department of Statistics, University of Pretoria, 0002, Pretoria, South Africa
E-Mail: inger.fabris-rotelli@up.ac.za

***Abstract*—We present a method for extracting regions of interest from grayscale images by the use of the Iterated Conditional Modes clustering algorithm, in conjunction with the Discrete Pulse Transform of image features. We then illustrate the improvement by comparison; using the luminosity, eccentricity, orientation and convexity as features of the regions of interest.**

I. Introduction

Image segmentation plays an integral part within the field of computer vision and image analysis, since the identification of regions of interest is usually the first step in extracting useful information from an image. To this end, we introduce a new segmentation algorithm in which the Iterated Conditional Modes clustering algorithm is applied to feature vectors calculated from the Discrete Pulse Transform of an image. Possible applications of this and other image segmentation techniques include: medical imaging [1], image compression [2], biometrics, such as facial or retinal recognition [3], [4], scene classification [5], and handwriting recognition [6]. A good overview of possible applications of computer vision (and hence image segmentation) can be found in [6].

The article structure proceeds as follows. Section II provides some initial preliminaries and notation and Section III provides the necessary background theory for the Discrete Pulse Transform. Section IV introduces the iterated conditional modes algorithm and Section V describes the feature measurements used, and lastly Section VI presents some applications.

II. Preliminaries and Notation

We introduce some basics before proceedings with the details. Discussions on the representation of images can be found in many books and articles on image analysis such as [6]. We will represent the luminosity of a pixel in the image matrix as $I(\boldsymbol{x})$ where $\boldsymbol{x} = (i, j)$ with $i = 1, 2, ..., n$; $j = 1, 2, ..., m$. For example, in a grayscale image, $I(\boldsymbol{x})$ will be a single value indicating the luminosity of pixel $\boldsymbol{x}$; while in an RGB image, $I(\boldsymbol{x})$ will be a three-dimensional vector of luminosity values for pixels $\boldsymbol{x}$, in the three different colour bands.

In general, we can represent a collection of different pixel feature measurements as a d-dimensional vector $f(\boldsymbol{x})$. These are the values for a certain pixel across the different layers in a three dimensional matrix image, where each entry in the vector corresponds to a feature value of the pixel, which could simply be it's luminosity $I(\boldsymbol{x})$.

A *neighbourhood* of a given pixel is a subset of the image matrix which surrounds the pixel, which may or may not contain the pixel itself. Such a neighbourhood for a pixel $\boldsymbol{x}$ can be described as

$$\begin{aligned} N(\boldsymbol{x}) = \quad & \{\boldsymbol{x} = (s,t) \,:\, (i,j)\,,(i \pm 1, j)\,,(i, j \pm 1)\,, \\ & (i \pm 1, j \pm 1)\,,(i \pm 1, j \mp 1)\,; \\ & s \in [0,n]\,;\, t \in [0,m]\}. \end{aligned}$$

This neighbourhood defines a *8-connectivity* of $\boldsymbol{x}$ since it contains the 8 pixels which neighbour $\boldsymbol{x}$. We may also define a *4-connectivity* by excluding the 4 pixels on the diagonals.

Image segmentation generally focuses on the method of thresholding, where a threshold value T is selected such that all luminosities $I(\boldsymbol{x})$ above (below) this value is classified as foreground (background) and represented by a black (white) pixel with luminosity 0 (1). This can be achieved through iterative selection where T is selected iteratively as the average of the average luminosities in the foreground and background; Otsu's method which minimises the within-segment variation; or the balanced histogram method which uses a weighted mid-point of the pixel luminosities as the threshold. More sophisticated methods, such as the fitting of Gaussian Mixtures to the luminosities and adaptive thresholding aim to improve on these segmentations. An overview of these, and other, methods can be found in [7].

III. Discrete Pulse Transform

The discrete pulse transform (DPT) is based on the framework of LULU operators. An important overview is presented in [8], whereas [9] extend the results to multidimensional arrays, with applications in image analysis. We present a summary of the definitions and results presented in the latter, as this work has been extensively published in detail in the references mentioned. First we define a connection, needed as a preliminary for the definition of the LULU operators in Definition 2.

Definition 1. If B is any non-empty set, then a family $\mathscr{C}$ of subsets of B is called a *connection* on B if

1) $\emptyset \in \mathscr{C}$,

Thanks to the Department of Statistics and Statomet at the University of Pretoria for support and funding.

2) $\{\boldsymbol{x}\} \in \mathscr{C}\ \forall\ \boldsymbol{x} \in B$, and
3) $\{C_i : i \in I\} \subseteq \mathscr{C}, \bigcap_{i\in I} C_i \neq \emptyset \implies \bigcup_{i\in I} C_i \in \mathscr{C}$.

A set C is called *connected* if it belongs to some connection $\mathscr{C}$ of B.

Definition 2. For I a function defined on a vector lattice $\mathcal{A}(\mathbb{Z}^d)$ of real functions defined on $\mathbb{Z}^d$, and $n \in \mathbb{N}$; then for all $\boldsymbol{x} \in \mathbb{Z}^d$, we define the *LULU operators* as

$$L_n(I)(\boldsymbol{x}) = \max_{V\in\mathcal{N}_n(\boldsymbol{x})} \min_{\boldsymbol{y}\in V} I(\boldsymbol{y})$$

and

$$U_n(I)(\boldsymbol{x}) = \min_{V\in\mathcal{N}_n(\boldsymbol{x})} \max_{\boldsymbol{y}\in V} I(\boldsymbol{y}),$$

where $\mathcal{N}_n(\boldsymbol{x}) = \{V \in \mathscr{C} : \boldsymbol{x} \in V, \text{card}(V) = n+1\}$.

The recursive application of the operators L_n and U_n for $n = 1, 2, .., N$, where N is the total number of pixels in the image, results in the Discrete Pulse Transform (DPT) of an image f.

Definition 3. The *DPT* of a function $I \in \mathcal{A}(\mathbb{Z}^d)$, with $N = \text{card}(\text{supp}(I)) = \text{card}\left(\{\boldsymbol{x} \in \mathbb{Z}^d : I(\boldsymbol{x}) \neq 0\}\right)$, is given by

$$DPT(I) = (D_1(I), D_2(I), ..., D_N(I))$$

where

$$D_1(I) = (\mathcal{I} - P_1)(I)$$

and

$$D_n(I) = (\mathcal{I} - P_n) \circ Q_{n-1}(I),$$

with $P_n = L_n \circ U_n$ or $U_n \circ L_n$ and $Q_n = P_n \circ ... \circ P_1$ for $n = 1, 2, ..., N$. The operator $\mathcal{I}$ is the identity operator in $\mathcal{A}(\mathbb{Z}^d)$.

The application of the DPT to an image I provides a multiscale decomposition into pulses (Definition 4) given in Theorem 5 below [9].

Definition 4. A function $\phi \in \mathcal{A}(\mathbb{Z}^d)$ is called a *pulse,* if for some connected set V and nonzero real number α, we have that $\phi(\boldsymbol{x}) = \alpha$ whenever $\boldsymbol{x} \in V$, and zero otherwise.

Theorem 5. *Let $I \in \mathcal{A}(\mathbb{Z}^d)$, then*

$$I = \sum_{n=1}^{N} D_n(I) \tag{1}$$

Also, for every $n \in \mathbb{N}$, the function $D_n(I)$ is a sum of discrete pulses with pairwise disjoint support. In other words, there exists some number $\gamma(n) \in \mathbb{N}$ and discrete pulses ϕ_{ns}, $s = 1, ..., \gamma(n)$ such that

$$D_n(I) = \sum_{s=1}^{\gamma(n)} \phi_{ns}$$

and

$$supp(\phi_{ns_1}) \cap supp(\phi_{ns_2}) = \emptyset\ \forall\ s_1 \neq s_2 \tag{2}$$

where ϕ_{ns} is the s^{th} discrete pulse on some connected set V with card(supp(ϕ_{ns})) $= n$. Further, for all $n_1 < n_2 \in \mathbb{N}$, $1 \leq s_1 < \gamma(n_1)$ and $1 \leq s_2 < \gamma(n_2)$ we have that

$$\begin{aligned} &supp(\phi_{n_1 s_1}) \cap supp(\phi_{n_2 s_2}) \neq \emptyset \implies \\ &supp(\phi_{n_1 s_1}) \subset supp(\phi_{n_2 s_2}). \end{aligned} \tag{3}$$

Hence, together with equation (1) we can write

$$I = \sum_{n=1}^{N} \sum_{s=1}^{\gamma(n)} \phi_{ns}$$

where properties (2) and (3) hold.

This decomposition of the function I thus extracts connected regions in the image matrix. It allows us to decompose the image into pulses whose support represent the disjoint connected regions of different sizes. For examples, supp(ϕ_{ns}) $\forall\ s = 1, 2, ..., \gamma(n)$ represents the pixel locations of the $\gamma(n)$ pulses with support of cardinality n. On these sets we are able to calculate different features of the connected regions within an image.

IV. Iterated Conditional Modes

Iterated Conditional Modes (ICM) is an algorithm first introduced by [10] to reduce the noise in dirty pictures. It takes into account both features of each pixel and spatial information based on a Markov Random Field of each pixel to be clustered [5]. For a further explanation of Markov Random Fields in relation to images, see [11].

The ICM algorithm (within the context of noise removal) is based on the assumption that neighbouring pixels tend to have similar luminosities, or other features; and that each pixel is corrupted independently with a given probability. Within the general context of image analysis we apply the method to the general feature vectors $f(\boldsymbol{x})$ of each pixel $\boldsymbol{x}$ and present the method as such.

Consider an image I with n pixel rows and m pixel columns in which there are K clusters of pixels which we would like to detect and extract. Then, for each iteration of the algorithm, indexed by α, we define

- $\omega_{ij}^{(\alpha)}$ as the class of pixel $\boldsymbol{x} = (i, j)$;
- $C_k^{(\alpha)} = \left\{\boldsymbol{x} : \omega_{ij}^{(\alpha)} = k\right\}$ as the set of pixels belonging to cluster $k = 1, 2, ..., K$;
- $N_k^{(\alpha)} = \text{card}\left(C_k^{(\alpha)}\right)$ as the number of pixels in cluster $k = 1, 2, ..., K$;
- $N_{ij}^{(\alpha)}(k) = \text{card}\left(C_k^{(\alpha)} \cap N(\boldsymbol{x})\right)$ as the number of pixels in the neighbourhood of $\boldsymbol{x}$ belonging to cluster $k = 1, 2, ..., K$;
- $\boldsymbol{\mu}_k^{(\alpha)} = \frac{1}{N_k^{(\alpha)}} \sum_{\boldsymbol{x}\in C_k^{(\alpha)}} f(\boldsymbol{x})$ as the d-dimensional mean vector of cluster $k = 1, 2, ..., K$;
- $\nu^{(\alpha)} = \frac{1}{nm} \sum_{k=1}^{K} \left[\sum_{\boldsymbol{x}\in C_k^{(\alpha)}} \left(f(\boldsymbol{x}) - \boldsymbol{\mu}_k^{(\alpha)}\right)' \left(f(\boldsymbol{x}) - \boldsymbol{\mu}_k^{(\alpha)}\right) \right]$ as the total within-cluster variance.

The aim of ICM clustering is then to minimise the total within-cluster variance, by assigning and reassigning each pixel in the image to a class, while taking spatial information into account. To this end we proceed as follows:

1) Initialise the the parameters using only the feature information for each pixel. We suggest, and have used, a multivariate K-means clustering procedure. A good overview of the method and it's implementation can be found in [12].

2) Calculate $C_k^{(\alpha)}$, $N_k^{(\alpha)}$, $\boldsymbol{\mu}_k^{(\alpha)}$ and $\nu^{(\alpha)}$ for each $k = 1, 2, ..., = K$.
3) For each pixel $\boldsymbol{x}$ and cluster k calculate

$$\Lambda(\boldsymbol{x}, k) = \left(f(\boldsymbol{x}) - \boldsymbol{\mu}_k^{(\alpha)}\right)^T \left(f(\boldsymbol{x}) - \boldsymbol{\mu}_k^{(\alpha)}\right) - \beta\nu^{(\alpha)} N_{ij}(k), \quad (4)$$

and find $k^* = \arg\min_k \{\Lambda(\boldsymbol{x}, k)\}$.
4) Set $\omega_{ij}^{(\alpha+1)} = k^*$ for each pixel in the image. In other words, reclassify the pixel as belonging to cluster k^*.
5) Repeat steps 2 through 4 until $C_k^{(\alpha+1)} = C_k^{(\alpha)}$ for all $k = 1, 2, ..., K$ or a predetermined number of iterations have passed.

If the algorithm converges we end up with a collection of sets (the $C_k's$) containing the clusters of pixels, grouped according to their likely cluster membership, taking into account the spatial location of the pixel.

Equation (4) is similar to the function which must be minimised within the K-means framework. The only difference is the second term, which is called the *spatial penalisation* term [5]. This term allow for the inclusion of spatial information when clustering the pixels. In effect, the within-cluster sum of square deviations is reduced by a multiple of the number of pixels in the neighbourhood which are in the class k, where the specific constant of multiplication is $\beta\nu^{(\alpha)}$. All else being constant, if a pixel is surrounded by many pixels which belong to class k^*, it is more likely that that pixel also belongs to class k^*, due to our first assumption. Hence the likelihood that k^* will minimise (4) is increased, by reducing the size of $\Lambda(\boldsymbol{x}, k^*)$ by a constant related to the spatial information.

Debba et al [5] suggests that a good choice for the parameter β is 1.5. A larger value of β will lead to a smoother image based more heavily on spatial data, while a lower value for β will lead to a clustering similar to the K-means algorithm. Figure 1 illustrates the effect of the β parameter. From Figure 1b, it can be seen that a smoother segmented image is obtained when ICM is applied directly to the pixel luminosities with a higher value for β, while, as seen in Figure 1c, the effect is minimal when applied to the feature vectors of the DPT. This can be explained by the fact that the spatial information is already largely contained within the pulse supports. Hence, the application of ICM to DPT is more robust to the choice of β, which constitutes a major advantage in application. A value of 2.5 was used for β throughout.

V. Region Features

We will now present a number of measurements of properties of regions within an image, with the aim of obtaining improved segmentation of an image.

Eccentricity is a feature which can be extracted from a set of connected pixels indicating the roundness of the region. It is defined as

$$\epsilon = \sqrt{1 - \frac{b^2}{a^2}}$$

where a and b are the lengths of the semi-major and semi-minor axes, respectively, of the ellipse with the same second moment as the region of pixels. For all $p, q \in \mathbb{N}_0$ the moments of order $p + q$ for an ellipse $f(x, y)$ are $\mu_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^p y^q f(x, y)\, dxdy$, with the corresponding moments, $m_{pq} = \sum_{\forall(x,y)\in V} x^p y^q I(x, y)$, for the connected image region V [13]. Now, since $a \geq b$, the eccentricity of a region is bounded by 0 and 1. When $\epsilon = 0$, the region is a perfect circle, whereas if $\epsilon = 1$, the region is simply a line segment. This can, also, then be interpreted as a measure of the shape and compactness of the region. Figure 2a illustrates the eccentricity of a region of 5 connected pixels.

We can inspect the direction in which a region of pixels lie. In order to do so, we calculate the region's *orientation* as the gradient of the major axis of the ellipse, with the same second moments as the region, in degrees from the horizontal axis. Figure 2b illustrates this measure for a region of 5 pixels. The gradient of the regression line (major axis) through the connected image region, V, is given by

$$\hat{m} = \frac{\sum_{\forall(x,y)\in V} xy - n\bar{x}\bar{y}}{\sum_{\forall(x,y)\in V} x^2 - n\bar{x}^2}$$

where $\bar{x} = \frac{1}{n}\sum_{\forall(x,y)\in V} x$, $\bar{y} = \frac{1}{n}\sum_{\forall(x,y)\in V} y$ and $n = \text{card}(V)$. It follows that the orientation of V is $\theta = \arctan(\hat{m})$ [14]. From this definition, it is clear that the orientation will be bounded by $-90°$ and $90°$. An example of the use of orientation in texture analysis can be found in [15].

The *convex hull* of an image region is the smallest convex set of points, which contains the entire image region [16] as can be seen in Figure 2c. Methods for calculating the convex hull can be found in [17] or [18].

Another measure of compactness, based on the convex hull, is the *convexity* of a connected region. Iivarinen and Visa [19] define the convexity of a connected region V as

$$c = \frac{\text{card}(\text{adj}(\text{CH}(V)))}{\text{card}(\text{adj}(V))}$$

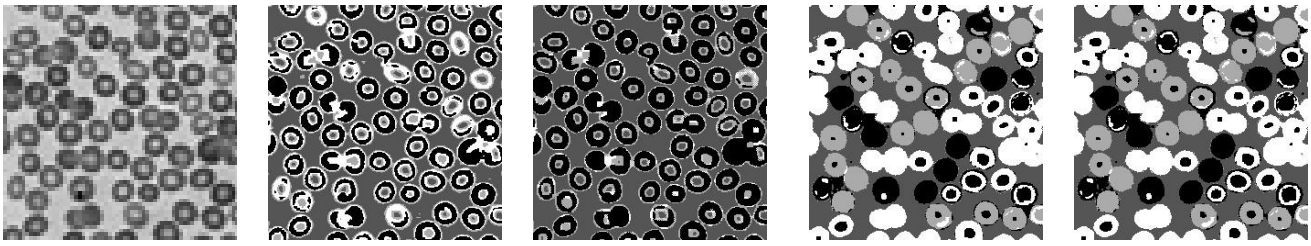
where $\text{CH}(V)$ is the convex hull of V and $\text{adj}(V) = \{\boldsymbol{x} \in \mathbb{Z}^d : \boldsymbol{x} \notin V, V \cup \{\boldsymbol{x}\} \in \mathscr{C}\}$. Stated differently, the convexity of a region is the ratio of the length of the perimeter of the convex hull of V, to the length of the perimeter of V. A convexity of 1, indicates that the region is perfectly convex, since the convex hull would coincide with the region; whereas a value significantly different from 1 would indicate a non-convex or concave region.

Other measurements of region features which could be made on an image include (but are not limited to) van der Walt's sharpness measure [20], entropy [21], Euler number [22], and shape number [23].

VI. The Application of ICM to DPT Decompositions of Image Features

We define the notation in Table I in order to apply ICM to the feature vectors of the Discrete Pulse Transform supports.

We use the standardised feature measures to remove the effect of the measurement scales, which allows for an even weighting between the feature measurements during the clustering process.

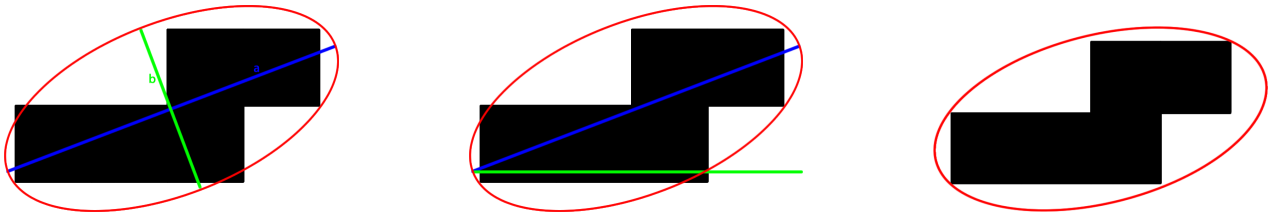


(a) Original image.

(b) ICM applied directly to image luminosities with $\beta = 1.5$ (left) and $\beta = 2.5$ (right).

(c) ICM applied to discrete pulses with $\beta = 1.5$ (left) and $\beta = 2.5$ (right).

Figure 1: Effect of β-values for ICM applied to the image luminosities, and to the feature vectors calculated on the discrete pulses.



(a) A superimposed ellipse with the same second moments as the region. The ratio of the semi-major and semi-minor axes are used to calculate the region's eccentricity.

(b) A superimposed ellipse with the same second moments as the region. The angle between the horizontal axis and major axis is a measure of the region's orientation.

(c) The convex hull of the region superimposed onto the region. The ratio of the perimeter lengths is a measure of the region's convexity.

Figure 2: Feature measures of a connected region of 5 pixels.

ϵ_{ns}, θ_{ns} and c_{ns}	Eccentricity, orientation and convexity of supp(ϕ_{ns}).
$\Omega(\boldsymbol{x}) = \{\phi_{ns} : \boldsymbol{x} \in \text{supp}(\phi_{ns})\}$	Set of all pulses to which pixel $\boldsymbol{x}$ belongs, with $\omega_{ij} = \text{card}(\Omega(\boldsymbol{x}))$.
$\bar{\epsilon}_{ij} = \sum_{\forall \boldsymbol{x} \in \Omega(\boldsymbol{x})} \frac{\epsilon_{ns}}{\omega_{ij}}$	Average eccentricity of pixel $\boldsymbol{x}$ over all pulses towhich it belongs, with standard value $\bar{\epsilon}^s_{ij} = \bar{\epsilon}_{ij}$.
$\bar{\theta}_{ij} = \sum_{\forall \boldsymbol{x} \in \Omega(\boldsymbol{x})} \frac{\theta_{ns}}{\omega_{ij}}$	Average orientation of pixel $\boldsymbol{x}$ over all pulses to which it belongs, with standard value $\bar{\theta}^s_{ij} = \frac{\bar{\theta}_{ij}+90}{180}$.
$\bar{c}_{ij} = \sum_{\forall \boldsymbol{x} \in \Omega(\boldsymbol{x})} \frac{c_{ns}}{\omega_{ij}}$	Average convexity of pixel $\boldsymbol{x}$ over all pulses to which it belongs, with standard value $\bar{c}^s_{ij} = \bar{c}_{ij}$.
$l^s_{ij} = \frac{I(\boldsymbol{x}) - I_{min}}{I_{max} - I_{min}}$	Standardised luminosity value of pixel $\boldsymbol{x}$, with $I_{min} = \min I(i,j)$ and $I_{max} = \max I(i,j)$.
$f^s(\boldsymbol{x}) = \left(l^s_{ij}, \bar{\epsilon}^s_{ij}, \bar{\theta}^s_{ij}, \bar{c}^s_{ij}\right)'$	Standardised feature vector of pixel $\boldsymbol{x}$ using the average discrete pulse support feature measurements of the pulses to which $\boldsymbol{x}$ belongs.

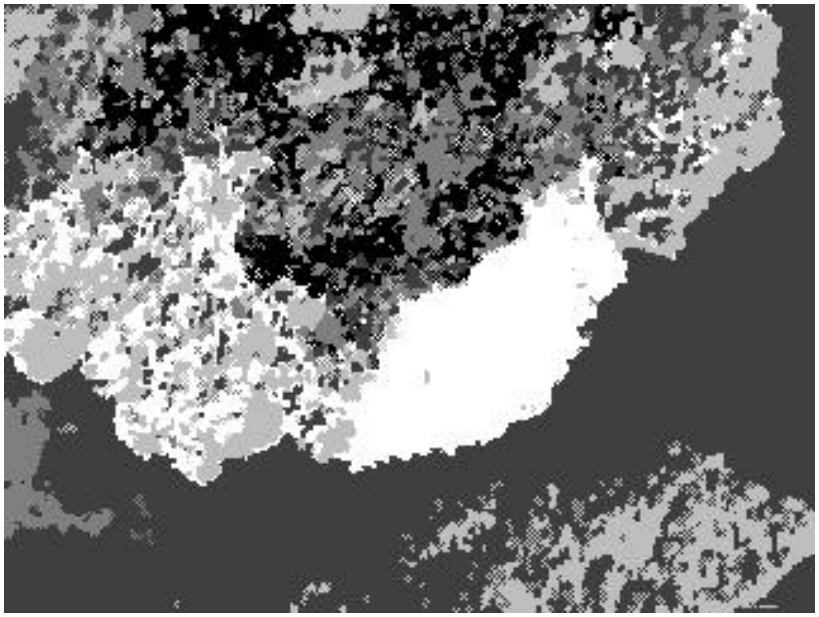
Table I: Summarised notation for the feature vectors

After calculating the standardised feature vectors for each pixel in the image, we apply ICM to obtain a segmented image based on the pixel luminosity; region eccentricity, orientation and convexity; and the spacial features of the DPT and ICM. Since we are including more information about the structure of the image, we expect to obtain an improved segmentation of the image when compared to ICM applied to the pixel luminosities, while extracting regions within the image with similar features.

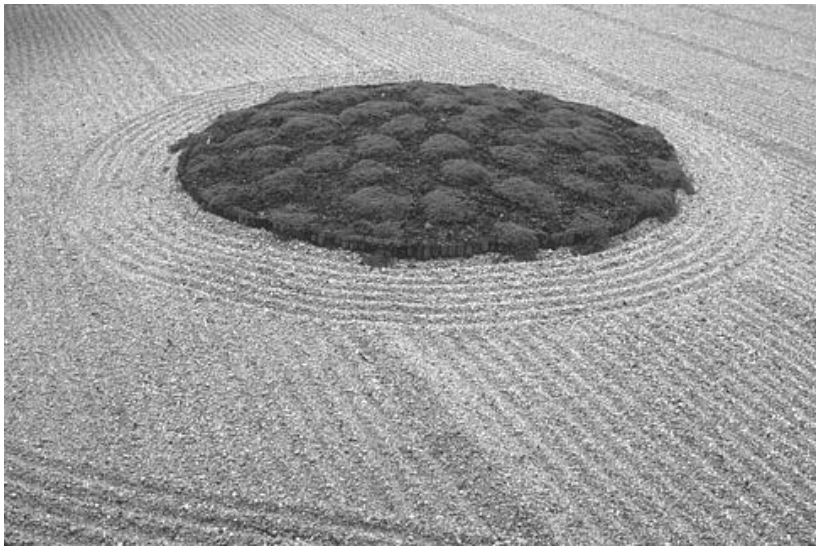
Figure 3b, presents the results of this algorithm when applied to the image provided in Figure 3a. It is clear that we are able to successfully distinguish between the body of water, trees and dirt within the image, while the apparent misclassification within the trees is due to the different types of vegetation and textures evident within those regions. This technique could, hence, be an effective method for the classification of different regions in aerial photographs or extended for use in target detection. For instance, in this example, successive photographs of the same area could be used to measure the volume of water in the lake, in order to monitor the effect of rainfall in the area; or measure the change in vegetation density for the area.

Figure 3d, presents the results of the algorithm applied to the image in Figure 3c. This constitutes an example of an application in texture analysis. From the original image it clear that there are two distinct sets of textures (the raked lines and the grass mound), with a secondary texture within each (the circular lines and straight lines; and the smaller mounds on top of the larger mound). These are visible to the human eye within the image. We see here that, allowing for a certain level of noise due to the complex textures and shadows, that this algorithm is able to separate the two major textures effectively, as well as the smaller mounds from the larger mound; but it fails to distinguish between the two different

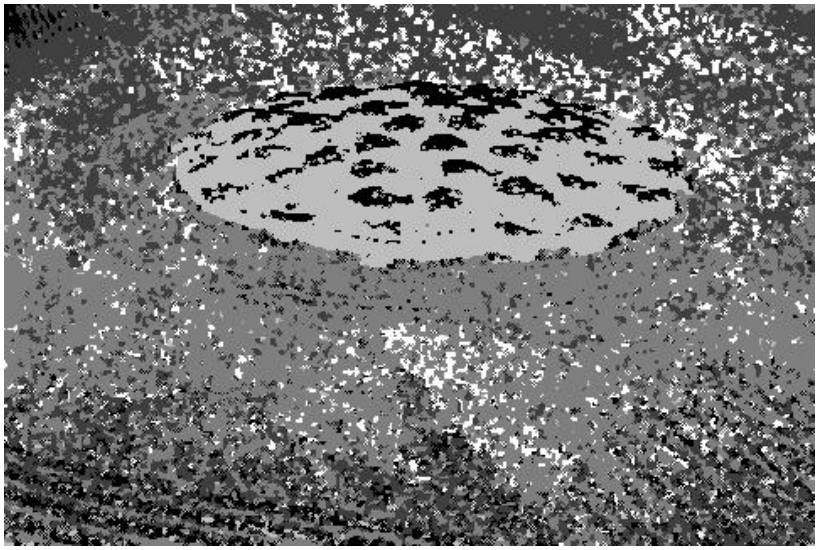
(a) Original image.



(b) $K = 5$ and $\beta = 2.5$.

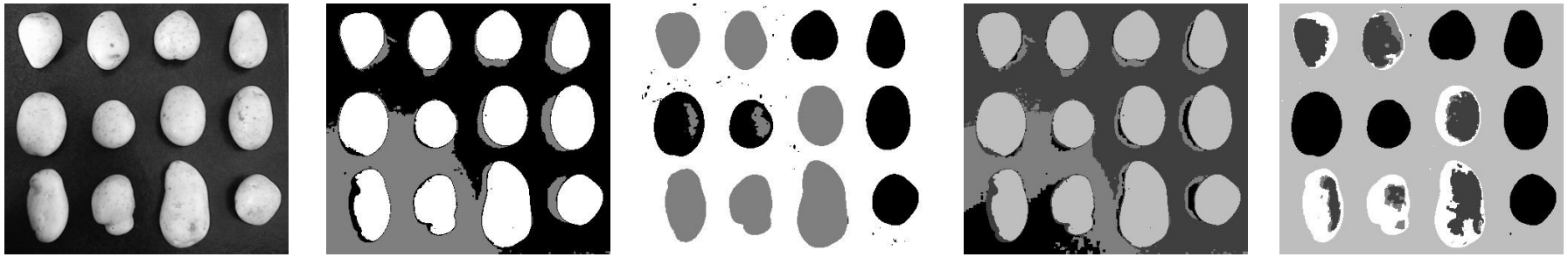


(c) Original image.

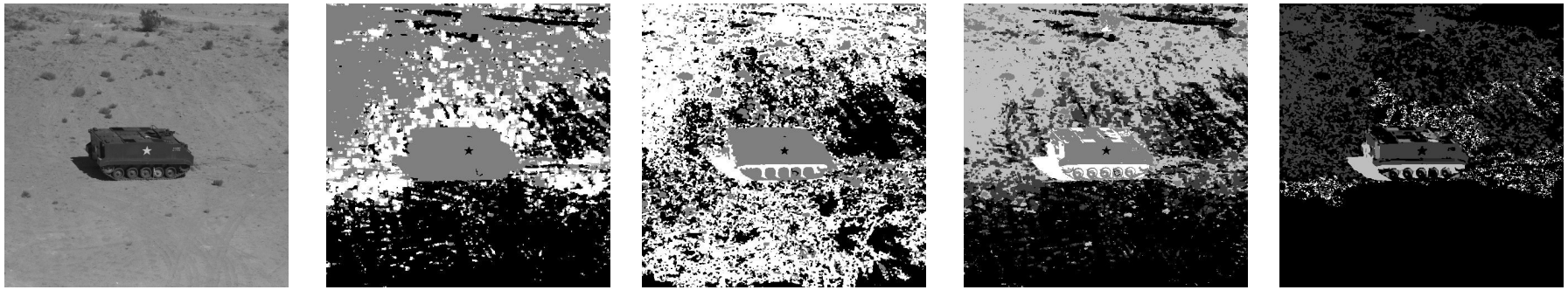


(d) $K = 5$ and $\beta = 2.5$.

Figure 3: ICM applied to feature vectors of discrete pulses.



(a) Original image. (b) $K = 3\,;\ \beta = 2.5$. (c) $K = 5\,;\ \beta = 2.5$.



(d) Original image. (e) $K = 3\,;\ \beta = 2.5$. (f) $K = 5\,;\ \beta = 2.5$

Figure 4: Comparison of ICM applied directly to the pixel luminosities (left), against ICM applied to the feature vectors of discrete pulses (right).

sets of lines. This can be explained by the use of shape and compactness as feature measures. At a pulse level, we expect the lines to exhibit similar elongated shapes, while the mounds would exhibit a more compact and round shape. Further, we see that the smaller mounds appear to present a consistent crescent shape while the large mound presents a more irregular shape. The orientation of the lines have aided in distinguishing between parts of the texture for the raked lines, as can be seen in the lower parts of the image. However, we could improve the effectiveness of the segmentation of the lines (and other textures) using a Hough Transform, as described in [24], or other feature measurements.

We compare the result of ICM applied directly to the pixel luminosities, against ICM applied to the feature vectors of the DPT in Figure 4. Figures 4a and 4d present the original images; while Figures 4b, 4c, 4e and 4f present the results of ICM applied directly to the luminosities (left), and the feature vectors of the discrete pulses (right), for various values of K.

From Figure 4b and 4c we see that ICM applied to the DPT is less affected by ambient light, present in the lower-left of the image. Also, this algorithm is able to distinguish between potatoes which have different shapes, whereas the original ICM algorithm only segments the potatoes from the background. The potatoes coloured black are observed to have a rounder, more compact shape compared to the other potatoes which exhibit a more elongated shape.

Figure 4e shows the improvement of ICM applied to the DPT in target detection, or the segmentation of man-made objects from natural backgrounds. In this case, the original ICM algorithm does not successfully segment the truck from the background, while when applied to the DPT, we can clearly distinguish between the truck and the background.

At first glance, Figure 4f appears to exhibit outperformance of the original ICM algorithm over ICM applied to the DPT. However, after further inspection, it is clear that ICM applied to the DPT is able to segment particular details, on top of the truck, which the original ICM fails to achieve. For use in image compression this constitutes a big advantage, since the loss of information is minimised.

One benefit of ICM applied to the DPT compared to the direct application to pixel luminosities, which is not directly apparent from the segmented images, but which is clear from the algorithm and its results, is the inclusion of the various region features of the pixels. Most of the improvements presented here are a direct result of the inclusion of these feature measurements. Although the ICM algorithm includes spatial features of the image, we are not able to include the measured features in the original ICM algorithm, since the features cannot be calculated on a single pixel. Overall, the results show that the ICM algorithm applied to the feature vectors, calculated on the DPT of an image, provides better segmentation results, through the incorporation of measured spatial features of image regions.

VII. Conclusion

The application of ICM to the feature vectors calculated on the DPT of an image yields an effective algorithm for image segmentation, which is robust to the choice of the coefficient β of the spatial penalisation factor. This leads to improved results in the application of the algorithm in image segmentation, which is easier to apply where automatic segmentation is required. Due to the complex nature of image analysis, a perfect segmentation would not be possible; and within each unique application, different sets of region features would be needed to obtain efficient results. We have used eccentricity, orientation and convexity of image regions as our feature measurements, but generally the choice is at discretion of the implementer in any given situation.

We have discussed the possible advantages of this algorithm in the context of scene classification, image compression and texture analysis, but the algorithm could be applied in any other segmentation task. Enhancements, including different feature measurements, have also been suggested.

When compared to ICM applied directly to the pixel luminosities, we observe a marked improvement in the segmentation results. We are able to extract smaller details within the image, which could lead to better image compression, and fewer distortions due to ambient light. These improvements can be attributed to the inclusion of calculated region features, not present in ICM, as well as an increased emphasis on spatial features. Further research will involve comparisons with other segmentation techniques making use of features such as those used herein. This work simply provides an indication of the ability of the DPT in image analysis. Performance analysis such as class separability indices will also be investigated in future [25], [26].

References

[1] D. Pham, C. Xu, and J. Prince, "Current Methods in Medical Image Segmentation," *Annu Rev Biomed Eng*, vol. 2, pp. 315–337, 2000.

[2] J. Vaisey and A. Gersho, "Image compression with variable block size segmentation," *IEEE T Signal Proces*, vol. 40, no. 8, pp. 2040–2060, 1992.

[3] A. Jain and U. Park, "Facial marks: Soft biometric for face recognition," in *Proceedings of Sixteenth IEEE International Conference on Image Processing*, 2009, pp. 37–40.

[4] J. Matey, R. Broussard, and L. Kennell, "Iris image segmentation and sub-optimal images," *Image and Vision Computing*, vol. 28, no. 2, pp. 215–222, 2010.

[5] P. Debba, A. Stein, F. van der Meer, E. Carranza, and A. Lucieer, "Field Sampling from a Segmented Image," in *Computational Science and Its Applications – ICCSA 2008*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2008, vol. 5072, pp. 756–768.

[6] L. Shapiro and G. Stockman, *Computer Vision*. Upper Saddle River, NJ: Prentice Hall, 2002.

[7] I. Fabris-Rotelli and J. Greeff, "An overview of image segmentation techniques," in *Proceedings of the Fifty-fourth Annual Conference of the South African Statistical Association*, 2012, pp. 34–41.

[8] C. Rohwer, *Nonlinear Smoothers and Multiresolution Analysis*. Cambridge, MA: Birkhäuser, 2005.

[9] R. Anguelov and I. Fabris-Rotelli, "LULU Operators and Discrete Pulse Transform for Multidimensional Arrays," *IEEE T Image Process*, vol. 19, no. 11, pp. 3012–3023, November 2010.

[10] J. Besag, "On the statistical analysis of dirty pictures," *Journal of the Royal Statitical Society*, vol. 48, no. 3, pp. 259–302, 1986.

[11] P. Perez, "Markov random fields and images," *CWI Quarterly*, vol. 11, no. 4, pp. 413–437, 1998.

[12] J. Hartigan and M. Wong, "A K-means clustering algorithm," *Applied Statistics*, vol. 28, pp. 100–108, 1979.

[13] J. Flusser and T. Suk, "Rotation Moment Invariants for Recognition of Symmetric Objects," *IEEE T Image Process*, vol. 15, no. 12, pp. 3784–3790, December 2006.

[14] A. Steyn, C. Smit, S. du Toit, and C. Strasheim, *Modern Statistics in Practice*. Pretoria: JL van Schaik Publishers, 1994.
[15] D. Chetverikov and A. Hanbury, "Finding Defects in Texture using Regularity and Local Orientation," *Pattern Recogn*, vol. 35, no. 10, pp. 2165–2180, 2002.
[16] E. Kreyszig, *Introductory Functional Analysis with Applications*. John Wiley and Sons, 1978.
[17] C. Ronse, "A Bibliography on Digital and Computational Convexity (1961-1988)," *IEEE T Pattern Anal*, vol. 11, no. 2, pp. 181–190, February 1989.
[18] P. Soille, "From Binary to Grey Scale Convex Hulls," *Fund Inform*, vol. 41, pp. 131–146, 2000.
[19] J. Iivarinen and A. Visa, "An Adaptive Texture and Shape Based Defect Classification," in *Proceedings of Fourteenth International Conference on Pattern Recognition, 1998*, vol. 1, August 1998, pp. 117–122.
[20] S. van der Walt, "Super-Resolution Imaging," Ph.D. dissertation, University of Stellenbosh, December 2010.
[21] A. Kadir and M. Brady, "Saliency, scale and image description," *Int J Comput Vision*, vol. 45, no. 2, pp. 83–105, 2001.
[22] C. Dyer, "Computing the Euler Number of an Image from it's Quadtree," *Comput Vision Graph*, vol. 13, no. 3, pp. 270–276, 1980.
[23] P. Danielson, "A New Shape Factor," *Comput Vision Graph*, vol. 7, no. 2, pp. 292–299, 1978.
[24] R. Duda and P. Hart, "Use of the Hough transformation to detect lines and curves in pictures," *Commun. ACM*, vol. 15, no. 1, pp. 11–15, January 1972.
[25] D. Aha and R. Bankert, "Feature selection for case-based classification of cloud types: an empirical comparison," in *Proceedings of the AAAI-94 Workshop on Case-Based Reasoning*, 1994.
[26] D. Zighed, S. Lallich, and F. Muhlenbach, "Separability index in supervised learning," *Principles of data mining and knowledge discovery, Lecture Notes in Computer Science*, vol. 2431/2002, pp. 241–267, 2002.

Improved transition models for cepstral trajectories

Jaco Badenhorst
Multilingual Speech Technologies
North-West University,
Vanderbijlpark 1900, South Africa
[2]Human Language Technology
Competency Area,
CSIR Meraka Institute
Email: jbadenhorst@csir.co.za

Marelie H. Davel
Multilingual Speech Technologies
North-West University,
Vanderbijlpark 1900, South Africa
Email: marelie.davel@gmail.com

Etienne Barnard
Multilingual Speech Technologies
North-West University,
Vanderbijlpark 1900, South Africa
Email: etienne.barnard@gmail.com

***Abstract*—We improve on a piece-wise linear model of the trajectories of Mel Frequency Cepstral Coefficients, which are commonly used as features in Automatic Speech Recognition. For this purpose, we have created a very clean single-speaker corpus, which is ideal for the investigation of contextual effects on cepstral trajectories. We show that modelling improvements, such as continuity constraints on parameter values and more flexible transition models, systematically improve the robustness of our trajectory models. However, the parameter estimates remain unexpectedly variable within triphone contexts, suggesting interesting challenges for further exploration.**

I. Introduction

Current approaches to automatic speech recognition (ASR) require large amounts of speech data to achieve high accuracies, since context-dependent modelling of phones is an important feature of these approaches. The requirement for context-dependent modelling results from the physical constraints of the human vocal tract, which results in co-articulation effects during the transition from one phone to the next. Since state-of-the-art ASR systems model speech with piecewise-constant statistical models, observations of the influences of various phonetic contexts on each phone are required to create adequate statistical models of the effects of co-articulation. Hence, sufficient examples are required for each representative context. Unfortunately, this leads to substantial data requirements.

Trajectory modelling approaches [1], [2] have attempted to model temporal information in a more explicit fashion in order to reduce these data requirements. It is clear that the effects of co-articulation are not constrained to the frame level – appropriate models need to operate at the segmental level, and even longer-term effects must be considered. Describing the observed variability on all these levels is a challenging problem. We are specifically interested to know whether systematic phone transition effects may be described more accurately. It is our belief that finding appropriate representations is important to enable more effective parameter sharing, and thus more data-efficient ASR.

With this work we improve on a model that can be used to isolate the key elements that occur in acoustic features during phone-to-phone transitions. We first show that trajectory tracking may be accomplished for a basic model and the ability of the model to predict trajectory behaviour at different context sizes is further evaluated. To better account for additional trajectory behaviour, a more complex model description is developed to characterise the observed variability.

This paper is structured as follows: Related research is discussed in Section II. Specific techniques used to model phone transitions and the measurement strategies thereof are presented in Section III. We then describe our experimental setup in Section IV and details regarding our experiments and results are given in Section V. Our concluding statements are made in Section VI.

II. Background

Accurate modelling of co-articulation effects in speech data has been the main driving force behind the development of large speech recognition corpora [3]. In fact, if unlimited training data were available, it would be more beneficial to model co-articulatory effects using whole word (or even phrasal) units instead of phones as the basic modelling unit, since co-articulation effects are increasingly well modelled by larger contexts. Limited training data, however, forces the use of smaller units. Context-dependent phones are currently widely used to approximate the co-articulation effects for accurate speech recognition [3]. Finding the correct segment size to model the diversity of all co-articulation effects can, however, prove difficult. A key motivating factor for the development of segmental models is the fact that it is possible to exploit acoustic features that are apparent at the segmental and not at the frame level [1], [2].

The hope is that more data-efficient models of co-articulation can be developed in this way, but this is not a straightforward goal to achieve. One problem is that any segmental approach needs to model extra-segmental variability (between different examples of speech segments) as well as intra-segmental variability (within a single example) accurately. The observed variability for segments of variable length may have multiple sources, and their interaction is currently not well understood. Possible origins for these sources include factors such as recording conditions, different speaking styles, phonetic reduction and finally co-articulation.

A wide variety of approaches to the development of segmental models have been proposed, based on several fundamental observations. For example, in [2] the fact that time-normalised phones tend to behave predictably in various phonetic contexts was used to develop a probabilistic trajectory model. Also, the speech production process suggests the influence of underlying articulatory patterns (trajectories) on speech data [4] and more recently, convolutional non-negative matrix factorisation (CNMF) has been used as an approach to discover temporal (sequential) patterns in speech data [5]. CNMF showed a great deal of time warping variation and therefore time-coded NMF (motivated by findings in neuroscience) has been attempted to improve pattern discovery [5].

Attempts to explicitly model temporal effects (trajectories) in speech data have, to date, achieved limited success [6]. Specific limitations of the HMM modelling paradigm, in particular the state-based independence assumption, are addressed in these methods. This is mainly accomplished by either incorporating explicit trajectories within the HMM framework [7] or by defining longer-term variable-length segmental models [8].

In a novel approach to implement a hidden trajectory model, bi-directional filtering of vocal tract resonances (VTR) yields promising results and also enables the implementation of variable-length representation of long-contextual-spanning speech effects) [9]. Conceptually, the opposite approach is to model the trajectories of the features used for speech recognition directly, and in [10], [11] it was found that such models of cepstral features are able to represent co-articulatory phenomena in a way that makes context dependency explicit. The current paper similarly models the cepstral trajectories directly, and demonstrates how more accurate parameter fits can be achieved by using more sophisticated transition models.

III. Approach

The piece-wise linear approximation that we use to track cepstral trajectories effectively captures temporal changes using sub-phone level segments (as opposed to the individual frames) for every phone transition. Applying a search to find variable-length positions for these segments allows us to characterise detailed transitional behaviour and obtain a direct comparison between the modelled trajectories and the actual speech data. By measuring how consistent the tracked changes are, different modelling choices may be compared, leading to new insights regarding cepstral transition behaviour.

A. Cepstral transition models

We model speech data using MFCC features, which are widely used in state-of-the-art speech recognition systems. Near phone transitions, co-articulatory effects on these features have been shown to be highly regular in [10]. In particular, the phones on either side of a transition generally determine a target value (which the trajectory may or may not reach), and the trajectories generally interpolate fairly smoothly between those targets. The authors of [11] utilised this finding, describing individual phone transition behaviour with a simple piece-wise linear approximation model. Their model consisted of three line pieces to fit the cepstral values (frames) of a single MFCC (cepstral transition), using least-squares optimisation. Start and end line segments were constrained to be constant values. We refer to these constant line segments as *stable values* and the remaining central line segments as the *change descriptor*. To find a complete piece-wise linear approximation for any cepstral transition, a search is required to determine the start and ending indices (model alignments) of the change descriptor. Similarly to the method described in [11], the squared error for all line pieces of the cepstral transition model can then be found, yielding a single error value for each approximation. Optimising the squared error enables us to find the best model alignments. In order to compare the different options, the squared errors (SE_f) of each parameter at each instant are estimated, followed by the mean square error (MSE_{model}) across features:

$$SE_f = |t(x_f) - y_f|^2 \quad (1)$$

where $t(x_f)$ is the trajectory value at frame x_f and $|t(x_f) - y_f|^2$ is the squared residual.

$$MSE_{model} = \frac{1}{F}\sum_{f=1}^{F} SE_f \quad (2)$$

In [11] an algorithm is described that allows the piece-wise linear model to share contextual information with other (similar) transitions. By constraining the stable values to reference value estimates of different context sizes, the context dependency of these models can be evaluated. Similarly, what constitutes a single "cepstral transition model" can be specified according to context length, phone identities or even broad classes of phones.

B. Model evaluation

Our first priority in modelling phone transitions is to accurately represent speech data. This will then subsequently serve as the enabling factor, so that systematic effects (if they are present) may be identified. In terms of the models described here, we analyse two main criteria to facilitate these goals. The first measurement (model fit) is used to evaluate the ability of a model to track observed trajectories. Secondly we characterise individual cepstral transitions by evaluating:

- The consistency of a measurement across multiple samples of the same transition in a data set.
- The ability of the model to predict parameters of unseen samples (we estimate transition model parameters on a training set and evaluate the error on a separate test set).

1) Model fit: In Figure 1 the linear approximations for the first four cepstra of a single diphone transition example can be seen. The separate model parts of the first cepstral coefficient (MFC 1) can clearly be identified: two stable values (frames $9-15$ and frames $17-21$) and a single change descriptor (frame 16, connecting the stable values). As this is a segmented model, the stable values are anchored to the start and ending

frames of the diphone segment (frames 9 and 21 respectively) and do not extend to adjacent transition frames numbers ($1-8$ or $22-26$). For all cepstra a single definite transition is observed near the ASR boundary.

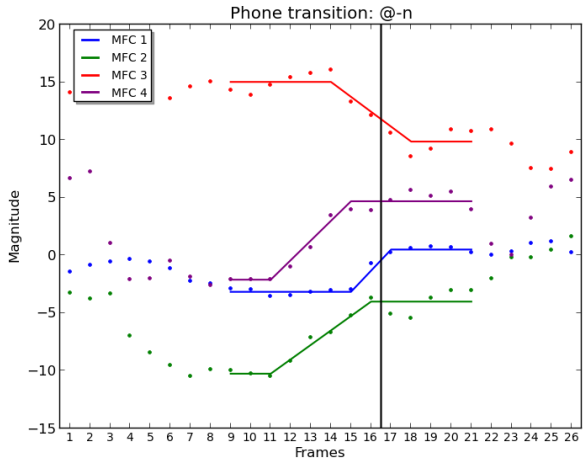


Fig. 1. Piece-wise linear model fit of the first four cepstra of the diphone transition /@-n/ using 3-piece segmented models

Through Equation 2, the MSE measurement can be calculated for the separate model parts, or for the whole piece-wise approximation of the specific coefficient and multiple transition examples, by including the relevant frames. To measure how well different trajectory estimation approaches compare with respect to the actual observed MFCC feature vectors, the MSE measurement (MSE_{trans}) of trajectories is particularly useful. This value allows the direct comparison of phone transitions, with regard to the training data, across all cepstral transition models. The MSE_{trans} measurement can be calculated as

$$MSE_{trans} = \frac{1}{\sum_{s=1}^{S} CF_s} \sum_{s=1}^{S} \sum_{c=1}^{C} \sum_{f=1}^{F_s} SE_{fcs} \quad (3)$$

where SE_{fcs} is the squared error for a specific frame f, a specific coefficient c and a specific sample s.

Every transition generates F squared errors (one for every frame) and there are $C = 13$ of these cepstra (one for every MFCC coefficient). To analyse the parameters for all of the examples (S) of a given class, the mean and standard deviation are calculated for the binned trajectories of the same MFCC coefficients.

Finally, to represent the entire set of transitions with a single error value, the summation of the contributions from each class is evaluated:

$$MSE_{global} = \frac{1}{T} \sum_{t=1}^{T} MSE_{trans}, \quad (4)$$

where MSE_{trans} are the mean trajectory MSE estimated for S examples of a contextual class and a total of T classes.

2) Model consistency: To identify systematic behaviour for cepstral transition trajectories, we present two consistency measurements, in which both stable values and change-descriptors are evaluated. Different modelling options can then be compared directly (for the same transition examples). More consistent model parameters are a more favourable choice for the representation of the transition model.

Reference stable values are estimated using the training data set. These values are obtained in a similar way to that described in [11]. Once an initial set of trajectories have been fitted to the training data, the mean is estimated for the stable (constant) parts of every particular context that is required. After estimating the reference stable values, these values can also be predicted for the unseen samples of the test set. We evaluate the model fit (as described in III-B1) in order to compare the trajectories obtained with predicted stable values.

For the measurements described here, change-descriptor model parts are treated differently. We choose to determine change-descriptor behaviour in terms of temporal information and define two representative parameters to evaluate the consistency: (1) Relative position to ASR boundary and (2) Absolute duration.

During the speech segmentation process, a single ASR boundary for every phone transition is obtained. This boundary has the same location for all 13 cepstra and is useful to provide an initial alignment of similar transition examples. In this way, we compensate for the fact that not all examples are equal in length. Measuring the centre positions (exactly half way between the model boundaries) of the change descriptors and relative to the ASR boundary then provides a good indication to the position where most of the change for a cepstral transition is occurring. The absolute duration is the length of the change descriptor as defined by the model alignments. Both position and duration measurements are given in terms of frame units.

For each cepstral transition class, we estimate:

$$\bar{x}_{cep} = \frac{1}{N} \sum_{n=1}^{N} x \quad (5)$$

and

$$\sigma_{cep} = \sqrt{\frac{1}{N} \sum_{n=1}^{N} (x - \bar{x}_{cep})}, \quad (6)$$

where x is the measured parameter value, $\bar{x}_{cep}$ the mean and σ_{cep} the standard deviation for N examples of the cepstral transition class. To represent an entire set of cepstal transitions with a single consistency value, we sum the contributions from each class:

$$C_{global} = \frac{1}{T} \sum_{t=1}^{T} \sigma_{cep}, \quad (7)$$

where σ_{cep} are the standard deviations estimated for N examples cepstral transitions and a total of T classes.

C. Cepstral model improvements

In order to gain a better understanding of the trajectory model, and to improve its capabilities, we have refined the basic model along a number of dimensions. These refinements are described below.

1) Connecting model segments: In the standard model, each segment is modelled separately. This means that stable value estimates of two adjacent models will not necessarily be the same, but could exhibit a 'gap'. We extend the piece-wise linear approximation algorithm to model the entire utterance in a single process, eliminating this artificial gap by forcing adjacent stable values to be equal to one another.

2) Predicting stable values with constrained alignments: Trajectory models with fixed reference stable values behave differently with regard to the model alignment algorithm than free trajectory models. An intermediate option would be to first fit free trajectories (to find transitions), then enforce stable values (from predicted reference values). This combined information then constitutes the final trajectory.

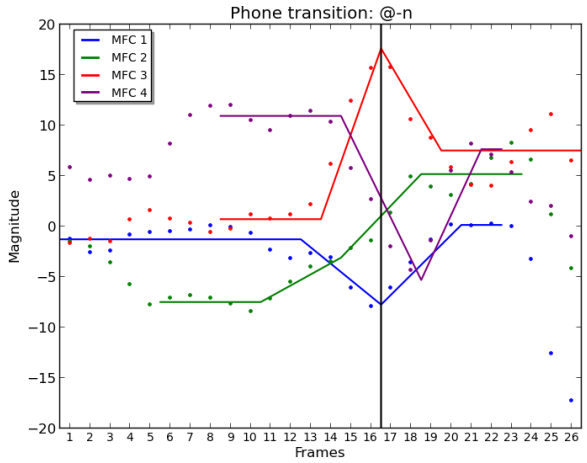


Fig. 2. Piece-wise linear model fit of the first four cepstra of the diphone transition /@-n/ using 4-piece connected models

3) 4-piece models: In Figure 2 another example of the same diphone transition as shown in Figure 1, but in a different context, is shown. While some of the cepstral transitions are seen to be moving (relatively) in similar directions and some start and end positions seem to agree, it is clear that the transition itself behaves rather differently. Instead of single transitional changes, characteristic peaks and troughs are now formed. This behaviour is seen quite frequently for certain transitions and coefficients. It is clear that in such cases, a more elaborate change descriptor could be of value to model the change accurately.

To improve change descriptor representation, we implement a 4-piece symmetrically constrained model. With this configuration, the change descriptors consist of two line pieces, which are kept at equal length. (This requirement drastically reduces the search space to find model alignment, compared to the requirement when any of the four line pieces could be of arbitrary length, and may be more robust in specifically detecting the peaks and troughs). In Figure 2 the linear approximations with 4-pieces are estimated for the first four cepstra. Since this is also a connected model, the stable value parts are now shared with the adjacent transition models (e.g. the stable value of the first cepstral coefficient, MFC 1, is now fitted to frames $1-12$ with this diphone transition example only beginning at frame 9).

IV. Experimental set-up

A. Overview

The experiments of this paper are performed on phone transitions that are selected to ensure that data scarcity does not interfere with our investigation. (Although we eventually want to apply our model in limited-data environments, our current goal is to understand its description of speech features in the absence of such a constraint.) In this section, we provide a discussion of the selection process. Each phone transition is selected from a high quality set of speech recordings (of a single speaker) and reviewed acoustically before being included in the final data set. To model any phone transition, specific MFCC features and the appropriate speech segmentation are derived. We also present the specifics of the features used to model the cepstral transitions.

B. Speech data

About 6000 short utterances were recorded for the experiments that we conduct. This provides a large corpus of high quality speech of a single male speaker. Only considering a single speaker allows us to focus on contextual effects first, without inter-speaker differences complicating the results. The recordings were made using a list of short Afrikaans prompts (1 to 5 words in length) with balanced phonetic coverage [12]. Additionally, a dynamic programming scoring algorithm was used with initial acoustic models to verify the speaker's pronunciations and obtain a high quality (aligned) set of recordings [13]. The number of utterances that showed perfect alignment was 4974 and had a total duration of about 3 hours.

From this "clean" data set, training and test data sets were selected. All diphone transitions that occur 30 or more times in the clean data set were retained, and greedy selection was used to select test utterances until the test set contained at least 3 examples of each of these diphones. The remaining clean utterances formed the training data set. After performing these steps the total number of utterances that were selected totalled 902 and 4072 for the test and training data sets, respectively.

C. Segmentation

Accurate identification of phone transition boundaries is very important, since our modelling approach relies on these boundaries. We use a standard HMM-based ASR system trained on all 4974 clean recordings to automatically align the speech data. A context-dependent cross-word phone recogniser with tied triphone models is employed; 39 MFCC features are used, which include the first 13 and their first and second

order derivatives. These features are computed with a window size of 25ms and a frame rate of 10ms. Semi-tied transforms are applied. Each triphone model has 3 emitting states with 7 Gaussian mixtures per state and a diagonal covariance matrix. Verifying phone recognition accuracy on the test set, using a flat-phone grammar yields a value of 92.71%.

Triphone model alignments are obtained using a forced alignment on all the data and the model alignment labels are then converted to the base label sequence (the actual phonemes observed in the training data).

D. Features for transition modelling

After transition boundaries have been obtained, we extract 13 MFCCs features for transition modelling. For these features, while we still use a window size of 25ms, the frame rate is adjusted to 5ms. This provides us with better time resolution. Only the raw MFCC coefficients are used and not any of the derivatives. Finally, for every utterance, each of the MFCC vectors is associated with the phone-boundary alignments from above, which provides contextual labelling at the triphone level.

E. Selection of transition examples

Transitions	All data	Train	Test
Total number	783	769	678
> 30 examples	470	436	173
Final selection	331	331	331

TABLE I
Number of unique diphone transition labels in data sets for various selection stages.

Given the test and training data sets, a further selection process was used to select the data for our experiments. In Table I, the total number of unique transition labels is given to show that for a large number of labels (470) we have more than 30 examples. (For the transition model analysis all transitions with fewer examples are ignored.) After excluding transitions including the silence label, we perform a final (per example) selection. A particular transition example is only allowed if the duration (in frames) is no more than a single standard deviation from the mean. The result of this selection provides us with the 331 most frequent transition labels and transition examples that have low speech rate variability.

V. Experiments and results

We compare various trajectory tracking techniques, reporting on the results obtained with each of the possible improvements described in Section III-C. For all of the model options, the MSE_{global} values are calculated to measure overall effectiveness. In the case of connected models, we always convert to a valid segmented representation, which ensures that direct comparison of the phone transitions on a per-segment basis is valid. Model options with predicted stable value parts require a train and test data set to assess trajectory tracking. Reference stable values are predicted using the training data and then applied as fixed stable value fits during model estimation on the transitions of the test data set.

To compare change-descriptor behaviour we estimate the global consistency values C_{global} of specific temporal parameters. More detailed comparison can be obtained (on a cepstral level), comparing the standard deviation σ_{cep} for the same cepstral transitions.

A. Connecting segments

As mentioned in Section III-C3, correctly representing the more stable parts of phone transitions given the imposed models requires us to extend the piece-wise linear approximation algorithm. Now, an entire utterance must be represented by a single piece-wise approximation. Finding the utterance-level trajectory model is accomplished in two estimation steps (adding to the definition in Section III-A):

- Locate the model alignments for all of the transitions in the utterance for segmented models.
- Use model alignments to fit all required line pieces of the utterance. On a per-segment basis (left-to-right), fit the change descriptor and ending stable value (except for the first transition), re-using the last stable value of the previous segment as the first stable value of the current segment.

Additionally, if fixed reference stable values are required, fit the mean of the two reference stable values contributing to a single shared stable value. By sharing the first stable value of the previous segment (transition), the segmented models are connected to form a single trajectory for the whole utterance.

Table II shows all of the estimated MSE_{global} values. Global MSEs are estimated for the phone transitions of different data sets (train and test) and two values are given per measurement: the means and standard deviations (in brackets) of the diphone transition class MSEs, respectively. Only free trajectories are constructed for the training data set. The global MSE for the test set, however, are compared for all options (free or fixed stable value trajectories). To aid the comparison, a ratio is also determined between the global MSE values with every fixed stable value trajectory option and its corresponding free trajectory. Finally, separate model parts can be evaluated, and the global MSE values for only the frames corresponding to the stable value of change-descriptor parts of trajectory models are given.

We observe that the error on the training set is in agreement of the test set results for free trajectories. There is a cost to connecting segments: Overall the error increases (as can be expected for the more constrained model). However, the ratios of error between fixed stable value and free trajectories are similar, and we see that the error increases at least five-fold when predicting stable values (rather than estimating them on each phone occurrence).

As previously observed [11], larger context sizes allow for more specific stable values and improved model fit. Therefore, we test reference stable values of different context sizes (monophones, biphones and triphones) and find that predicted stable value model fits improve up to the triphone context

Model	Stable reference value	Global MSE (train)	Global MSE (test)	Ratio (with free fit)	Global MSE (stable values)	Global MSE (change)
3-piece segmented	Monophone		24.553 (6.338)	6.803	27.636 (6.662)	8.861 (5.165)
	Biphone		19.228 (4.223)	5.328	21.908 (4.447)	6.756 (2.979)
	Triphone		19.118 (3.574)	5.297	21.961 (4.105)	6.974 (2.141)
	No ref	**3.604 (1.265)**	**3.609 (1.306)**		**4.047 (1.448)**	**1.821 (0.794)**
3-piece connected	Monophone		47.659 (8.232)	6.196	54.594 (10.458)	18.354 (5.893)
	Biphone		43.595 (7.319)	5.668	50.038 (9.503)	17.339 (4.226)
	Triphone		42.157 (7.064)	5.481	48.826 (9.219)	17.038 (3.512)
	No ref	**7.710 (2.335)**	**7.692 (2.433)**		**8.299 (2.667)**	**5.415 (1.837)**
4-piece connected	Monophone		22.788 (3.614)	5.450	38.089 (6.691)	13.482 (2.362)
	Biphone		21.497 (3.410)	5.142	35.968 (6.109)	12.461 (2.107)
	Triphone		21.398 (3.453)	5.118	36.246 (5.978)	12.303 (2.220)
	No ref	**4.211 (1.243)**	**4.181 (1.242)**		**4.959 (1.481)**	**3.427 (1.106)**

TABLE II

Overall MSE_{global} measurements for train and test data trajectories, including options with predicted stable values.

level. Finally, consistency measures of the mean position of the change descriptor and the mean duration of the change descriptor show similar distributions for 3-piece segmented and connected models.

B. Aligned transitions

Once stable values have been estimated, the timing of the change descriptor of a specific transition is determined by finding the best fit from one stable value to another. This may not produce optimal change descriptor alignment, especially if a specific stable value does not suit a specific sample of a transition well. Change descriptor alignments can be constrained to the free trajectory alignments for better change detection. The fixed stable values can then still be applied without allowing the model to find a further optimal fit for chosen parameters.

In Table III, a global free trajectory baseline consistency (C_{global}) of the change descriptor centre position is estimated. Since trajectories with fixed stable values only exist for the test data, all comparisons are made for the transitions of the test set. We find that the measured change descriptor position is less consistent for trajectory models with fixed stable values (free trajectory models show most consistent change descriptor positions in all cases).

The more consistent change descriptor positions of the free trajectory models motivate further investigation of free trajectory alignments. To better understand the relationship between reference stable values, free trajectory alignments and model fit, we also determine the MSE parameters when constraining fixed stable value trajectory models to have free trajectory alignments. Similarly to the values in Table II, Table IV shows the MSE_{global} values, now with free trajectory alignments. As expected, the overall MSE measurements show increased error for constrained alignments. Since 3-piece model change-descriptors are so dependent on stable values we find substantial error increases when comparing the values of Table II.

Model	Stable reference value	Position (centre)
3-piece segmented	No ref	2.847 (1.215)
	Biphone	4.095 (1.753)
	Triphone	4.107 (1.753)
3-piece connected	No ref	2.848 (1.215)
	Biphone	3.924 (1.708)
	Triphone	4.024 (1.721)
4-piece connected	No ref	2.167 (0.751)
	Biphone	2.289 (0.832)
	Triphone	2.281 (0.827)

TABLE III

Overall consistency C_{global} measurement of change descriptor position on test set

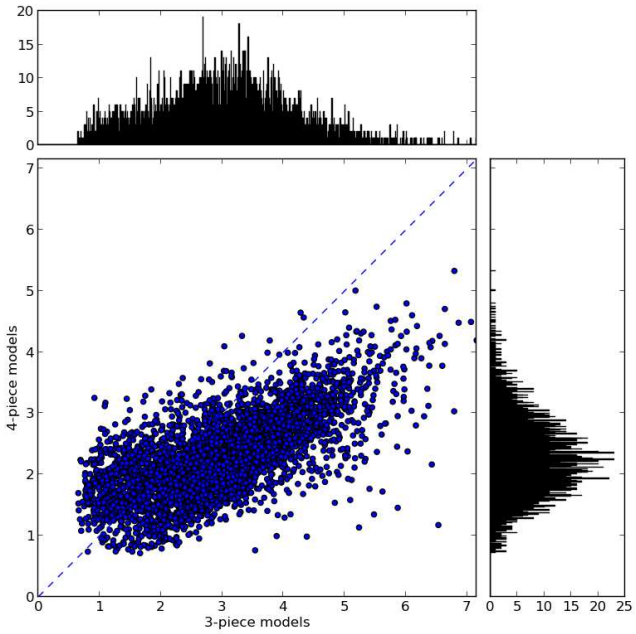


Fig. 3. Comparing consistency σ_{cep} of change descriptor position on a per cepstrum basis

C. 4-piece segments

For all the previous trajectory options, a change descriptor consisted of a single straight line, connected to the start and end points of the stable values (at the model alignments). With 4-piece models, the complexity of the change descriptor is

Model	Stable reference value	Global MSE	Global MSE (stable values)	Global MSE (change)
3-piece connected	Monophone	50.245 (8.563)	54.156 (9.826)	36.059 (7.502)
	Biphone	45.429 (7.315)	49.102 (8.512)	32.014 (5.904)
	Triphone	44.521 (7.057)	48.169 (8.161)	31.199 (5.870)
4-piece connected	Monophone	32.890 (5.485)	50.626 (8.655)	15.527 (2.696)
	Biphone	29.856 (4.771)	45.772 (7.339)	14.268 (2.409)
	Triphone	29.373 (4.888)	45.001 (7.151)	14.002 (2.521)

TABLE IV
Overall MSE_{global} measurement on test set, when applying fixed stable values and constrained alignments

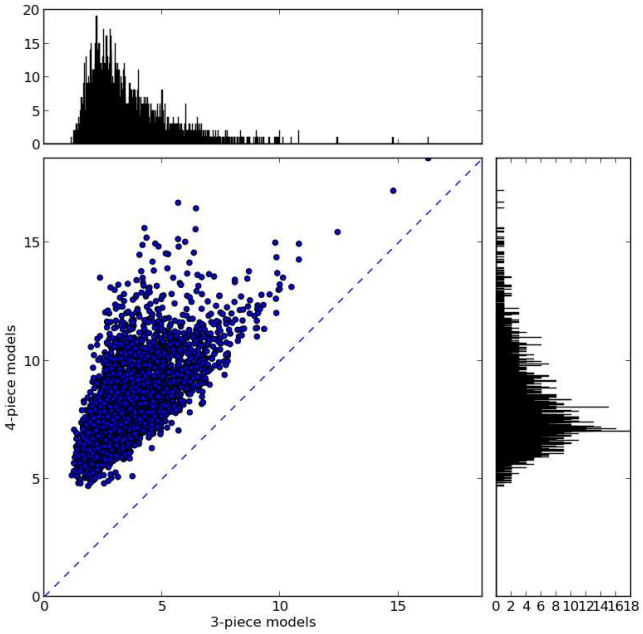


Fig. 4. Comparing mean duration $\bar{x}_{cep}$ of the change descriptors on a per cepstrum basis

increased to include two straight lines. This allows the change descriptor to have a freely varying centre point (connecting the two change descriptor line pieces). As a final constraint, the change descriptor must be symmetrical along the time axis (two lines of equal duration) during model alignment. Final model fits (when connecting segments) of the utterance may however find "shared" stable values different to the ones used during model alignment, leading to non-symmetric change descriptors.

Comparing the overall consistency C_{global} of the change descriptor position with that of the 3-piece models shows the 4-piece models to be the most consistent choice for free trajectory models (Table III). Furthermore, rather than becoming much less consistent when trajectories with fixed stable values are used, 4-piece models show comparable consistency for fixed stable value trajectories.

Figure 3 provides a more detailed comparison. Measuring the centre position of the change descriptors relative to the ASR boundary for each cepstral transition example, and computing the standard deviation σ_{cep} allows transition comparison on a per cepstrum basis. The scatter plot therefore depicts these values for 3 and 4-piece models and the same transition examples. We find that most of the cepstral transitions have larger standard deviations when 3-piece models are used. According to the histogram frequencies and the placement of cepstral transition measurements, only a relatively small number of cepstral transitions have smaller standard deviation for 3-piece models. Generally 4-piece models also tend to have lower standard deviation for most of these cepstral transitions.

Model	Duration (absolute)
3-piece segmented	2.710 (0.858)
3-piece connected	2.583 (0.876)
4-piece connected	2.842 (0.930)

TABLE V
Overall consistency C_{global} measurement of change descriptor durations on all data

To understand more about the differences between the 3-piece and 4-piece change descriptors, we also compare their absolute durations (length in frames). Figure 4 shows the mean duration in frames compared for every cepstral transition class between 3-piece and 4-piece models. It is clear that for all cepstral transition classes, the mean durations of the change descriptors are longer for 4-piece models compared to 3-piece models and the same class.

Confirming the overall variability C_{global} on the mean duration (free trajectories), we find that connecting segments for 3-piece models seems to provide the most consistent mean change descriptor durations in general (Table V). Although 4-piece models with longer change descriptors are less consistent, this value is still very comparable to the 3-piece model case.

Finally, the overall MSE_{global} values in Table II confirm that the additional freedom of the 4-piece model reduces overall error by considerable amounts; this is true for both the model fit of change descriptor and stable value parts, as well as trajectories with predicted stable values compared to 3-piece models of similar configuration. The ratio of the MSE for trajectories with predicted stable values and free trajectories is also seen to improve for 4-piece models.

Additional insight regarding the predictability of reference stable values may be achieved by exchanging ("swapping") the matching reference values between 3 and 4-piece models. Table VI shows the MSE_{global} values for the different context

Model	Stable reference value	Global MSE	Global MSE (stable values)	Global MSE (change)
3-piece connected	Monophone	47.860 (8.270)	54.505 (10.367)	18.474 (6.062)
	Biphone	43.701 (7.381)	49.706 (9.341)	17.306 (4.309)
	Triphone	42.638 (7.440)	48.940 (9.445)	17.159 (3.738)
4-piece connected	Monophone	23.095 (3.708)	38.847 (6.920)	13.518 (2.393)
	Biphone	22.070 (3.612)	37.100 (6.586)	12.665 (2.184)
	Triphone	21.906 (3.597)	37.245 (6.375)	12.465 (2.271)

TABLE VI
Overall MSE_{global} on test set for "swapping" reference stable values between 3 and 4-piece models

sizes. Improved model fit of stable regions for 3-piece models (using the 4-piece predicted stable values) are obtained, in all cases except the triphone case. Similarly, the 4-piece model model fit for these regions degrades in all cases. Overall model tracking degrades slightly in all cases.

VI. Conclusion

With this work we improve upon the piece-wise linear model approximation of cepstral transitions. This is accomplished by the introduction of new approximation options (connecting segments, constraining model alignments and more complex change descriptors). Trajectory model tracking is analysed in more detail and for separate model parts (change descriptors and stable values). We find that connecting segments, to form a single linear approximation for the entire utterance, proves to be successful and leads to similar distributions for the change descriptors. Although we do obtain similar context dependent improvements to [11] for predicting stable values, these predictors are confirmed not to be very accurate representations of the actual magnitudes for frames of the stable regions of individual transition examples.

Our analysis of change descriptor behaviour shows free trajectories to be the most consistent at detecting the relative position of change. Change descriptor behaviour is tightly coupled to the chosen stable values for 3-piece models and are therefore strongly affected, introducing large error, for aligned model fits. In contrast, the extra degree of freedom for the change descriptor of the 4-piece model is seen to be much less dependent on the stable value parts, resulting in comparatively consistent positions of detected changes. Further examination of the change descriptors shows the 4-piece approximation to model much longer changes in general, which agrees with plots of cepstral transitions where characteristic (longer) double transition behaviour can frequently be observed near the ASR boundary for some cepstra and phone transition labels. This also implies that fewer frames are assigned to stable regions.

In spite of these factors, the error in the stable values of the 4-piece approximation increases substantially for constrained alignments and is fairly similar to the stable value error for the 3-piece model case. Swapping the predicted stable values between 3 and 4-piece models also generate similar error for these parts, with slight improvement when using the 4-piece predictors. The exact reason why these regions show so much intra-segmental variability is not yet well understood and additional investigation may prove valuable.

References

[1] V. Digalakis, "Segment-based stochastic models of spectral dynamics for continuous speech recognition," Ph.D. dissertation, Boston University, 1992.

[2] W. Holmes and M. J. Russell, "Probabilistic-trajectory segmental HMMs," *Computer Speech and Language*, vol. 13, no. 1, pp. 3–37, January 1999.

[3] K.-F. Lee, "Large-vocabulary speaker-independent continuous speech recognition: The sphinx system," Ph.D. dissertation, Carnegie Mellon University, 1988.

[4] L. Deng, G. Ramsay, and D. Sun, "Production models as a structural basis for automatic speech recognition," *Speech Communication*, vol. 33, no. 2-3, pp. 93–111, 1997.

[5] H. V. hamme, "An on-line NMF model for temporal pattern learning. theory and application to automatic speech recognition," in *Proc. International Conference on Latent Variable Analysis and Signal Separation (LVA/ICA)*, 2012, pp. 306–313.

[6] K. Sim and M. Gales, "Discriminative semi-parametric trajectory model for speech recognition," *Computer Speech and Language*, vol. 21, no. 4, pp. 669–687, October 2007.

[7] K. Tokuda, H. Zen, and T. Kitamura, "Trajectory modeling based on HMMs with the explicit relationship between stochastic and dynamic features," in *Proc. Eurospeech*, September 2003, pp. 865–868.

[8] M. Ostendorf, V. Digalakis, and O. Kimball, "From HMMs to segment models: A unified view of stochastic speech recognition," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 4, no. 5, pp. 360–378, May 1996.

[9] D. Yu, L. Deng, and A. Acero, "A lattice search technique for a long-contextual-span hidden trajectory model of speech," *Speech Communication*, vol. 48, no. 9, pp. 1214–1226, 2006.

[10] J. Badenhorst, M. Davel, and E. Barnard, "Analysing co-articulation using frame-based feature trajectories," in *Proc. PRASA*, November 2010, pp. 13–18.

[11] J. Badenhorst, M. Davel, and E. Barnard, "Trajectory behaviour at different phonemic context sizes," in *Proc. PRASA*, November 2011, pp. 1–6.

[12] N. de Vries, J. Badenhorst, M. Davel, E. Barnard, and A. de Waal, "Woefzela - an open-source platform for ASR data collection in the developing world," in *Proc. Interspeech*, August 2011, pp. 3177–3180.

[13] M. Davel, C. van Heerden, N. Kleynhans, and E. Barnard, "Efficient harvesting of internet audio for resource-scarce ASR," in *Proc. Interspeech*, August 2011, pp. 3153–3156.

Acoustic model optimisation for a call routing system

Neil Kleynhans*, Raymond Molapo* and Febe de Wet*†
*Human Language Technologies Research Group Meraka Institute, CSIR, South Africa
*†Department of Electrical and Electronic Engineering, Stellenbosch University, South Africa
Email: {ntkleynhans,rmolapo,fdwet}@csir.co.za

***Abstract*—The paper presents work aimed at optimising acoustic models for the AutoSecretary call routing system. To develop the optimised acoustic models: (1) an appropriate phone set was selected and used to create a pronunciation dictionary, (2) various cepstral normalization techniques were investigated, (3) three South African corpora and multiple training data combinations were used to train the acoustic models, and, (4) model-space transformations were applied. Using an independent testing corpus, which contained proper names and South African language names, a named-language recognition accuracy of 95.11 % and proper name recognition accuracy of 93.31% were obtained.**

I. Introduction

Interactive voice response (IVR) systems are widely used by companies to automatically assist their clients. The automation of services can greatly reduce company costs and in certain instances can be used by company staff to improve their productivity. Through Dual Tone Multi-Frequency (DTMF) keypads and Automatic Speech Recognition (ASR), IVR systems can capture digit information (such as account numbers) and more sophisticated information via a person's speech (e.g. person's name and surname). Unfortunately, DTMF input has an innately low information carrying capacity which is largely limited to digit-centric information. To overcome DTMF shortcomings, adding a natural spoken input and ASR information extraction capability can greatly increase the versatility of an IVR system.

A typical IVR application that makes use of speech processing capabilities is a call routing service, i.e. a system that routes incoming calls automatically to appropriate services or individuals. One such system is the AutoSecretary system introduced by Modipa *et. al.* [1], which routes incoming calls to a person based on a spoken name.

In this paper we describe the development of acoustic models for the AutoSecretary IVR application. Specifically, we focused on acoustic model optimisations which would:

- enable the system to route calls to an operator based on the callers language preference, and,
- allow new names to be added to the system relatively easily.

The next Section II describes the AutoSecretary system and provides background on some application-specific ASR issues. Section III details the ASR development effort as well as corpus selection and design. Our experiments are described in Section IV and results and a discussion are presented in Section V. Lastly, the conclusion and possible future work appear in Section VI.

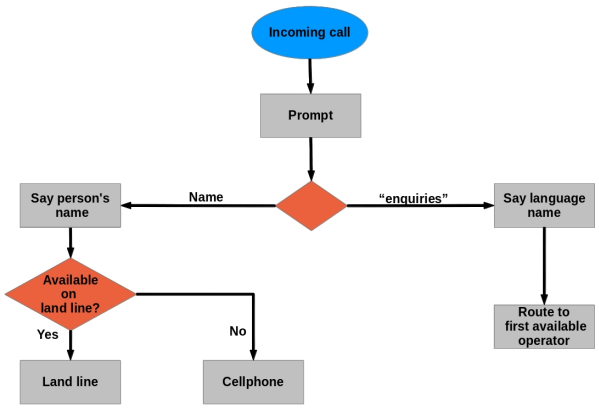


Fig. 1. High level AutoSecretary call flow.

II. Background

A. AutoSecretary IVR System

Figure 1 shows the high level call flow of the AutoSecretary call routing system. At the beginning of a session the system prompts the caller to say the name of person they are looking for or the word "*enquiries*". Following a valid name request the system will route the caller to the registered land line number. In addition, the system has the ability to route to a mobile number if it could not make a connection via the land line. If the word "*enquiries*" was spoken instead of a name, the system prompts the user for a language option - any of the eleven official South African languages - which allows the system to route the call to an operator who speaks the requested language.

The simple confidence scoring method implemented by ATK [2], is used to make a decision to either accept the recognition output if the confidence score is high or re-prompt the user to repeat their request if the confidence score is too low. Following two successive re-prompts, the system will automatically route the caller to a default operator. Figure 2

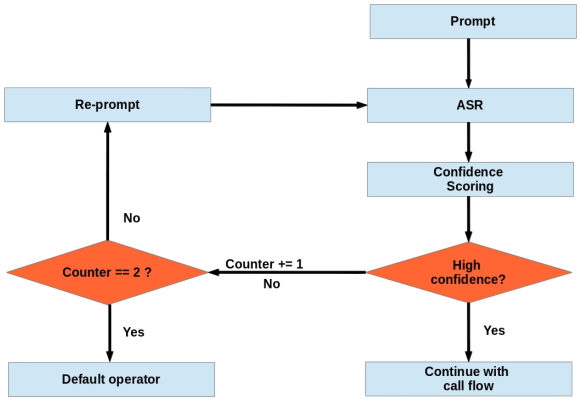


Fig. 2. AutoSecretary confidence scoring mechanism.

shows the AutoSecretary confidence scoring mechanism and how its application to the call flow.

On all successful recognitions, the system will parrot back the name to verify the selection. The caller may interrupt the transfer if the system selection is incorrect by saying "*stop*". The AutoSecretary system previously described by Modipa *et. al.* [1] had a similar call flow but did not include the functionality to route a user to an operator that spoke a particular language.

B. AutoSecretary ASR

The main issue in developing robust acoustic models for the AutoSecretary system is accurate proper name recognition. This type of problem has been encountered previously in directory assistance systems [3] and voice-navigation systems [4].

The first challenge in achieving accurate proper name recognition is robust pronunciation modelling. Initially, a phone set that can effectively represent the speech acoustic space must be chosen. This becomes an important aspect when dealing with multilingual environments which, in general, contain many sound classes and require careful phone set selection. Another major problem is that the phonemic representation of a word and the way it is pronounced vary greatly [4]. A possible cause of this mismatch is that people are altering the way in which they are pronouncing the proper name [5] based on what they think the word should sound like. This generally happens when an unknown or foreign word has to be spoken and the speaker has no prior knowledge. In multilingual environments this problem increases and becomes more difficult to solve as more languages are added. A way in which to partly overcome this problem, is to add multiple pronunciation variants to the pronunciation dictionary [1]. Adding pronunciation variants is a manually intensive task but affords greater accuracy compared to automatic methods. Automatic methods, such as G2P, have been shown to work well for common words but extracting rules for proper names still proves to be difficult [6]. Accurate proper name prediction is made difficult because proper names do not have set ways of pronouncing them [4], which makes robust rule extraction hard to accomplish. Also, predicting foreign words adds to incorrect pronunciations [5].

The second challenge is to develop robust acoustic models. In the standard HMM paradigm, creating word-based Hidden Markov Models is infeasible. As reported in [7], in the United States alone there were an estimated 1.5 million surnames with a third of these being unique. In multilingual environments, such as South Africa, these numbers would increase drastically. Another point of failure for word-based HMMs is the effort required to add new names to the system. Thus, a better approach would be to follow a large vocabulary ASR system development cycle. Here, development corpus selection is important as one would require large amounts of data to train robust acoustic models. A benefit of large vocabulary ASR systems is that they allow easier modification of the recognition grammar - for instance adding language name recognition - which adds flexibility to the system. Collecting a corpus of names per application [1] would be impractical as this would not in general produce robust acoustic models. In addition, if one would require the system to be re-deployed, a time consuming audio data collection process would have to be run before the system can be reliably operated in a new environment.

Corpus Name	# utterances	duration in hours
Lwazi English	5843	5.03
Lwazi English plus Lwazi language prompts	7770	5.57
NCHLT English	106018	76.97
AST English (5 dialects)	51745	29.80

TABLE I
THE NUMBER OF TRAINING UTTERANCES AND DURATION FOR EACH DEVELOPMENT CORPUS.

III. ASR DEVELOPMENT

In this section we describe the speech corpora used for acoustic model development, the phone set selection and pronunciation dictionary creation, the feature extraction process, acoustic model development as well as the recognition grammar and concept mapping that were used during system evaluation.

A. Training Corpora

To enable robust acoustic model development in a multilingual South African context we focused on three South African corpora. Table I shows the number of training utterances per corpus and indicates the duration in hours.

1) Lwazi: The Lwazi corpus contains annotated telephony speech data covering eleven South African languages [8], [9]. Each language-specific corpus was produced by collecting read and elicited speech data from approximately 200 speakers; with each speaker contributing roughly 30 utterances [9]. A portion of the utterances were randomly selected from a phonetically balanced corpus and the remainder are words or short phrases. Importantly, each corpus contains utterances

which captured the response of the speakers when queried about their first language.

2) NCHLT: The NCHLT ASR corpus contains annotated high-bandwidth speech data collected for eleven South African languages [10]. The individual corpora contain a minimum of 50 hours of speech data collected from 200 speakers (gender-balanced) with each speaker contributing in the order of 500 utterances. The volume of collected data improves triphone coverage and should make it easier to add new names or short phrases to the recognition grammar.

3) AST: The African Speech Technology (AST) corpus contains annotated telephony speech data for five South African languages [11]. The speech data was collected from 300 - 400 speakers and the prompts were chosen to support information retrieval, transactional teleservices and hotel booking applications. Given the prompt design, the corpus contains a large proportion of proper names and a good coverage of language prompts. Additionally, the English corpus contains data collected from five common South African English accents which should add to the robustness of the acoustic models. In the current investigation the same train, development and test sets were used as those described in Kamper *et. al.* [12].

B. Testing Corpus

A testing corpus was developed by, firstly, expanding the recognition grammar to create text prompts and then collecting speech data from a variety of speakers. The testing corpus contained speech data from approximately 20 unique speakers with each speaker contributing 22 names- and 46 language-specific utterances. The data was collected from both land line and mobile handsets which represents a close approximation to the proper testing environment. After manual validation, the final utterance count was 555 names-related and 1003 language-related utterances. The duration of the testing audio at this point was 1.42 hours. To increase the testing data size further, we included a previously collected name-surname corpus which contained 31 unique name-surname pairs. The final testing corpus contained 2.13 hours of audio data, 1480 names-related and 1003 language-related utterances.

C. Phone Set and Pronunciation Dictionary

The initial phone set was a union of all the phones found in the Lwazi corpus [8] and consisted of 87 unique phones. These were then mapped to a simplified set of 62 phones where affricates were split (*e.g.* [tS] $\rightarrow$ [t] [S], [d_0Z] $\rightarrow$ [d] [Z]) and clicks and subtle phone distinctions merged (*e.g* [h\] became [h], etc.). The motivation for simplifying the phone set is that multilingual speakers will probably not pronounce the distinctions correctly, thus removing them from the start would be better.

The corpora-specific pronunciation dictionaries were mapped to the simplified 62 phone set. As the majority of the training corpora used in our investigation were South African English (SAE) the final phone set only contained 41 South African English phones. The reduction in the number of phones, is due to English not containing phones which occur in other languages. As a final phase, foreign words had phonemic representation generated manually using the closest English phones.

The recognition pronunciation dictionary or AutoSecretary dictionary contained 158 unique entries which included multilingual person and South African language names as well as a few English honorifics (ms, mr, mrs, dr). With pronunciation variants this count increased to 415. The 41 phones in the English set were used to manually create all the pronunciations.

D. Feature Extraction

39 (13 static, 13 delta and 13 delta-delta) dimensional Mel Frequency Cepstral Coefficient (MFCC) features were generated using the Application Toolkit (ATK) [2] and the Hidden Markov Model Toolkit (HTK) [13]. These MFCCs were extracted every 10 ms from a 25 ms speech frame. The frequency bandwidth was limited to 150-3600 Hz and is applied by HTK independent of sampling rate.

Channel normalisation was performed by means of cepstral mean normalization (CMN). Four different options were considered, i.e. *no CMN*, *HTK CMN*, *Global CMN*, and *ATK CMN*.

HTK CMN is implemented by estimating a cepstral mean vector on a per utterance basis and removing the cepstral vectors' offset [13]. The *Global CMN* method estimates a cepstral mean vector from the entire training data set and then uses the vector to normalize the training and testing cepstral vectors. *ATK CMN* is implemented by first loading an initial mean vector which, for our experiments, was a global mean cepstral vector estimated on the training data [2]. This cepstral mean is updated on every *speech* frame according to the formula:

$$\boldsymbol{\mu'} = \alpha(\boldsymbol{\mu} - \boldsymbol{x})\boldsymbol{x}, \tag{1}$$

where $\boldsymbol{\mu'}$ is the updated cepstral mean, α is the time constant set to 0.995, and $\boldsymbol{x}$ the input cepstral vector. For each utterance, ATK resets the mean cepstral vector to the initial mean vector $\boldsymbol{\mu_0}$. To determine whether a frame is speech, ATK uses the first 40 frames of each utterance to train a silence detector and performs a speech / non-speech analysis on each frame. The first 10 frames of an utterance are not used to update the mean cepstral vector.

When experimenting with a specific CMN approach both the training and testing data were normalized using the same CMN technique.

E. Acoustic modelling

A standard acoustic model development strategy was used as detailed in HTK book [13]. The acoustic models were tied-state context-dependent (triphone) Hidden Markov Models (HMMs), using a three state left-to-right topology. Question-based tying was used to create the tied-state models. Eight Gaussian mixtures per HMM state were used to model the cepstral densities. Different sets of acoustic models were

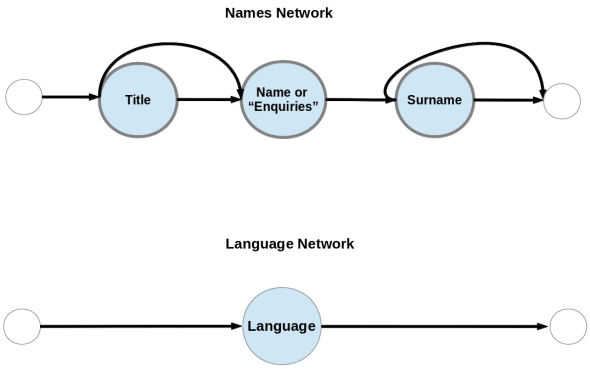


Fig. 3. The AutoSecretary name and language recognition networks.

created using the corpora described in Section III-A as well as using combinations of some of the corpora.

F. Recognition Grammar and Concept Mapping

The test set vocabulary contained 40 unique name-surname pairs and 11 unique language options. The recognition networks for names and languages are shown in Figure 3.

Expanding the recognition network provides 156 name and 46 language possibilities. The name network also contained the following words: “*enquiries*”, “*switchboard*”, “*reception*”, and “*stop*”.

During normal AutoSecretary operation the application is only aware of the unique name and language options and maps the expanded ASR text output. For example, the following mappings would be performed:

- “*Mr John Doe*” or “*John Doe*” mapped to “*John Doe*”
- “*Sesotho sa leboa*” or “*Northern Sotho*” mapped to “*Sepedi*”

For system performance evaluation we defined the various unique names and language options as “concepts” and performed “concept mapping” which reduced the expanded ASR recognition output to their unique name and language equivalents. When reporting the system results we report on “concept” accuracies unless otherwise stated.

IV. Experiments

A. Training data combinations and cepstral normalization

In addition to the English sub-corpora of the Lwazi, AST and NCHLT corpora, different combinations of the various corpora were used as training data. We defined the data combinations as follows:

1) *Lwazi English + Langs*: Training data pooled from the English sub-corpus of Lwazi and all language prompts from the remaining 10 language-specific corpora.
2) *Lwazi English + Langs + AST*: Training data pooled from (1) and the five AST English dialects.
3) *Lwazi English + Langs + NCHLT*: NCHLT English sub-corpus added to (1).
4) *Lwazi English + Langs then AST*: (1) was used to train single mixture tied-state HMMs. Then, the five AST English dialects data was added to the training data and used during mixture incrementing.
5) *Lwazi English + Langs then NCHLT*: Similar to (4) except that the NCHLT English sub-corpus was used instead of the AST data.

The four different options for channel normalization described in Section III-D (“*No CMN*”, “*Global CMN*”, “*HTK CMN*” and “*ATK CMN*”) were tested in combination with each of these training sets.

B. Semi-tied versus Constrained MLLR

HMM-based large vocabulary ASR systems generally use diagonal covariance matrices to reduce the number of model parameters. Full covariances matrices, however, are able to model the non-Gaussian nature of data which could potentially provide an increase in accuracy. Semi-tied transformations [14] transform diagonal matrices into full covariance matrices but instead of estimating state-specific transformations, estimate class-specific transforms. These classes are usually defined by a regression class tree which groups similar HMM states together [13]. In this way the parameter count may be kept relatively low which prevents excessive recognition times. The semi-tied transform is defined as:

$$\boldsymbol{\Sigma}^{(m)} = \boldsymbol{H}^{(r)}\boldsymbol{\Sigma}^{(m)}_{diag}\boldsymbol{H}^{(r)T}, \tag{2}$$

where $\boldsymbol{\Sigma}^{(m)}$ is the component-specific diagonal covariance matrix and $\boldsymbol{H}^{(r)}$ is the class-specific semi-tied transform. Unfortunately, ATK does not implement semi-tied transforms but does support constrained maximum likelihood linear regression (CMLLR) transforms [15], [13]. Constrained MLLR is typically used for speaker and channel adaptation and performs the adaptation by transforming the mean and covariance components in the HMM set. If one compares the semi-tied and the CMLLR transforms, the forms are quite similar. The CMLLR transform is defined as:

$$\hat{\boldsymbol{\mu}}^{(m)} = \boldsymbol{H}^{(r)}\boldsymbol{\mu}^{(m)} + \boldsymbol{b}^{(\boldsymbol{r})},\ \hat{\boldsymbol{\Sigma}}^{(m)} = \boldsymbol{H}^{(r)}\boldsymbol{\Sigma}^{(m)}_{diag}\boldsymbol{H}^{(r)T}, \tag{3}$$

where $\hat{\boldsymbol{\mu}}^{(m)}$ and $\hat{\boldsymbol{\Sigma}}^{(m)}$ are the transformed component-specific mean and covariance matrices, $\boldsymbol{\mu}^{(m)}$ and $\boldsymbol{\Sigma}^{(m)}$ are the original component-specific mean and covariance matrices and $\boldsymbol{H}^{(r)}$ and $\boldsymbol{b}^{(r)}$ are the class-specific CMLLR transforms.

Both methods iteratively solve for the transform parameters by optimising a modified Expectation-Maximization auxiliary function. The auxiliary functions, found in [14] and [13], highlight the differences in the equations which change the iterative optimisation equations. As a final experiment we wanted to determine whether CMLLR transforms could perform comparably to semi-tied transforms.

		System B	
		# Correct	# Incorrect
System A	# Correct	w	x
	# Incorrect	y	z

TABLE II
A 2X2 CONTINGENCY TABLE USED IN A MCNEMAR'S TEST.

C. McNemar's Test

McNemar's test can be used to establish whether the differences in error-rates, produced by two systems, are statistically significant [16]. This test requires that the errors produced by the system are independent events and in terms of speech recongition, can be used to test isolated-word recognition results [16]. The first step in performing a McNemar's test is to create a 2x2 contingency table as shown in Table II.

From this, we define the *null* and *alternative* hypotheses as,

$$\begin{aligned} H_0 : p_x &= p_y \\ H_1 : p_x &\neq p_y \end{aligned}$$

The test statistic is a one degree of freedom chi-squared distribution (χ^2) with Yates's correction for continuity [17] and is given by,

$$\chi^2 = \frac{(\mid x - y \mid -0.5)^2}{x + y}. \tag{4}$$

The null hypothesis can be rejected or accepted by calculating the two-sided P-value of the χ^2 distribution and comparing it to standard significance levels of 0.05, 0.01 or 0.001.

V. RESULTS AND DISCUSSION

In this section we present results on various training data combinations and cepstral normalization techniques which were used to perform acoustic model optimisations for the AutoSecretary system. We also show results around our hypothesis that CMLLR can be used as an approximate replacement for semi-tied transforms. Throughout this section the tables show concept accuracies (refer to Section III-F for a description) unless otherwise stated. Training data combinations and their labels are detailed in Section IV-A and cepstral normalization techniques are described in Section III-D.

A. Training data combinations and cepstral normalization

Table III shows language and name concept accuracies for different training data combinations, various training schemes and cepstral normalization techniques. Focusing on the cepstral normalization techniques (compare results within rows) we can see that some normalization methods produced surprising results. The "*HTK CMN*" produced the worst results which indicates that for this type of application utterance-based normalization is not ideal. This may be due to the short testing utterances which are often less than a second in duration and long-term biases are not estimated properly.

The "*No CMN*" and "*Global CMN*" results are quite similar which indicates that "*Global CMN*" did not perform effective normalization. In the majority of cases (except for language experiments using "*Lwazi Eng + Langs*" and "*Lwazi Eng + Langs then NCHLT*" data combinations) the ATK normalization proved to be the best cepstral normalization approach. The "*ATK CMN*" normalization method begins with the same initial mean cepstral vector as the "*Global CMN*" normalization but adapts the mean cepstral vector as it progresses through the utterance and only updates on speech frames. This selective adaptation seems to provide a good normalization mechanism. Previously it was observed by Modipa *et. al.* that there was a large discrepancy between the off-line and online ASR accuracies. A possible cause could be the differences in HTK and ATK cepstral normalization procedures.

Turning to the language recognition results and considering only our best normalization method (compare results within the *ATK CMN* column), we see that adding the Lwazi language prompts gave quite a large boost in performance, which was to be expected. Surprisingly, the *AST* and *NCHLT* only experiments produce rather poor results. In the case of *NCHLT*, this may be put down to a channel mismatch as the *NCHLT* corpus contains high-bandwidth audio data. More investigation is needed to establish why the *AST* data performed so badly. Combining data ("*Lwazi Eng + Langs + AST*", "*Lwazi Eng + Langs + NCHLT*") resulted in a slight increase in performance when adding the *AST* data but did not achieve any increase in accuracy when adding the *NCHLT* data. Training a system on the "*Lwazi English + Langs*" then adding *AST* for mixture incrementing produced the best results. It is interesting that state-tying on the smaller "*Lwazi English + Langs*" corpus resulted in an increase in performance. Further investigation is needed to determine why state-tying on a smaller corpus produced such an increase and to establish whether such a gain would be seen if the testing vocabulary was much larger. The last experiment, where *NCHLT* was used for mixture incrementing manage to achieve a slight increase in accuracy.

For name recognition (compare results within the *ATK CMN* column), the *AST* data and combinations with the AST data produced the top results with the "*Lwazi Eng + Langs then AST*" producing the best names recognition performance. The "*Lwazi Eng + Langs then NCHLT*" produced the best result out of the non-*AST* experiments but other *NCHLT* combinations performed marginally better or worse than the "*Lwazi Eng + Langs*" data combination.

B. Semi-tied versus CMLLR

In Section IV-B we speculated if it were possible to use CMLLR as a semi-tied replacement since ATK does not support semi-tied transformations. Table IV shows name and language concept recognition accuracies for various training data combinations and using either no, semi-tied or CMLLR transformation. ATK cepstral normalization was used for all the experiments.

If one compares the semi-tied and CMLLR columns of Table IV, for both language and name recognition results, we can see that the semi-tied approach outperforms CMLLR technique in the vast majority of the experiments (12 out of

	No CMN	HTK CMN	Global CMN	ATK CMN
Lwazi Eng	85.33 / 78.72	60.88 / 70.14	85.33 / 78.92	87.13 / 83.18
Lwazi Eng + Langs	93.11 / 80.27	68.66 / 75.88	93.01 / 80.27	92.81 / 84.19
AST	58.78 / 60.14	64.57 / 72.70	60.58 / 60.07	75.25 / 77.43
NCHLT	79.84 / 78.85	67.07 / 74.73	79.54 / 78.65	80.44 / 81.28
Lwazi Eng + Langs + AST	90.12 / 77.84	69.36 / 77.64	89.52 / 77.77	93.71 / 87.91
Lwazi Eng + Langs + NCHLT	89.92 / 82.36	73.35 / 78.85	90.82 / 82.16	92.81 / 84.46
Lwazi Eng + Langs then AST	86.53 / 90.27	75.85 / 84.80	85.83 / 89.93	**95.11 / 93.31**
Lwazi Eng + Langs then NCHLT	91.82 / 87.70	77.84 / 83.92	92.22 / 87.70	91.92 / 89.46

TABLE III
Language AND *Name* CONCEPT ACCURACIES (%) FOR VARIOUS TRAINING DATA COMBINATIONS AND CEPSTRAL NORMALISATION TECHNIQUES. THE RESULTS ARE PRESENT IN PAIRS - LANGUAGE ACCURACY % / NAME ACCURACY %.

	None	Semi-tied	CMLLR
Lwazi Eng	87.13 / 83.18	86.43 / 83.65	85.83 / 81.82
Lwazi Eng + Langs	92.81 / 84.19	93.21 / 84.26	92.81 / 83.72
AST	75.25 / 77.43	78.64 / 81.42	78.54 / 78.78
NCHLT	80.44 / 81.28	78.34 / 81.28	80.64 / 79.19
Lwazi Eng + Langs + AST	93.71 / 87.91	93.01 / 89.32	94.01 / 87.23
Lwazi Eng + Langs + NCHLT	92.81 / 84.46	93.71 / 84.32	93.51 / 83.31
Lwazi Eng + Langs then AST	**95.11** / 93.31	93.71 / **94.32**	94.91 / 93.65
Lwazi Eng + Langs then NCHLT	91.92 / 89.46	91.22 / 89.46	91.32 / 88.85

TABLE IV
Language AND *Name* CONCEPT ACCURACIES (%) FOR VARIOUS TRAINING DATA COMBINATIONS AND SEMI-TIED AND CMLLR TRANSFORMATION TECHNIQUES. THE RESULTS ARE PRESENT IN PAIRS - LANGUAGE ACCURACY % / NAME ACCURACY %.

16), however, the differences in accuracies are relatively small. To investigate whether there was any significant difference between the semi-tied and CMLLR results, McNemar's test was used to analyse the recognition outputs. Refering to the fourth column of Table V, we can see that only the "*Lwazi Eng*", "*AST*" and "*Lwazi Eng + Langs + AST*" experiments produced a significant difference in the results, if one chooses a significance level of 0.05. At a stricter significance level, 0.001, all the null hypothesis would be accepted, which implies that the semi-tied and CMLLR are quite similar.

Comparing the McNemar's test P-values, calculated between semi-tied and no transform (column two Table V) and CMLLR and no-transform (column three Table V), we can see that only a few experiments produced a significant difference between the results. These are semi-tied and CMLLR "*AST*" experiments, CMLLR "*Lwazi Eng*" experiment and CMLLR "*NCHLT*" experiment. The remaining results (12 of 16) allow us to accept the null hypothesis and conclude, for these experiments, that using semi-tied or CMLLR transforms does not produce a significant increase or decrease in accuracy, as compared to a ASR system that does not implement these transforms.

To investigate further we performed a few experiments with the *Timit* and *NTimit* corpora. The results are presented in Table VI and indicate word accuracies in percent. The standard ASR system was developed (see Section III-E) and a flat recognition grammar was used which only contained words from the testing vocabulary. ATK cepstral normalization was utilized.

The results in Table VI show that semi-tied transforms provide an increase in accuracy for within corpus experiments but

Training Corpus	Testing Corpus	
	Timit	NTimit
Timit with semi-tied	60	10.11
Timit	56.60	19.44
NTimit with semi-tied	16.92	46.50
NTimit	23.38	43.46

TABLE VI
WORD ACCURACIES (%) WHEN BASELINE AND SEMI-TIED TRANSFORMS SYSTEM ON THE *Timit-NTimit* CORPORA.

for cross-corpus experiments applying semi-tied transforms reduced the ASR accuracy. The semi-tied transforms seem to amplify the data mismatch and thus decrease performance. This might explain why semi-tied transforms did not produce an average gain in performance for the AutoSecretary ASR models since there are slight mismatches between the training and testing environments which were amplified by the transform.

VI. CONCLUSION AND FUTURE WORK

The paper presented work aimed at optimising acoustic models for the AutoSecretary call routing system. The optimised acoustic models were developed by:

- creating a modified South African English phone set and an appropriate pronunciation dictionary,
- investigating various cepstral normalization techniques,
- experimenting with three South African corpora and training data combinations, and,
- applying model-space transformations.

The pronunciation dictionary contained a simplified South African English phone set which was used to robustly represent the acoustic sounds found in the South African multilin-

	None & Semi-tied	None & CMLLR	Semit-tied & CMLLR
Lwazi Eng	1.00000	0.00626	0.01383
Lwazi Eng + Langs	0.71830	0.52480	0.28980
AST	1.2e-08	0.00017	0.00347
NCHLT	0.19390	0.02830	0.62410
Lwazi Eng + Langs + AST	0.21100	0.50500	0.04658
Lwazi Eng + Langs + NCHLT	0.60010	0.38650	0.16210
Lwazi Eng + Langs then AST	1.00000	0.82200	0.90350
Lwazi Eng + Langs then NCHLT	0.59440	0.15520	0.51570

TABLE V
P-values CALCULATED USING MCNEMAR'S TEST, FOR VARIOUS TRANSFORMATION COMBINATIONS (NONE, SEMI-TIED AND CMLLR).

gual acoustic space. Each name and language entry contained multiple pronunciation variants to cope with the variability found in proper name pronunciation. For future work an investigation into automatically generating proper name pronunciations and variants should be performed to reduce the amount of manual intervention required during dictionary development. An automatic pronunciation-prediction method will also help to rapidly customize the AutoSecretary application.

The choice of cepstral normalization technique is important since the approach used to normalize the training and testing data does affect the results produced by the ASR system, as was shown by the results captured in Section V-A. The ATK normalization method proved to be the best approach while the generally used utterance-based normalized performed poorly.

Our data combination experiments showed that the best training corpus was a combination of "*Lwazi English, Lwazi language prompts and AST*". Specifically, by developing a tied-state ASR system on the Lwazi English and Lwazi language prompts, then adding the *AST* data for mixture incrementing we managed to achieve a language recognition accuracy of 95.11% and a name recognition accuracy of 93.31% on an independent test corpus. These optimised acoustic models should:

- with high accuracy be able to detect a spoken South African language name which the system can use to route a caller based on language preference, and,
- accurately recognize new names provided that an adequate number of accurate pronunciations and relevant variants are included in the pronunciation dictionary.

Surprisingly, the larger *NCHLT* corpus did not provide substantial gains in accuracy and in some cases no gains were achieved. The most likely explanation is that the data mismatch hindered its effectiveness due to the corpus containing high-bandwidth recordings instead of telephony recordings which make up the AST corpus.

In Section IV-B we postulated that CMLLR could be used as an approximate replacement for semi-tied transforms. Our results in Section V-B showed that overall the accuracies produced by both methods are quite similar and only three experiments showed statistical significant results. Furthermore, when comparing the results between systems that did not implement semi-tied or CMLLR trasforms, to those that did, the vast majority of experiments failed to produce statistcally significant improvements.

Our results indicated that although semi-tied transforms can increase the ASR system performance when the training and testing data are relatively matched, care should be taken when applying the transform when there is a data mismatch as this could degrade the system performance.

ACKNOWLEDGEMENTS

We would like to thank the following people:

- The sprint team: Marelie Davel, Charl van Heerden, Nic de Vries, Thipe Modipa, Willem Basson
- Bryan Mcalister for helping us with the testing data collection
- Herman Kamper and Thomas Niesler for sharing their AST experimental set-up with us.

REFERENCES

[1] P. Modipa, F. de Wet, and M. Davel, "ASR performance analysis of an experimental call routing system," in *20th Annual Symposium of the Pattern Recognition Association of South Africa (PRASA)*, Stellenbosch, South Africa, Nov. 2009, pp. 127–130.

[2] S. Young. (2012) ATK Manual. [Online]. Available: http://mi.eng.cam.ac.uk/research/dialogue/ATK_Manual.pdf

[3] F. Bechet, R. de Mori, and G. Subsol, "Very large vocabulary proper name recognition for directory assistance," in *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU '01).*, Madonna di Campiglio, Italy, Dec. 2001, pp. 222–225.

[4] B. Rveil, J.-P. Martens, and H. van den Heuvel, "Improving proper name recognition by adding automatically learned pronunciation variants to the lexicon," in *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta, may 2010, pp. 2149–2154.

[5] A. F. Llitjos and A. W. Black, "Knowledge of language origin improves pronunciation accuracy of proper names," in *Seventh European Conference on Speech Communication and Technology*, Aalborg, Denmark, Sep. 2001, pp. 1919–1922.

[6] O. Giwa, M. Davel, and E. Barnard, "A Southern African corpus for multilingual name pronunciation," in *22th Annual Symposium of the Pattern Recognition Association of South Africa (PRASA)*, Vanderbijlpark, South Africa, Nov. 2011, pp. 49–53.

[7] M. Marx and C. Schmandt, "Putting people first: Specifying proper names in speech interfaces," in *Proceedings of the 7th annual ACM symposium on User interface software and technology*, Marina del Rey, CA, USA, Nov. 1994, pp. 29–37.

[8] Meraka Institute. (2012) Lwazi ASR corpus. [Online]. Available: http://www.meraka.org.za/lwazi

[9] E. Barnard, M. Davel, and C. van Heerden, "ASR corpus design for resource-scarce languages." in *10th Annual Conference of the International Speech Communication Association (Interspeech 2009)*, Brighton, UK, Sep. 2009, pp. 2847–2850.

[10] N. J. de Vries, J. Badenhorst, M. H. Davel, E. Barnard, and A. de Waal, "Woefzela - an open-source platform for ASR data collection in the developing world," in *12th Annual Conference of the International Speech Communication Association (Interspeech 2011)*, Florence, Italy, Aug. 2011, pp. 3176–3179.

[11] J. C. Roux, P. H. Louw, and T. R. Niesler, "The African Speech Technology project: an assessment," in *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal, May 2004, pp. 93–96.

[12] H. Kamper, F. J. M. Mukanya, and T. R. Niesler, "Multi-accent acoustic modelling of South African English," *Speech Communication*, vol. 54, pp. 801–813, Feb. 2012.

[13] S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, X. Liu, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland. (2012) HTK book. [Online]. Available: http://htk.eng.cam.ac.uk/docs/docs.shtml

[14] M. J. F. Gales, "Semi-tied covariance matrices for Hidden Markov Models," *IEEE Trans. Speech Audio Process.*, vol. 7, pp. 272–281, May 1999.

[15] V. Digalakis, D. Rtischev, L. Neumeyer, and E. Sa, "Speaker adaptation using constrained estimation of Gaussian mixtures," *IEEE Trans. Speech Audio Process.*, vol. 3, pp. 357–366, Sep. 1995.

[16] L. Gillick and S. Cox, "Some statistical issues in the comparison of speech recognition algorithms," in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Glasgow, Scotland, May 1989, pp. 532–535.

[17] F. Yates, "Contingency tables involving small numbers and the χ^2 test," *Supplement to the Journal of the Royal Statistical Society*, vol. 1, no. 2, pp. 217–235, 1934.

Context-dependent modelling of English vowels in Sepedi code-switched speech

Thipe I. Modipa* †, Marelie H. Davel†, Febe de Wet*,
*Multilingual Speech Technologies, North-West University, Vanderbijlpark 1900, South Africa
†Human Language Technologies Research Group, CSIR Meraka Institute, Pretoria, South Africa
Email: {tmodipa,fdwet}@csir.co.za, marelie.davel@gmail.com

***Abstract*—When modelling code-switched speech (utterances that contain a mixture of languages), the embedded language often contains phones not found in the matrix language. These are typically dealt with by either extending the phone set or mapping each phone to a matrix language counterpart. We use acoustic log likelihoods to assist us in identifying the optimal mapping strategy at a context-dependent level (that is, at triphone, rather than monophone level) and obtain new insights in the way English/Sepedi code-switched vowels are produced.**

I. Introduction and Background

Code switching – using words and phrases from more than one language within a single utterance – is a common phenomenon among multilingual speakers. There are a number of reasons why multilingual speakers engage in code switching. In the case of Sepedi, speakers often use a foreign language (English) for numbers, dates and time, a phenomenon that has been observed in other South African languages as well [1].

For automatic speech recognition (ASR) systems, code-switched (CS) speech provides an interesting challenge. This can be dealt with by building fully multilingual systems (combining dictionaries, language and/or acoustic models from multiple languages) or by running more than one monolingual system in parallel, switching from the one to the other [2], [3]. We are interested in the first approach, and specifically where acoustic models are combined at the phone or sub-phone level.

Various techniques have been used when deciding how and when to combine the acoustic model of a phone from the embedded language (English in this case) with a phone from the matrix language (Sepedi in this case). One such technique consists of mapping the embedded phones to the matrix phones prior to system training. This can be achieved in different ways, specifically:

- Using IPA features directly: mapping phones based on existing linguistic knowledge. (IPA features classify sounds based on the phonetic characterisation of those speech sounds [2]).
- Using a confusion matrix from an existing ASR system: calculating the rate of confusion between two phones using a phoneme recogniser in the matrix language and acoustic data from the embedded language [3].
- Using log likelihood differences directly as a distance measure that tests how well two different models fit the same data [4], [2].
- Using acoustic distance measures such as Kullback-Liebler measure, Battacharyya distance metric, Mahalanobis measure or a simple Euclidean measure [5].
- Using a probabilistic phone mapping [6], that is, a model for mapping phones between source sequence X, and target sequence Y, where the model parameters are given by

$$PM(x \mid y) : x \in X, y \in Y \tag{1}$$

and this model is estimated from the results of a phoneme recogniser and the modelled pronunciations. Note that this model (like the current work) is context-sensitive.

In an earlier analysis of English/Sepedi CS speech [7], it was found that applying grapheme-to-phoneme (g2p) rules of the matrix language (Sepedi) to the code-switched words directly, outperformed more sophisticated mapping approaches, and specifically one whereby the g2p rules of the embedded language (English) is used to predict possible pronunciations and these then mapped on a per-phone basis to the closest matching Sepedi phone. This was an unexpected result: it could either mean that the mapping used (obtained from a confusion matrix, as described in [7]) was not optimal, or that Sepedi speakers do interpret some English words according to Sepedi pronunciation rules, for example, pronouncing the word 'chocolate' as / S O k O l a t / rather than as / t S Q k l @ t / (using X-SAMPA notation).

In this work we investigate the process of obtaining a phone mapping from the embedded language to the matrix language. The main goal is to determine whether a better mapping can be obtained, given the specific corpus we are modelling, and to explore tools to analyse this task. We focus on English vowels (English consonant mappings are more predictable), and investigate the use of model likelihoods to guide the mapping choice at a context-dependent level. When unlimited training data is available, using all matrix language and embedded language models combined is expected to perform best; with constrained corpora, extending the phone set indiscriminately is expected to hurt performance due to data scarcity. The optimal mapping is therefore dependent on the specific speech corpus being modelled: our goal is to investigate tools that can guide this mapping process.

In the current work we first verify and extend the earlier English/Sepedi code-switched ASR results (as discussed above) to determine whether these were corpus-specific or whether trends are retained across corpora; we then use log likelihood ratios to analyse the possible context-dependent

phone mappings from the embedded language phones to the matrix language phones.

The paper is organised as follows: In Section II we describe the approach we use to analyse context-dependent mappings. In Section III we describe the speech corpora used in a fair amount of detail, as this provides the context for the various experiments undertaken. Experiments and results are discussed in Section IV. Section V summarises the findings from this analysis and provides some suggestions for future work.

II. Approach

The approach we use to determine an optimal phone mapping is fairly straightforward: we score the English vowels against context-dependent acoustic models of vowels from both the embedded and matrix language and compare the likelihood ratios. These ratios give us an indication of 'model closeness' and suggest mapping candidate(s) at a triphone level. We analyse these mapping candidates to determine whether a triphone should be mapped, and if so, to which matrix language triphone.

The specific process we use to determine mappings is as follows:

1) Context-dependent acoustic models are trained with pure Sepedi data (not containing any code-switched speech).
2) Context-dependent acoustic models are trained from the available Sepedi code-switched data by extending the Sepedi phone set with all English phones.
3) For each English phone, possible mapping candidates are selected using a confusion matrix (as described in more detail later in Section IV-A). Note that these mapping candidates are selected at the monophone level.
4) Analysis is performed at triphone level:
   a) The English data is force aligned using the English triphone model.
   b) The same data is similarly aligned using each of the Sepedi candidate triphone models. These models are constructed from the actual left and right contexts observed, with only the centre phone replaced.
   c) The likelihood ratio between (a) and (b) is evaluated per candidate triphone, per code-switched sample, in practice by calculating the difference in log likelihood, for each English triphone e, matching Sepedi candidate s_e and data sample d_e, referred to from here onwards as $ll\_diff(e, s_e, d_e)$.
   d) The average of the values in (c) is obtained per candidate triphone s_e by averaging over all data samples d_e, giving a single value of $ll\_mean(e, s_e)$ per English and Sepedi candidate triphone pair.
5) The relative scores are used to determine mappings:
   a) If there is a clear Sepedi triphone winner, only that candidate triphone is selected for mapping, that is, if the difference in $ll\_mean(e, s_e)$ between two candidate triphones exceeds a threshold α.
   b) If there is not a clear winner, all triphone candidates that have a value of $ll\_mean(e, s_e)$ less than a second threshold value β are selected as possible mappings (introducing variants for that specific context).
   c) For triphones that have no suitable counterpart (no candidate mappings that obtain a value of $ll\_mean(e, s_e)$ smaller than β), phone set extension is considered.

III. Data

In this section we describe the data used during experiments: the audio corpora, phone sets and dictionaries.

A. Audio corpora

We use two different audio corpora for the experiments: a general Sepedi corpus (NCHLT [8]) and a custom-designed code-switched corpus (SPCS [9]).

The NCHLT corpus was collected using a locally developed smart-phone based speech data collection tool, Woefzela [8]. The corpus consists of prompted speech, mostly in Sepedi but also including some English speech (generated from English text) as produced by Sepedi first language speakers. The corpus consists of 12 560 unique word tokens produced by 113 speakers. We use both the full corpus (referred to as $nchlt\_all$ from here onwards) consisting of all Sepedi and English data and create a subset ($nchtl\_sep$) consisting only of pure Sepedi utterances. This corpus contained no code-switched sentences. Table I shows the distribution of male and female speakers, and the duration of the train and test sets in the different corpora.

TABLE I
Distribution of the number of male and female speakers.

		Speakers	Duration (min)
nchlt_sep	Train	92 (38 female, 54 male)	1 417.62
	Test	20 (10 female, 10 male)	247.28
nchlt_all	Train	82 (33 female, 49 male)	2 782.48
	Test	30 (15 female, 15 male)	1 055.68

The SPCS corpus was collected using prompts that were derived from code-switched transcriptions generated from actual radio broadcasts [9]. It was also collected using Woefzela. Twenty speakers (12 females, 8 males) each read approximately 450 utterances, resulting in 10 hours of prompted speech.

Table II lists the number of unique English and Sepedi words found in the corpus. As discussed in [9], we also list *semi-modified* words (giving a total of 787 unique words): English words that are transformed when embedded in Sepedi speech, for example the word *graduate* that can be pronounced as *graduata* when used within general Sepedi speech.

TABLE II
Number of unique words and total number of utterances in the SPCS corpus.

# Semi-modified	# Eng words	# Sepedi words	# Utterances
58	345	384	12 386

B. Phone sets and dictionaries

The pronunciation rules were obtained from two sources:

1) Standard Sepedi g2p rules (Default&Refine [10] trained on the 5 000-word Lwazi dictionary [11]). In addition, affricates were split according to [12] resulting in 32 Sepedi phones being used in practice.
2) English g2p rules (Default&Refine trained on a South African English (SAE) dictionary created using manually created British-to-SAE phone-to-phone (p2p) mappings [13])

All pronunciations of words occurring in the SPCS corpus were manually verified and corrected, where necessary. The final dictionary contained 29 phones that occur in English but are not found in the Sepedi phone set, as shown in Table III.

TABLE III
Number of phones of different categories found in the various phone sets used.

	Sepedi standard	Sepedi split affricates	English	English phones not occurring in Sepedi
Affricates	9	-	2	1
Fricatives	11	11	10	5
Stops	8	8	6	6
Nasals	5	5	3	-
Vowels	7	7	12	8
Trill	1	1	-	-
Approximants	4	4	4	1
Diphthongs	-	-	8	8
Total	45	36	45	29

IV. Experiments and Results

First, we repeat the experiments as performed in [7] on the NCHLT corpus, for two reasons: to determine whether trends are consistent across corpora, and to obtain a comparable baseline for the phone mapping analysis. Once the baseline has been established we analyse the context-dependent likelihood ratios for the English vowels to obtain a possible mapping.

A. Baseline ASR systems

As a baseline implementation we create four systems on the same training data ($nchlt\_all$) using four standard approaches, the first three of which were used in [7]:

1) Sepedi-only phone set: all words (English and Sepedi) are predicted using Sepedi g2p.
2) Extended phone set: English words are predicted using English g2p, Sepedi words are predicted using Sepedi g2p and all phones retained.
3) Mapped phone set: All English phones (from (2)) are mapped to the single best candidate based on a confusion matrix; no English phones are retained. The confusion matrix was obtained as follows:
   - Freely decoded phone-level labels are obtained from the Sepedi system (using $nchlt\_all$, but only Sepedi phones).
   - The SCPS data is aligned using a dictionary containing the extended phone set (English and Sepedi phones).
   - Iterative dynamic programming (using tools from [14]) is used to obtain an accurate confusion matrix at phone-level.
   - For every English phone, the Sepedi phone with the highest confusability is selected.
4) Code switched variants: Sepedi pronunciations from (1) and English mapped pronunciations from (3) are added as variants both during training and testing.

All four systems are created in a similar way: a fairly standard Hidden Markov Model (HMM) based ASR system is implemented using the HTK toolkit [15]. Acoustic models consist of cross-word tied-state triphones modelled using a 3-state continuous density HMM. Each HMM state distribution is modelled by an 8-mixture multivariate Gaussian with a diagonal covariance matrix. The 39-dimensional feature vector consists of 13 static Mel-Frequency Cepstral Coefficients (MFCCs) with 13 delta and 13 acceleration coefficients appended. The Cepstral Mean and Variance Normalisation (CMVN) preprocessing is used and Semi-tied transforms applied.

These four systems are then tested on three different test sets, obtained from the Sepedi-only NCHLT data ($nchlt\_sep$), all NCHLT data ($nchlt\_all$) and all SPCS data ($spcs$), respectively. Note that the SPCS data is always used as a test set: it is never included in data used either for training or system tuning.

TABLE IV
Phone error rates of different baseline systems on each of three test sets.

Test set	Sepedi-only	Extended phone set	Mapped phone set	CS variants
nchlt_sep	30.72	45.09	33.65	31.92
nchlt_all	33.37	42.54	34.32	35.88
spcs	39.63	56.46	44.16	42.27

The phone error rates (PER) of NCHLT and SPCS test data using different approaches to modelling code switched words are obtained as shown in Table IV. Utterances that cannot be decoded by any of of the systems are removed from the corpus to ensure a fair comparison across systems.

In this careful analysis across different test sets, we see that the previously observed trends remain consistent: Sepedi-only g2p provides the most effective approach to dealing with code-switched speech. Simply extending the phone set results in a large increase in error rate. When the English phones are mapped to their Sepedi counterparts, error rate decreases (compared to the extended phone set); error rate again decreases when two variants (the English remapped version and the Sepedi g2p version) are added per code-switched word. Even though error rates decrease during this process, the best results are still obtained when using a straightforward Sepedi g2p prediction.

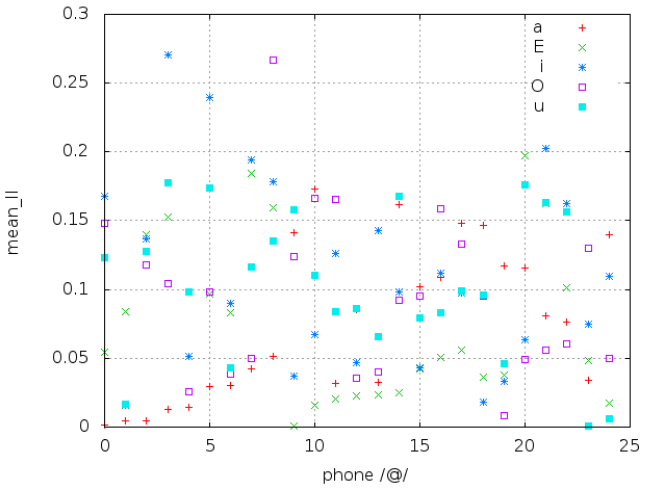


Fig. 2. Mean log likelihood differences (ll_mean) for one phone /@/ in different context and mapping candidates /a/, /E/, /i/, /O/ and /u/.

B. Selecting candidate mappings

We obtain mapping candidates from the same confusion matrix described in section IV-A (previously used to identify a single best match). This time, we flag all phones that are confused with the target phone more than 20% of the target phone occurrences.

Table V lists the frequency of occurrence of the English vowels in the NCHLT training set, and the SPCS corpus, respectively. For each vowel, the mapping candidates are identified and per candidate, the number of times a target phone to mapping candidate pair was observed in the confusion matrix is provided in brackets. We also show the number of unique phone contexts observed in the SPCS corpus.

TABLE V

Phone mapping candidates obtained from confusion matrix. For each English vowel, the number of times it was observed in each corpus is provided. For each phone-candidate pair, the number of times that the confusion was observed in the testing data is provided in brackets.

phone	train counts (nchlt_all)	test counts (spcs)	candidates	unique phone contexts
@	59 652	10 445	a (4448), E (2534) i (1165), O (1156) u(78)	121
i:	21 789	711	i (389), E (205)	15
A:	2 731	749	a (635), E (51)	11
{	2 265	2 479	a (1775), E (536)	39
u:	1 220	1065	u (434), O (216)	23
Q	1 214	1811	O (1208), a (429)	32
O:	1 174	1 333	O (1009), a (283)	19
E:	972	991	E (663), a (196)	18

C. Context-dependent analysis

Once the mapping candidates have been identified, the triphone analysis as described in Section II (4) can be performed. The English models are obtained from the $nchlt\_all$ corpus and the Sepedi models from the $nchlt\_sep$ corpus. The $ll\_mean(e, s_e)$ values are calculated for all the vowels e and mapping candidates s_e as listed in Table V. In this work, we only consider contexts where the left and right contextual phones occur in both the English and Sepedi phone sets. (This means, for example, that we do not include a triphone such as /T-Q+@/ in the current analysis.)

To illustrate the concept, we first plot the results for a single context /S-@+n/ when found in different words. Results are averaged over all speakers. As can be seen in Fig. 1, the best matching context (/S-E+n/) is always the closest match, irrespective of the word in which it is used. The runner up is /S-i+n/: this context always provides a poorer match than /S-E+n/, with results most comparable in the word 'national', which interestingly, does have a different morphological construct than the others. The results displayed in Fig. 1 is better contextualised by considering the mean log likelihood difference between standard Sepedi /S-E+n/ contexts and the Sepedi /S-E+n/ model, which is 0.004 (indicated in Fig. 1) by a horizontal line.

In Fig. 2 we provide the same results, but now averaged over all words that contain a specific context. We plot the results for one phone /@/ when found in different contexts. Again, results are averaged over all speakers. From Fig. 2 it is clear that /E/ provides the best match in general, but that there are some contexts where other phones are better mapping candidates. The phones /a/, /O/ and /i/ also provide best matches in a limited number of contexts, whereas the phone /u/ only provides a best match in two instances.

This process was repeated for all the vowels. Two more examples are shown in Figures 3 and 4, illustrating the mean log likelihood differences for vowels /Q/ and /{/, respectively.

When this process is repeated for additional contexts, we are able to identify additional context-dependent Sepedi candidates that provide the best match to each of the context-dependent English vowels.

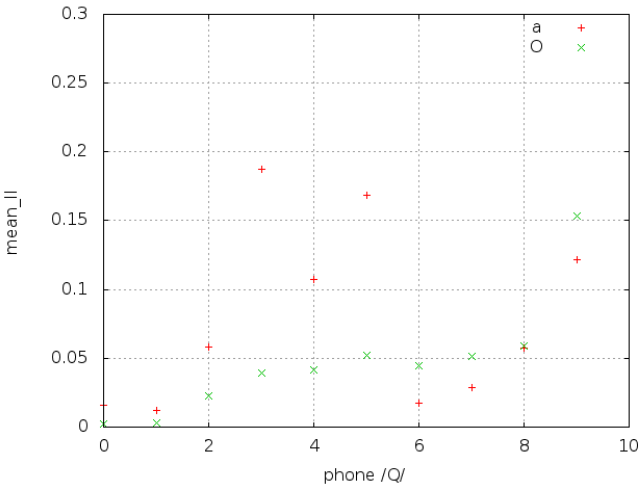


Fig. 3. Mean log likelihood differences (ll_mean) for phone /Q/ in different context and mapping candidates /a/ and /O/.

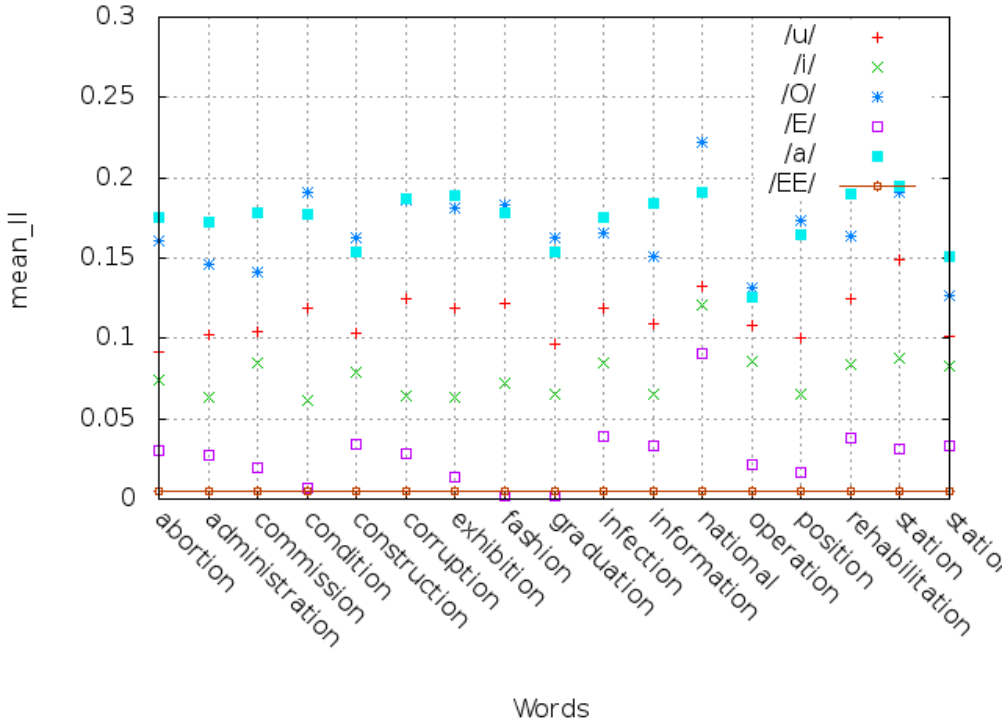


Fig. 1. Mean log likelihood differences (ll_mean) for one context /S-@+n/. Each mapping candidate is displayed using a different colour. /EE/ is displayed as calibration: the ll_mean of standard Sepedi /E/ data measured against the standard Sepedi /E/ model.

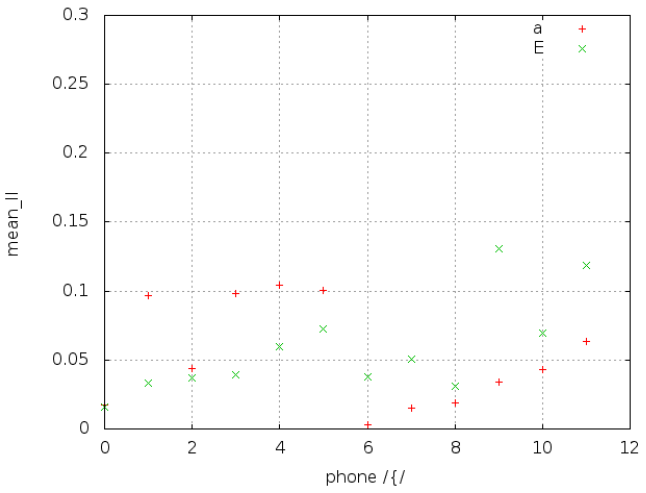


Fig. 4. Mean log likelihood differences (ll_mean) for phone /{/ in different context and mapping candidates /a/ and /E/.

D. Obtaining a mapping from likelihood results

The above likelihood results are used to determine possible actions to take with regard to the English vowels. As mentioned in Section II, the possible options per phone context are to:

1) Extend the matrix language phone set by adding the embedded language phone (if no candidate with an ll_mean value of less than β);
2) Map the embedded language phone to the single closest matrix language phone; or
3) Map the embedded phone to more than one candidate matrix phone (if candidates closer than α).

Both α and β can be tuned on a development set. The context-dependent mapping is obtained by finding the most appropriate candidate triphones using these thresholds. For every winning candidate triphone (see 2), we determine which other candidate triphone is within the defined threshold.

In order to illustrate the concept, we use the analysis in IV-C to select thresholds such that α is 0.02 and β is 0.1 (implying that the phone set is not extended).This results in the mappings determined for /@/, as shown in Table VI.

V. Conclusion

In this investigation, we have shown that acoustic log likelihoods provide a useful tool when analysing the optimal mapping of embedded language phones to matrix language phones, and that context is important when applying such mappings. We also introduced a new corpus of Sepedi/English codes-switched speech, and confirmed that (for this corpus, as found earlier in [7]), Sepedi g2p predictions of the pronunciations of English words provide a viable alternative to more sophisticated modelling approaches, and that, in fact, it is difficult to obtain a better alternative with context-insensitive mappings.

TABLE VI
The context-dependent mapping for phone /@/.

Phone	Mapping
n-@+S	a
s-@+m	a,i,O,u
m-@+f	a
S-@+l	a
m-@+sil	a,O
n-@+l	a
d_0Z-@+l	a,O,u
d_0Z-@+h_b	a,O
s-@+d_0Z	a
f-@+f	E
S-@+n	E
s-@+l	a,E
n-@+m	E,O
s-@+n	a,E,O
h_b-@+l	E
n-@+s	E,i
d_0Z-@+n	E
m-@+n	E
s-@+s	E,i
l-@+s	O
s-@+w	i,O
i-@+w	O
l-@+n	a,O
i-@+f	u
l-@+d_0Z	E,u

The next step in our research will be to determine the impact of the identified mappings on ASR system performance. This will also require a thorough investigation of the thresholds α and β, balancing the need for accurate mappings with the additional confusability introduced by extra pronunciation variants.

Future work will include extending the phone mapping analysis to contexts where the left and right phones themselves are only in one of the two phone sets. This will also allow us to extend the analysis to the full phone set by iteratively mapping phones, in the process increasing the matched phone sets. In addition, we would like to analyse whether some of the observed mappings are speaker-specific, or robust across speakers (the current assumption); and whether the graphemic context of the triphone also plays a role in producing an optimal mapping.

While the above would provide a practical (and more nuanced) tool when producing phone mappings for code-switched speech, the current analysis already provides some interesting insights with regard to the acoustic properties of English/Sepedi code-switched speech.

REFERENCES

[1] T. Niesler and D. Willett, "Language identification and multilingual speech recognition using discriminatively trained acoustic models," in *Multilingual Speech and Language Processing*, 2006.

[2] D. Yu, L. Deng, P. Liu, J. Wu, Y. Gong, and A. Acero, "Cross-lingual speech recognition under runtime resource constraints," 2009.

[3] V. B. Le, L. Besacier, and T. Schultz, "Acoustic-phonetic unit similarities for context dependent acoustic model portability," in *Proc. ICASSP*, 2006.

[4] J. Kohler, "Multi-lingual phoneme recognition exploiting acoustic-phonetic similarities of sounds," in *Proc. International conference on spoken language processing (ICSLP 96)*, 1996, pp. 2195–2198.

[5] J. J. Sooful and E. C. Botha, "Comparison of acoustic distance measures for automatic cross-language phoneme mapping," in *Proc. ICSLP*, 2002.

[6] K. C. Sim and H. Li, "Context-sensitive probabilistic phone mapping model for cross-lingual speech recognition," in *Proc. Interspeech*, 2008, pp. 2715–2718.

[7] T. Modipa and M. H. Davel, "Pronunciation modelling of foreign words for Sepedi ASR," in *Proc. 21st Annual Symposium of the Pattern Recognition Association of South Africa (PRASA)*, 2010, pp. 185–189.

[8] N. J. de Vries, J. Badenhorst, M. H. Davel, E. Barnard, and A. de Waal, "Woefzela an open-source platform for ASR data collection in the developing world," in *Proceedings of Interspeech*, 2011, pp. 3177–3180.

[9] T. Modipa, F. de Wet, and M. H. Davel, "An acoustic corpus of english/sepedi code-switched speech," *South African Journal of African Languages*, in preparation.

[10] M. Davel and E. Barnard, "Pronunciation prediction with Default&Refine," *Computer Speech and Language*, vol. 22, pp. 374–393, 2008.

[11] M. H. Davel and O. Martirosian, "Pronunciation dictionary development in resource-scarce environments," in *Proc. Interspeech*, 2009, pp. 2851–2854.

[12] T. Modipa, M. H. Davel, and F. de Wet, "Acoustic modelling of Sepedi affricates for ASR," in *Proc. Annual Research Conference of the South African Institute of Computer Scientist and Information Technologists (SAICSIT 2010)*, 2010, pp. 394–398.

[13] L. Loots, M. H. Davel, E. Barnard, and T. Niesler, "Comparing manually-developed and data-driven rules for p2p learning," in *Proc. 20th Annual Symposium of the Pattern Recognition Association of South Africa (PRASA)*, 2009, pp. 35–40.

[14] M. H. Davel, C. J. van Heerden, and E. Barnard, "Validating smartphone-collected speech corpora," in *Proc. SLTU*, 2012, pp. 68–75.

[15] HTK, "The Hidden Markov Model Toolkit (HTK)," 2009.

On the leakage problem with the Discrete Pulse Transform decomposition

Inger Fabris-Rotelli and Gene Stoltz
Department of Statistics, University of Pretoria, 0002, Pretoria, South Africa
E-Mail: inger.fabris-rotelli@up.ac.za

***Abstract*—Connected operators act directly on connected components in an image, and though they present a strong framework for extraction of meaningful structures in an image, always suffer from the issue of leakage. The concept of leakage within any connectivity framework refers to situations in which two connected components are connected to each other via a thin, possibly long, pixel-sized connected component and are subsequently considered a single connected component. The LULU operators L_n and U_n used to derive the Discrete Pulse Transform are also connected operators and suffer from leakage. We present the Pulse Reformation algorithm to combat leakage in the pulses extracted by the DPT, making use of erosion and subsequent restricted dilations. This enables extraction of meaningful objects consisting of partial pulses of the DPT related over various scales. The examples presented illustrate a useful technique.**

I. Introduction

The concept of an axiomatic connectivity was introduced by Serra [1] and Matheron [2], for use in Mathematical Morphology. The need arose due to elements of the discrete grid on $\mathbb{Z}^2$ not satisfying a total order such as that achieved by a sequence on $\mathbb{Z}$. On $\mathbb{Z}$ one can see that we have an obvious ordering of the elements, namely x_{i+1} follows x_i and x_{i-1} precedes x_i. It is then natural to consider the elements x_{i+1} and x_{i-1} as the neighbours of x_i. However, consider the case of images defined on a discrete grid in $\mathbb{Z}^2$. Although it is natural to consider the 8 surrounding pixels for a pixel x as the neighbours, there is no immediate ordering of the neighbours as is the case in one dimension. This is because $\mathbb{Z}^2$ is only partially ordered. We can apply a raster scan to the grid, that is, starting with the first row move left to right from pixel to pixel and then repeat at next row and subsequent rows. This would however mean we have reduced the grid in $\mathbb{Z}^2$ to a sequence on $\mathbb{Z}$ and we won't have achieved a logical extension from one to two and higher dimensions. Serra and Matheron recognised this need for the concept of an axiomatic connectivity defined in Definition 1.

Definition 1: $\mathcal{C}$ is a **connectivity class** or a **connection** on $\mathcal{P}(E)$ if the following axioms hold:

(i) $\emptyset \in \mathcal{C}$
(ii) $\{x\} \in \mathcal{C}$ for each $x \in E$
(iii) For each family $\{C_i\}$ in $\mathcal{C}$ such that $\bigcap C_i \neq \emptyset$, we have $\bigcup C_i \in \mathcal{C}$.

A set $C \in \mathcal{C}$ is called **connected**.

The well-known 4- and 8-connectivity are examples of a connectivity forming a connectivity class. The concept presented in Definition 1 has been used extensively in mathematical morphology with regard to image processing applications. However, the problem of leakage has been discussed extensively in the same setting.

The concept of leakage within any connectivity framework refers to situations in which two connected components are connected to each other via a thin, possibly long, pixel-sized connected component and subsequently considered a single connected component. See Figure 1 for an illustration of this. Most commonly, this occurs due to noise or the intensity difference between objects and backgrounds. More realistically, such a connected component should be separated into the two larger connected components as these most likely represent two separate objects of the scene and have only been joined together due to noise, low resolution and such quantization effects. It results in, for example, oversegmentation or fragmentation [3], [4].

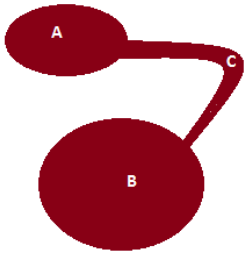


Fig. 1: An illustration of leakage in a connected component. Objects A and B are likely separate objects in reality but are connected by a thin connected component C resulting in a single connected component

Various methods have been employed to counter the problem of leakage. O'Callaghan and Bull [5] explain that leakage occurs in image segmentation due to the existence weak points in the gradient of object boundaries so that the object 'leaks' into the background. They present an improved watershed algorithm for segmentation using decimated wavelets to take care of such cases. Li and Wilson [6] make use of a multiresolution technique via the Fourier transform and a Markov random field to deal with leakage. Leakage also occurs frequently in active-contour techniques. Law and Chung [7] adapt the active-contour algorithm using a minimal weighted local variance condition to estimate where edges should be instead of allowing leakage. Lu and Bao [8] also adapt the active-

contour algorithm by requiring contours to be significantly concave or convex. Graham et al [9] also encounter leakage when developing an algorithm for human-airway segmentation, and adapt the parameters of the segmentation to be more conservative when leakage is observed. Terol-Villalobos et al [10] introduce a stopping criterion to combat leakage and obtain a result between a morphological opening and an opening by reconstruction. Wilkinson [11] defines a second generation connectivity to combat the leakage problem which occurs for all connected filters, that is, filters that operate on the connected components defined by the connectivity involved. Salembier and Oliveras [12] relax the definition of a connection (Definition 1) to define pseudo-connectivity and enable a solution to the leakage problem. Tzafestas and Maragos [13] work with multiscale connectivity obtained via their generalized connectivity measure which essentially measures the degree to which a connected component exhibiting leakage should be connected. Santillán and Herrera-Navarro [14] introduce connected viscous filters to combat leakage. Ouzounis [15] incorporates shape orientation to deal with leakage.

In this article we propose a new algorithm to combat leakage which makes use of the structure of objects in an image obtained via the Discrete Pulse Transform (DPT) [16]. The Discrete Pulse Transform based on the LULU operators for sequences was derived in [17]. Using the extension of the LULU operators L_n and U_n to functions on $\mathbb{Z}^d$ presented in [16] we present the DPT for functions in $\mathcal{A}(\mathbb{Z}^2)$. Similar to the case of sequences we obtain a decomposition of a function $f \in \mathcal{A}(\mathbb{Z}^2)$, with finite support. As usual $\mathrm{supp}(f) = \{p \in \mathbb{Z}^2 : f(p) \neq 0\}$. Let $N = \mathrm{card}(\mathrm{supp}(f))$. We derive the DPT of $f \in \mathcal{A}(\mathbb{Z}^2)$ by applying iteratively the operators L_n, U_n with n increasing from 1 to N as follows

$$DPT(f) = (D_1(f), D_2(f), ..., D_N(f)), \tag{1}$$

where the components of 1 are obtained through

$$D_1(f) = (id - P_1)(f) \tag{2}$$
$$D_n(f) = (id - P_n) \circ Q_{n-1}(f),\ n = 2, ..., N, \tag{3}$$

and $P_n = L_n \circ U_n$ or $P_n = U_n \circ L_n$ and $Q_n = P_n \circ ... \circ P_1$, $n \in \mathbb{N}$. This decomposition has the property that each component D_n in (1) is a sum of discrete pulses with pairwise disjoint supports of size n, where in this setting a discrete pulse is defined as follows.

Definition 2: A function $\phi \in \mathcal{A}(\mathbb{Z}^2)$ is called a pulse if there exists a connected set V and a nonzero real number α such that

$$\phi(x) = \begin{cases} \alpha & \text{if} \quad x \in V \\ 0 & \text{if} \quad x \in \mathbb{Z}^2 \setminus V. \end{cases}$$

The set V is the support of the pulse ϕ, that is $\mathrm{supp}(\phi) = V$. The concept of a pulse as defined in Definition 2 is similar to the idea of a flat zone from mathematical morphology. It should be remarked that the support of a pulse may generally have any shape, the only restriction being that it is connected. It follows from (2)-(3) that

$$f = \sum_{n=1}^{N} D_n(f) = \sum_{n=1}^{N} \sum_{s=1}^{\gamma(n)} \psi_{ns}, \tag{4}$$

where $\psi_{ns}, n = 1, 2, ..., \gamma(n)$ are the pulses extracted by the DPT at scale n and $\gamma(n)$ is the number of pulses of size n extracted at scale n.

The representation (4) provides a multiscale decomposition of the image f. This extracts information from the image at all possible scales and provides connected components (the pulses) which are related through through scale. We thus have multiscale objects at hand for more robust image analysis.

In Section II we present the implementation of the DPT and the proposed Pulse Reformation algorithm to deal with leakage. In Section III we provide illustrations of the technique with comparisons.

II. Algorithm

An algorithm within the DPT scale-space was developed to introduce pulse reliability and pulse 'meaning'. The algorithm defines objects within the DPT scale-space by clustering and reforming various sets of pulses. We first discuss the DPT implementation.

A. DPT Implementation

The algorithm is based on the technique developed by Laurie utilizing graph-theory [18]. The algorithm developed here makes use of two separate graphs, the Work-Graph and the Pulse-Graph. The Work-Graph is an undirected graph denoted $G_{work} = (V_{work}, E)$ representing the finite data sequence $\mathbf{x} = \{x_0, x_1, x_2, ...x_N\}$. This data sequence presents the pixel intensities in a one-dimensional array, namely $x_i = f(m_i, n_i)$ where m is the column position, n the row position and f the discrete pixel intensity function. The Pulse-Graph is a directed graph representing the extracted pulses ϕ_{ns}, where n is the scale and s the pulse number, and is denoted by $G_{Pulse} = (V_{Pulse}, A)$.

The Work-Graph is used directly in executing the DPT. The edges E represent the connectivity used in the execution of the DPT, for example 4- or 8-connectivity. The Pulse-Graph is the output of the DPT with directed edges known as arcs. The arcs show the relationship between pulses at different scales. From the data a Work-Graph is first created and then transformed into a Pulse-Graph by executing the DPT. A visual representation of the algorithm is provided in Figure 2 with a simplistic example.

In the example, the algorithm starts by using the input signal to create the Work-Graph and the basis of the Pulse-Graph. The Work-Graph is created by using each data point in the input signal as a node and two zero nodes of infinite size are added at the beginning and end of the input signal. The edges of the Work-Graph for each node are created by utilizing the required connectivity scheme to connecting the appropriate

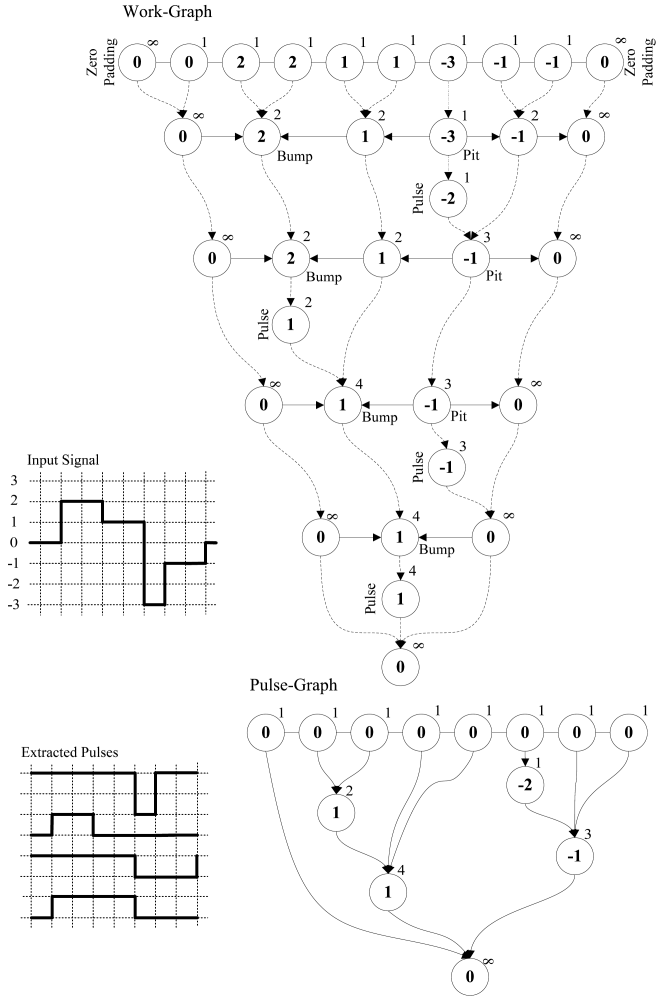


Fig. 2: Illustration of the DPT decomposition

nodes. Here a 1-dimensional wave signal is used with a 2-connectivity scheme, this scheme entails connecting each data point with its two closest neighbours.

To successfully extract the pulses their height and position must be stored and to reduce the memory requirement for the storage of the pulses, they are stored in a graph like format where the basis of this graph stores the positional information. Each node consists of arcs and a strength value. The arcs propagate the different positional information through the graph. For example, a pulse connected to the basis of the Pulse-Graph can be translated as all the base nodes connected to the pulse have a value equal to the strength of the pulse, remembering that the base nodes represents the pixel positions in an image.

The Pulse-Graph is constructed by firstly creating the basis, that is, each pixel in the image creates one node in the basis of the Pulse-Graph with a strength of zero and a size of 1. The size is one because each node only contains 1 element in the data-sequence where the strength must be zero as the sum of all nodes must be equal to the original image.

The Work-Graph is optimized by joining all connected nodes with the same value into one node retaining all relevant edges thus having one node per flat zone, and is then searched for features of every size creating a Feature-Table. A feature is defined as a local maximum(bump) or local minimum(pit) node [16]. The local neighborhood of a node is only one edge deep. The Feature-Table contains all possible features in the Work-Graph.

The decomposition is executed by searching for all size n features in the Feature-Table, $n = 1, 2, ..., N$, depending on the decomposition type. For U_nL_n or L_nU_n the pits or bumps will first be extracted respectively. Each identified feature must be reaffirmed. A feature is reaffirmed by re-checking the node, making sure it still a pit or a bump. It is possible that a feature in the Feature-Table can become a non-feature when another feature in the Feature-Table is extracted. The identified feature is extracted and a new node is created in the Pulse-Graph with the arcs connecting to the pulses that constructed the extracted pulse. By extracting a feature the relating node in the Work-Graph is merged with the node nearest in height, this node is then reaffirmed as a feature. If it is a feature the current entry in the Feature-Table is updated by changing the scale of the feature, otherwise it is removed from the Feature-table. This process is repeated by increasing the scale N each time until no more features are left in the Feature-Table, which is equivalent the final single pulse obtained by the DPT. The algorithm is implemented in the c programming environment and runs in $O(n)$ complexity.

B. Pulse Reformation

1) The Problem: Consider the four separate images in Figure 3 and regard each structure in each image as an object. One image can be created by including all four objects in it as shown in Figure 4a. The challenge is to separate these four objects from the image.

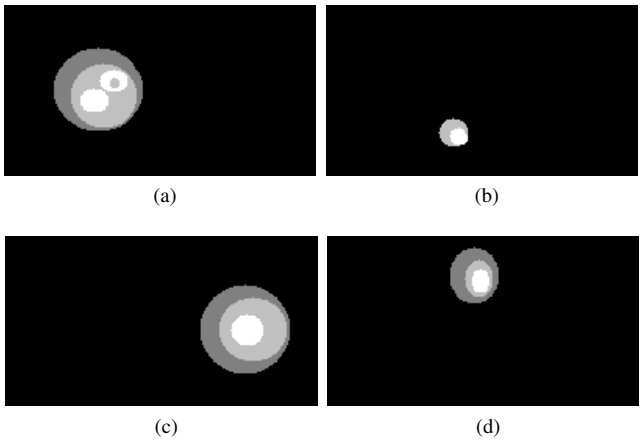


Fig. 3: The four objects which will be combined into one image and then extracted as four separate objects from the original in Figure 4a with the proposed algorithm.

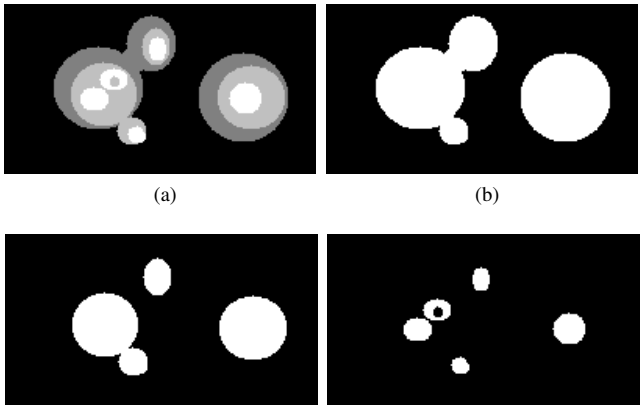


Fig. 4: Attempted extraction of the four objects given in Figure 3 from Figure 4a by using different threshold values. Thresholded images shown in (b), (c) and (d).

To extract the four objects a very simplistic method such as thresholding can be applied. This is achieved by choosing a range of intensity values where all pixel intensities outside the range becomes 0 and inside becomes 1. The number of detected objects is then directly related to the number of connected sets in the image. It can be observed in Figures 4b, 4c and 4d that all possible threshold values have been applied and none have resulted in the correct extracted connected sets.

Applying the DPT to the image a range pulse sizes chosen also provides a type of thresholding. Four ranges has been chosen and can be seen in Figure 5. Here it is also evident that the extracted connected components do not correctly present the true objects in the image.

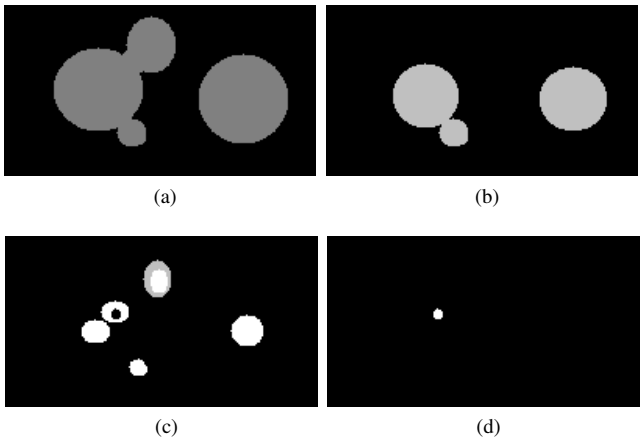


Fig. 5: Attempted extraction of the four objects in Figure 3 from Figure 4a by using different DPT pulse size ranges

By the above examples it is seen that neither thresholding in the intensity domain or in the DPT scale-space has the desired effect of successfully extracting the objects. We now present an algorithm which improves the reliability of the pulses in the DPT by removing the effect of leakage so that the pulses more accurately represent objects in the image.

2) The Solution: Within the DPT scale-space it is easy to see that an assumption of each pulse of an object containing the medial axis [19] of the pulse smaller but closest in scale of the same object is justified due to the theoretical results in [16]. The medial axis is equivalent to the morphological skeleton. This assumption is made to provide a framework for excluding texture from a shape. Texture is mostly the collection of small pulses on a much larger pulse so that it is evident that only the texture on the medial axis will be preserved and all other small pulses will be regarded as noise. We would like to separate objects that are incorrectly joined by leakage. This is achieved by dividing the medial axis of the joined objects into separate medial axis each containing one object. It is assumed that most medial axis has only one center point. To approximate the center point of a medial axis from an object, the object can be eroded consecutively until one element remains. As objects differ in size a set of joined objects can not be eroded until only one element remain as the other objects will then be lost. A set of joined objects are thus eroded until a maximum number of connected sets have been created. These connected sets, called eroded sets of the pulse, are assumed to be the approximations to the medial axis of each object in the set of joined objects. Algorithm 1 (provided in the appendix) shows the process of eroding a pulse such that the number of connected sets remaining is a maximum. It creates a binary image from the pulse. This image is then eroded with the smallest compact structuring element, a 3×3 sized element. After each erosion the number of connected components is checked and the maximum number of connected components is saved for later usage.

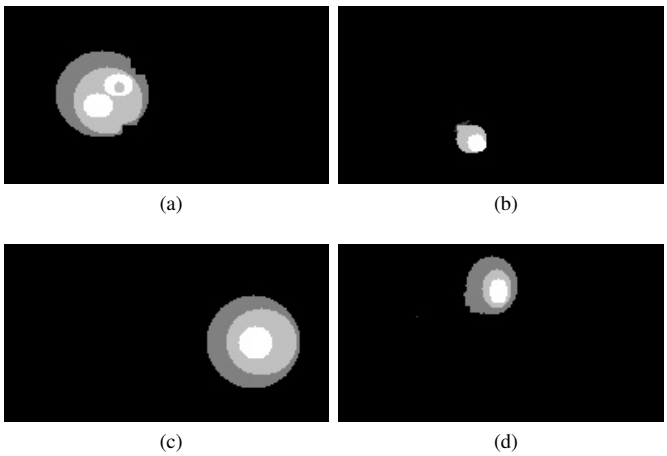
Although all the medial axis centers have been approximately located the remaining elements in the pulse must be assigned to one of these sets, each containing one medial axis. A method to reconstruct a binary image from a medial axis is to utilize morphological openings [20], which is an erosion followed by a dilation. An equivalent approach is followed here where each set is dilated within the pulse boundaries until all elements in the pulse has been assigned to a region. Algorithm 2 (provided in the appendix) shows how each eroded set in each pulse gets dilated on a ratiometric merit until all elements in the pulse have been assigned to an eroded set (region).

Another problem arises when two sets are dilated and the resulting dilations have a non-zero intersection. An element cannot belong to more than one set. To prevent this a ratiometric merit system is implemented. The eroded set with the highest ratio gets dilated first. This ratio is calculated by dividing the cardinality of the dilated eroded set with the cardinality of the set before it was dilated. Using the ratio is very important as this gives an approximation of how centered the medial axis is. However, it also hinders the dilation through leakage areas in the pulse resulting in the desired effect of eliminating leakage. Algorithm 3 (provided in the appendix) creates the regions in each pulse by utilizing the relative eroded

sets of the pulse. The cardinality of each eroded set is recorded before they are dilated. The dilation of a eroded set happens within the boundaries of the pulse, excluding the elements which is already assigned to other eroded sets. The ratio of the cardinality before and after the dilation is calculated. The elements of the eroded set with the highest ratio gets assigned too that specific eroded set. This process is repeated until all elements in the pulse have been assigned.

At this point all the pulses are divided into regions and using the Pulse-Graph, all the regions sharing a pulse also shares the same arcs. Each region must have its own unique set of arcs as it is assumed that each region is a unique object within the pulse. By taking one pulse and starting at a region in the pulse, all regions in pulses connected to the current pulse through arcs must be evaluated. To determine whether two regions are connected the related eroded set of the larger pulse must intersect with the smaller pulse. Algorithm 4 (provided in the appendix) traverses through each region created previously and determine the current region's connected regions. Each region has a relative pulse and each pulse has other pulses connected to it, which in turn have their own regions. The regions in the connected pulses are possible connected regions of the current region. The algorithm traverses through all these possible regions to determine whether they contain the approximate medial axis of the current region. If the region being tested contains the approximate medial axis it become a connected region of the current region.

All the newly created regions can now be seen as new pulses in the Pulse-Graph. This algorithm changes the structure of the original DPT, however more meaningful objects can be extracted from the image setting the scene for object detection and tracking.



(a) (b)

(c) (d)

Fig. 6: The four objects of Figure 3 extracted from Figure 4a by the using Pulse Reformation algorithm

3) Cracking the problem: The algorithm presented in Section II-B2 was applied to the discussed problem in Section II-B1 with the results shown in Figure 6. It can be seen that there is some leakage onto other objects where the object is reconstructed from the approximated medial axis but all four objects was successfully extracted without applying any type of threshold or additional processing.

III. Examples

A. Text Removal

A typical problem in image processing is the successful removal of letters imposed on an image without influencing the structures not directly related to the letters. An image with some lettering is shown in Figure 7. In the image it is clear that there will be a leakage problem where the 'T' touches the horizon.

Fig. 7: An image with imposed text

To remove text with the DPT and Pulse Reformation two automatic assumptions are made. Each letter consists of a flat-zone and an approximate range for the number of elements in each flat zones is known. The Pulse Reformation creates objects with specific properties. For text one can expect to see an object with a large flat zone on the top followed with a few pulses of approximately the same size which is then supported by the background pulses. The result is shown in Figure 8.

Fig. 8: Using the Pulse Reformation algorithm the lettering can be removed from Figure 7

It is clear from Figure 8 that the text was successfully removed and that the 'T' has been successfully separated from the mountain. Visually the image did not loose any other detail. It is possible that other flat zones approximately the same size as the text were removed but this is unobservable. There is however a small change where the 'T' was. It is observable that a small part of the mountain has also been removed. This is due to the creation of the regions and the ratiometric merit. The medial axis of the two different object is only approximated and then grown from there thus that part has grown in favour of the 'T' and not the mountain. Evaluating the image, a large flat zone can be observed where the text was removed. This flat zone assumes an arbitrary intensity value approximately equivalent to the mean value of the text background. This also means that in a highly texture

environment the text will still be evident as it is not replaced with texture but only an approximate mean value of the neighbouring texture area. Further algorithmic developments could improve this technique. However, this example clearly shows the power of Pulse Reformation and a strong solution to the leakage problem.

B. Object Extraction

The algorithm main aim is to extract meaningful objects from an image via the DPT. To test this, the DPT and Pulse Reformation algorithm was applied to an image of blood cells shown in Figure 9a. The 8 strongest objects were extracted and are shown in Figure 9. Here, the object strength refers to the number of pulses it contains since the most salient structures in an image are those that are present over a wide range of scales [21].

Choosing an appropriate range for the objects required, such as the expected size of the cells, the objects in Figures 9b to 9g are easily extracted with the algorithm. Examining the extracted objects it is evident that the objects are not a true presentation of the original observed objects. The extracted objects are circular without the inside hole, with jagged edges and small missing groups of pixels. The inside hole is excluded from the object as it is treated as texture, the jagged edges are formed where a pulse is divided into multiple regions, and the pixels get lost when the approximated medial axis is dilated and no dilation can fill the pixel. To involve texture on objects, one can analyze the number and size of objects formed on top of the current object. If there are many objects of approximately the same size one can assume that it is texture and include it in the final object if necessary. Another addition is to calculated the center of mass of objects created on top of one another and if the centers are close together the separate objects can be combined to form one. One can also observe that the extracted objects have drastic variable intensity differences where it should be approximately equal. This is evidence of the formation of a region and occurs near other possible objects.

The two larger objects in Figures 9h and 9i are a good example where pulse leakage occurred and it was assumed to form one large object. By example Figures 9b and 9c forms part of the object shown in Figure 9h and can be seen as texture on the object. To distinguish between such cases an approximated size for an object is required.

The Pulse Reformation can be compared to similar techniques such as the extraction of λ-connected components [14] which must be used in conjunction with a thresholding technique In this case Otsu's method [22] will be utilized. The λ-connected components are those in which the center of a disk structuring element of radius λ can be moved along a continuous path throughout the connected component such that the entire disk stays within the domain of the connected component. A few examples are shown in Figure 10.

It can be observed in the samples that the λ-connected components do not successfully extract the correct objects. In Figure 10c a manually tuned threshold which provided the best results was used. With this image the correct number of objects can be extracted by using a known cardinality range of the connected components. Figure 10 demonstrates that it

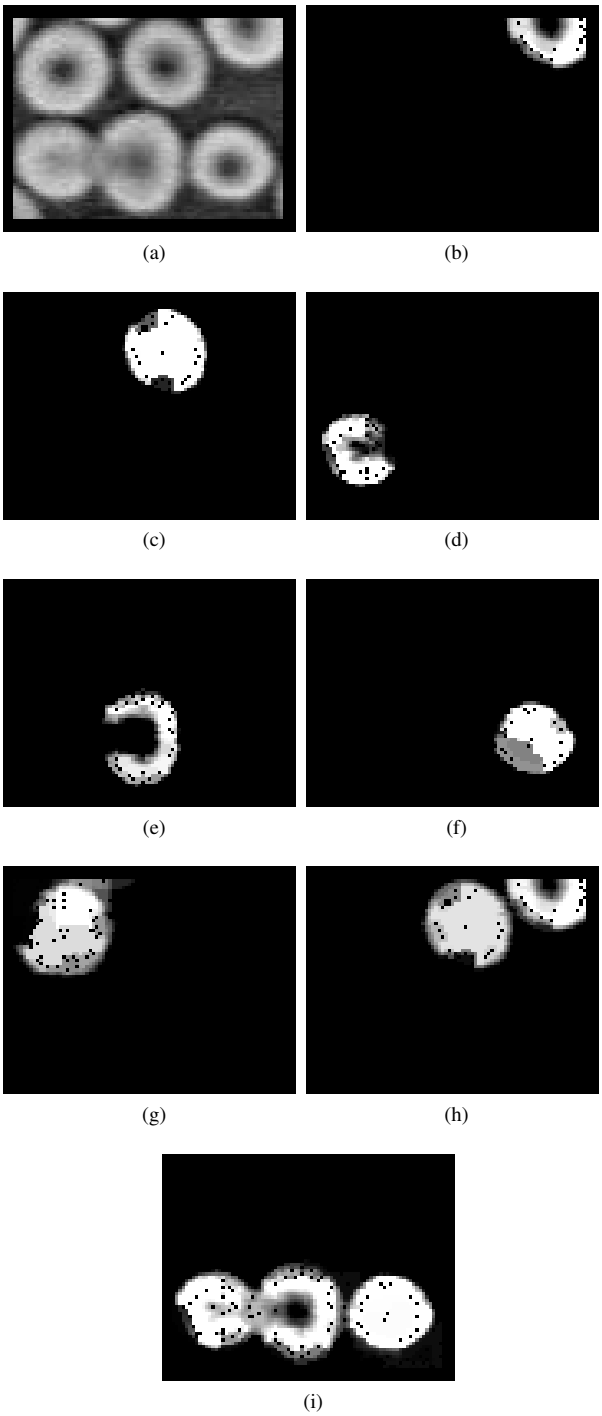
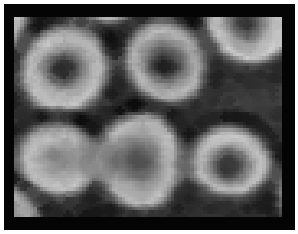


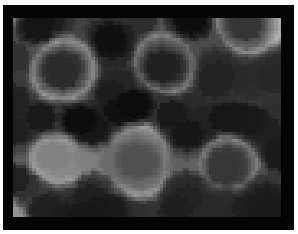
Fig. 9: The Pulse Reformation algorithm is used to extract possible objects (blood cells) within Figure 9a. The 6 blood cells are successfully extracted in addition to two other possible larger objects.



(a) $\lambda = 1$

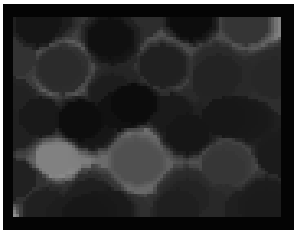
(b) $\lambda = 1$ Thresholded

(c) $\lambda = 1$ Manual Threshold



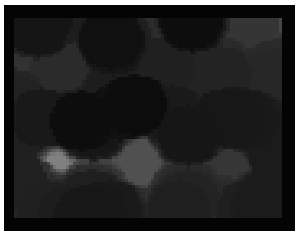
(d) $\lambda = 4$

(e) $\lambda = 4$ Thresholded

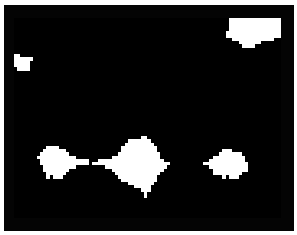


(f) $\lambda = 6$

(g) $\lambda = 6$ Thresholded



(h) $\lambda = 9$



(i) $\lambda = 9$ Thresholded

Fig. 10: λ-connected components are created and thresholded in an attempt to extract the connected sets which presents the objects(blood cells) in Figure 9a. Only when using a manual threshold was this achieved.

is clear that the Pulse Reformation is successful at extracting objects without the use of any thresholding regarding pixel intensity. The fact that thresholding is not required gives the algorithm a large advantage, the only information still required is an approximate range of the size of the objects in question.

IV. Conclusion

Connected operators [23] act directly on connected components, and though they present a strong framework for extraction of meaningful structures in an image, always suffer from the issue of leakage defined in Section I. The LULU operators L_n and U_n used to derive the Discrete Pulse Transform are also connected operators and suffer from leakage. We have presented the Pulse Reformation algorithm to combat leakage in the pulses extracted by the DPT. This enables extraction of meaningful objects consisting of pulses of the DPT related over various scales. The examples presented illustrate a useful technique which will be theoretically investigated as well as refined in future research.

V. Appendix

Algorithm 1 Approximate the medial centers

```
for (each pulse[i]) {
  BinaryI = Create binary image of pulse;
  do {
    BinaryI = eroded BinaryI;
    eroded_k = amount of connected sets in
        BinaryI;
  } while (eroded_k is not maximum);
  for (each eroded_k) {
    Eroded set[i][k] = Connected set[k] in
        BinaryI;
  }
}
```

Algorithm 2 Create regions in a pulse

```
for (each pulse[i]) {
  for (each Eroded Set[i][k]) {
    RegionI[k] = Eroded Set[i][k];
  }
  TotalRegions = Union of all RegionI[k];
  TotalRegionSize = Cardinality of
      TotalRegions;
  PulseSize = Cardinality of pulse[i];
  while (TotalRegionSize != PulseSize)
    Dilate the RegionI[k] with largest
                  Ratio[k];
    TotalRegions = Union of all RegionI[k];
    TotalRegionSize = Cardinality of
        TotalRegions;
  }
}
```

Algorithm 3 Dilate RegionI[k] with largest Ratio

```
for (each Eroded Set[i][k]) {
  MaskI = Union of all RegionI excluding
      RegionI[k];
  MaskIPulse = Pulse[i] excluding MaskI;
  DilatedI[k] = Dilation of RegionI[k]
      intersecting with MaskIPulse;
  Ratio[k] = Cardinality(DilatedI[k]) /
      Cardinality(RegionI[k]);
}
Max_k = k value related to maximum value of
    Ratio[k];
RegionI[Max_k] = Union of RegionI[Max_k] and
    DilatedI[Max_k];
```

Algorithm 4 Connecting regions through arcs.

```
for (each pulse[i]){
  for (each RegionI[i][k]){
    for (each pulse connected by Arc[i][m]){
      for (each RegionI[m][n]){
        If (Eroded Set[i][k] Intersects with
            RegionI[m][n]){
          Add Arc from RegionI[m][n] to
                       Region[i][k];
        }
      }
    }
  }
}
```

REFERENCES

[1] J. Serra, *Image Analysis and Mathematical Morphology, Volume II: Theoretical Advances*. London: Academic Press, 1988, ch. Mathematical Morphology for Boolean Lattices.

[2] G. Matheron, *Image Analysis and Mathematical Morphology, Volume II: Theoretical Advances*. London: Academic Press, 1988, ch. Filters and lattices.

[3] G. Ouzounis and M. Wilkinson, "Countering oversegmentation in partitioning-based connectivities," in *Proc. Int. Conf. Image Processing*, 2005, pp. 844–847.

[4] M. Wilkinson, "Attribute-space connectivity and connected filters," *Image Vis. Comput.*, vol. 25, pp. 426–435, 2007.

[5] R. O'Callaghan and D. Bull, "Combined morphological-spectral unsupervised image segmentation," *IEEE Transactions on Image Processing*, vol. 14, no. 1, pp. 49–62, 2005.

[6] C.-T. Li and R. Wilson, "Image segmentation based on a multiresolution bayesian framework," in *Proceedings of the 1998 International Conference on Image Processing, ICIP*, vol. 3, 4-7 October 1998, pp. 761–765.

[7] W. Law and A. Chung, "Minimal weighted local variance as edge detector for active contour models," in *Computer Vision-ACCV 2006, Lecture Notes in Computer Science*, vol. 3851/2006. Springer-Verlag, Berlin, Heidelberg, 2006, pp. 622–632.

[8] H. Lu and S. Bao, "Physical modeling techniques in active contours for image segmentation," submiteed 22 June, last revision 30 June 2009 2009, cornell University Library Archives.

[9] M. Graham, J. Gibbs, and W. Higgins, "Robust system for human airway-tree segmentation," in *Medical Imaging 2008: Image Processing, Proceedings of SPIE*, J. Reinhardt and P. Pluim, Eds., vol. 6914 69141J-1, 2008.

[10] I. Terol-Villalobos, J. M.-S. nez, and S. Canchola-Magdaleno, "Image segmentation and filtering based on transformations with reconstruction criteria," *Journal of Visual Communication and Image Representation*, vol. 17, pp. 107–130, 2006.

[11] M. Wilkinson, "Connected filtering by reconstruction: basis and new advances," in *Proceedings of 15th IEEE International Conference on Image Processing, ICIP*, 12-15 October 2008 2008, pp. 2180–2183.

[12] P. Salembier and A. Oliveras, *Mathematical Morphology and its Applications to Images and Signal Processing*. Kluwer Academic, 1996, ch. Practical extensions of connected operators, pp. 97–110.

[13] C. Tzafestas and P. Maragos, "Shape connectivity: multiscale analysis and application to generalized granulometries," *Journal of Mathematical Imaging and Vision*, vol. 17, pp. 109–129, 2002.

[14] I. Santillán, A. Herrera-Navarro, J. M.-S. nez, and I. Terol-Villalobos, "Morphological connected filtering on viscous lattices," *Journal of Mathematical Imaging and Vision*, vol. 36, pp. 254–269, 2010.

[15] G. Ouzounis, "Generalized connected morphological operators for robust shape extraction," PhD Thesis, University of Groningen, 2009.

[16] R. Anguelov and I. N. Fabris-Rotelli, "LULU operators and discrete pulse transform for multi-dimensional arrays," *IEEE Transactions on Image Processing*, vol. 19, no. 11, pp. 3012–3023, 2010.

[17] C. Rohwer, *Nonlinear Smoothers and Multiresolution Analysis*. Birkhäuser, 2005.

[18] D. Laurie, "The roadmaker's algorithm for the discrete pulse transform," *IEEE Transactions on Image Processing*, vol. 20, no. 2, pp. 361–371, 2011.

[19] H. Blum, *Models for the Perception of SPeech and Visual Form*. Cambridge: MIT Press, 1967, ch. A Transformation for Extracting New Descriptors of Shape, pp. 362–380.

[20] J. Serra, *Image Analysis and Mathematical Morphology*. Orlando, FL, USA: Academic Press, Inc., 1983.

[21] T. Lindeberg and K. Mardia, "Scale-space theory: A basic tool for analyzing structures at different scales," *Journal of Applied Statistics*, vol. 21, no. 1/2, pp. 225–271, 1994.

[22] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[23] U. Braga-Neto and J. Goutsias, "Grayscale level connectivity: Theory and applications," *IEEE Transactions on Image Processing*, vol. 13, no. 12, pp. 1567–1580, 2004.

Automatic alignment of audiobooks in Afrikaans

Charl J. van Heerden
Multilingual Speech Technologies
North-West University
Vanderbijlpark, South Africa
Email: cvheerden@gmail.com

Febe de Wet[1,2]
[1]Human Language Technology
Competency Area
CSIR Meraka Institute
[2]Department of Electrical and
Electronic Engineering
Stellenbosch University, South Africa
Email: fdwet@csir.co.za

Marelie H. Davel
Multilingual Speech Technologies
North-West University
Vanderbijlpark, South Africa
Email: marelie.davel@gmail.com

***Abstract*—This paper reports on the automatic alignment of audiobooks in Afrikaans. An existing Afrikaans pronunciation dictionary and corpus of Afrikaans speech data are used to generate baseline acoustic models. The baseline system achieves an average duration independent overlap rate of 0.977 on the first three chapters of an audio version of *"Ruiter in die Nag"*, an Afrikaans book by Mikro. The average duration independent overlap rate increases to 0.990 when the speech data from the audiobook is used to perform Maximum A Posteriori adaptation on the baseline models. The corresponding value for models trained on the audiobook data is 0.996. An automatic measure of alignment accuracy is also introduced and compared to accuracies measured relative to a gold standard.**

I. Introduction

Audiobooks are available in many languages. Before the advent of the digital era, books were made available in analogue format. More recently new books are created in digital format and older books that were published on cassettes are gradually being converted to digital format.

Some digital formats facilitate audiobook access and navigation by people who have challenges using regular printed media. DAISY is an internationally established standard for creating digital audiobooks for use by print-disabled people [1]. DAISY books exist in a variety of formats. For some books, both the audio and text are available and the audio and text are aligned at word level. However, many DAISY books are published with limited alignment between audio and text (typically at the chapter level) or with no text at all.

Automatic speech recognition (ASR) technology can enhance audiobook publication in two ways. Firstly, for books that are published as audio only, ASR can be used to generate the text corresponding to existing audio. Secondly, ASR can be used to enhance the level of mark-up for books that are currently only aligned at chapter level. Finer grained alignments between audio and text enable word level search in audiobooks as well as synchronised reading, i.e. the text corresponding to the audio is highlighted during playback.

In this paper we will focus on using ASR technology to align large audio files at word level. The process will specifically be investigated for an under-resourced language for which, until fairly recently, only limited text and speech resources were available, namely Afrikaans. The ultimate aim of the work reported here is to improve the level of mark-up for existing books in any language by automatically converting the recognition output into DAISY *.smil* files. Section II provides some background on previous research on audiobook alignment. The pronunciation dictionary and acoustic data that were used during the study are described in Section III. Section IV describes the ASR systems that were used to perform alignment and Section V introduces a measure to verify alignment accuracy automatically. Results are presented in Section VI and conclusions in Section VII.

II. Background

Word and phone-level alignments between the audio and text versions of audiobooks are used either to enhance the level of accessibility of the books [2], [3] or to develop resources for text-to-speech (TTS) development [4], [5], [6].

A large project was undertaken in Portugal to improve the access to digital audiobooks by print-disabled readers [2]. Amongst other things, an ASR system was developed to automatically align the audio and text at phone level. The authors reported challenges such as bad audio quality of the original analogue recordings, differences of quality within the same book, inconsistent reading of tables, figures, chapter numbers, etc. A pilot corpus was therefore compiled for the development of their alignment system which used a hybrid of Hidden Markov Models (HMMs) and a Multi-Layer Perceptron (MLP) to perform acoustic modelling and a Weighted Finite State Transducer (WFST) framework for pronunciation modelling. The system achieved phone level alignment accuracies of more than 90%. Speaker adaptation as well as pronunciation variation modelling were found to enhance system performance substantially [2]. Pronunciation variation seems especially beneficial to capture phenomena like vowel reduction that are often observed in read speech [2]. In addition to an automatic alignment system, a *Digital Talking Book* player incorporating TTS playback and ASR-enabled navigation were also developed during the same project [3].

From a TTS point of view, aligned audiobooks constitute rich speech databases for more natural acoustic modelling because they capture broader prosodic contexts such as discourse, information structure and affect that are expressed

beyond sentence level. However, many books are published as large, unsegmented audio files and traditional alignment strategies may fail because of the huge memory requirements associated with the alignment of big audio files. In [4] and [5] the authors propose modifications to the Viterbi algorithm that enable the automatic segmentation of large, multi-paragraph speech databases. The proposed technique is independent of the duration of the target audio file.

Another technique that was proposed in the TTS domain is Lightly Supervised alignment [6]. The book under investigation was first segmented into small audio chunks of about 30 seconds each. The resulting audio files were submitted to a two-pass recognition strategy. During the first pass the files were processed by a large-vocabulary, speaker independent system for general segmentation and during the second pass the alignments were improved by using Maximum Likelihood Linear Regression (MLLR) to adapt the models to the speaker specific characteristics of the reader. In addition, the acoustic models are supported by a language model that consists of an interpolation between a general background language model and one trained on the text of the audiobook. The authors show that the proposed approach is able to extract the majority of correctly read sentences without any manual intervention [6].

In this study, automatic alignment was first performed with acoustic models trained on out-of-domain but channel-matched data. Alignment was subsequently repeated using acoustic models that were either adapted using Maximum A Posteriori (MAP) estimation or trained with in-domain data, and the effectiveness of the various approaches compared.

III. Pronunciation dictionary & Speech data

A. Pronunciation dictionary

An existing Afrikaans pronunciation dictionary containing around 24 000 entries [7] was used during system development. Grapheme-to-phoneme (g2p) rules [8] were extracted from the dictionary to generate pronunciations for words in the text that are not in the dictionary.

B. Speech data

In 2010 the National Centre for Human Language Technology (NCHLT) launched a number of projects to support HLT resource development for all 11 official languages of South Africa. During one of these projects broadband (16 kHz) speech corpora were collected for each language. The corpora all contain in the order of 80 to 90 hours of speech data. In this study, the Afrikaans NCHLT speech corpus was used to *train* the baseline acoustic models.

The *test* data constitutes an audio version of *"Ruiter in die Nag"*, an Afrikaans book by Mikro that was published in 1936. The audiobook was originally recorded on analogue tapes in 1960 and was recently converted to digital format. "Ruiter in die Nag" (loosely translated as *"The Rider in the Night"*) was chosen because we had access to both an audio and a text version and because the copyright on it has already expired, so the data can be made available freely for research purposes. The book consists of 17 chapters, each with an average duration of about 12 minutes. In total, it yielded 3.25 hours of read speech produced by a single speaker.

IV. ASR systems

Three different ASR systems were developed in order to evaluate the effect of different acoustic modelling approaches on alignment accuracy. The systems all had the same basic system architecture and were implemented using HTK [9], a well-known Hidden Markov Model Toolkit.

A. Feature extraction

Standard 39-dimensional (13 static, 13 delta and 13 delta-delta) MFCC features were extracted from the data. Cepstral mean and variance normalisation was applied.

B. Acoustic models

All the acoustic models were standard 3-state, left-to-right context dependent triphone HMMs with decision tree clustering and semi-tied transforms, corresponding to the Afrikaans phone set. Three different sets of acoustic models were used to perform alignment: baseline, MAP-adapted and audiobook models.

1) Baseline models: The baseline acoustic models were trained on approximately 90 hours of broadband (16 kHz) Afrikaans speech data from the Afrikaans NCHLT corpus.

2) Maximum A Posteriori (MAP) adapted models: A second set of acoustic models was created by using the speech data from the audiobook to perform MAP adaptation on the baseline models.

3) audiobook models: The third set of acoustic models was trained on the audiobook itself.

V. Automatic alignment verification

Once the audiobook has been aligned, it would be ideal to have a clear measure of the accuracy of the alignment without requiring manual verification. As an automatic measure of alignment accuracy, we compare the difference in the final aligned starting position of each word, with an estimate of the starting position obtained using phoneme recognition.

Specifically, we decode each chapter using a flat phone grammar, creating a single string of phonemes. We also generate a target phoneme string per chapter, using the aligned text and dictionary as input. Forced alignment is used to select the best among competing pronunciation variants. Once these two phone strings have been obtained, we use dynamic programming to find the corresponding phones (and therefore words) in the two strings. As each phone is associated with timing information (either from the alignment, or from the decoding process) we now have two estimates of the word starting position. If there is a discrepancy in starting position estimates, we flag this as a potential alignment error.

This is related to the validation technique used in [10], except that the dynamic programming scores are not used at all, and the difference in timing information is directly used as a confidence measure. As in [10] the dynamic programming process to match the two phone strings can be made more accurate by using a variable cost matrix or, if limited errors in the corpus, a flat scoring matrix can be used.

VI. RESULTS

Manually verified word-level segmentations of the first three chapters of the audiobook were created to serve as a gold standard. Specifically, the alignments obtained using the baseline models were manually verified by a language practitioner and word boundaries moved where these were not correctly aligned with the audio. This is illustrated in Fig. 1: four different alignments are displayed below the waveform and spectrogram. The language practitioner was provided with the first (top) alignment, and moved word boundaries where words were not correctly aligned. This resulted in the gold standard alignment shown fourth (at the bottom). In this example, the word 'oom' was wrongly aligned to the left of the silence portion, and corrected.

Note that, while this provides a trustworthy alignment when identifying word-level errors, the gold standard will at the millisecond-level be biased towards the models that were used to create the initial alignments. See for example the boundaries of the word 'renen'in Fig. 1; these are at identical positions for the gold standard and the first two alignments (baseline and MAP-adapted), but drawn in a slightly different position by the Audiobook models, which are the models that are most different from the initial baseline.

Before extracting final results, the gold standard itself was evaluated. All possible alignment errors of more than 100ms (obtained using the automated verification tools, which does not use the gold standard at all) were flagged for manual evaluation. All segments flagged by all three models were reviewed. This resulted in a subset of 'difficult-to-align' segments that were carefully reviewed for protocol errors, which were corrected if the observed error caused a discrepancy of more than 50ms. Two main protocol errors were observed: silence that was not inserted when needed and word starting points that were not correctly set if a silence preceded the word. 240 segments were reviewed and 24 segments corrected. (An additional random selection of 50 segments resulted in no addiontal corrections.)

The audiobook was already aligned at chapter level. Forced alignment was performed for each chapter individually using ASR systems based on the three sets of acoustic models described in Section IV-B. Alignment accuracy was evaluated by comparing the automatically generated word boundaries to the gold standard. The comparison was quantified in terms of *duration independent overlap rate* (DIOR), defined in [11] as:

$$DIOR = \frac{D_{com}}{D_{max}} = \frac{D_{com}}{D_{ref} + D_{auto} - D_{com}} \quad (1)$$

where D_{com}, D_{max}, D_{ref} and D_{auto} are the common, maximum, reference and automatic durations, respectively. This definition is not as directly applicable to audiobook alignment as to TTS; we therefore propose a modified measure where words are considered correct as long as their start times in the gold and automatic alignments respectively, are within ϵ of each other. At a value of $\epsilon = 100ms$ we obtain the DIOR results reported on in Table I. The values in the table represent the average value over the three chapters for which a gold standard was available.

Acoustic models	Average modified DIOR
Baseline	0.977
MAP-adapted	0.990
audiobook	0.996

TABLE I
AVERAGE MODIFIED DIOR FOR BASELINE, MAP-ADAPTED AND AUDIOBOOK MODELS

Table I shows that using the baseline acoustic models to perform forced alignment already result in an average DIOR of 0.977. This value increases to 0.990 for the MAP-adapted models and to 0.996 for the audiobook acoustic models.

Comparing the gold standard (manually corrected) alignments with the automatically obtained alignments, we find that fairly few errors occur. Table II lists the alignment errors found in the first three chapters of the audiobook, when using different error margins. (These errors represent individual words where the difference in starting time between the automated alignment and the manual alignment is more than the error margin ϵ.

Acoustic models	50ms	100ms	150ms	200ms
Baseline	484	270	182	131
MAP-adapted	334	114	72	46
audiobook	396	61	36	24

TABLE II
ALIGNMENT ERRORS FOR DIFFERENT ERROR MARGINS

If the 50ms margin is not considered, it is clear that the MAP-adapted models provide an accuracy improvement over the baseline, and that the audiobook models are again an improvement over the MAP-adapted models. At the 50ms margin, the superior performance of the MAP-adapted models (over the audiobook models) may be due to the bias of the gold standard, as described in Section VI.

Next, we evaluate our ability to flag possible alignment errors in the final aligned audiobook. Fig. 2 shows Detection Error Trade-off (DET) curves for the three acoustic models. Each curve plots the percentage of true errors flagged versus the percentage of correctly accepted alignments (where the number of true errors flagged depends on the error margin selected). The example illustrated in Fig. 2 corresponds to an error margin of 150ms. The difference in ms between aligned and decoded (estimated) word starting points is used as threshold when constructing the DET curves.

The effect of requiring stricter or more lenient error margins is illustrated in Fig. 3. We compare the DET curves for different error margins and the audiobook acoustic models. At one second, perfect error detection is achieved; at around 150 ms an equal error rate of 0.861 is obtained.

Further error analysis indicated that the main causes of alignment errors were (a) speaker errors resulting in hesitations, missing or repeated words, (b) rapid speech containing

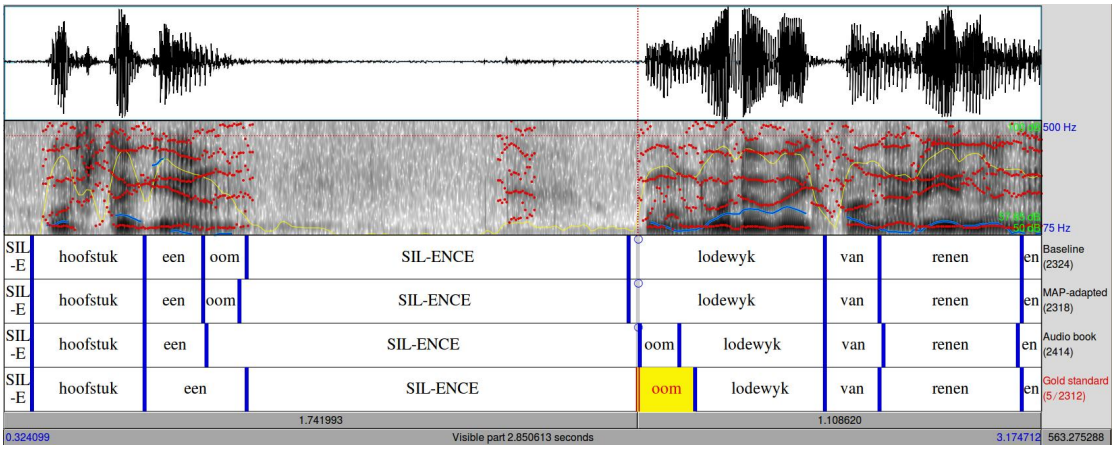


Fig. 1. Example of different alignments obtained for a sentence in the audiobook.

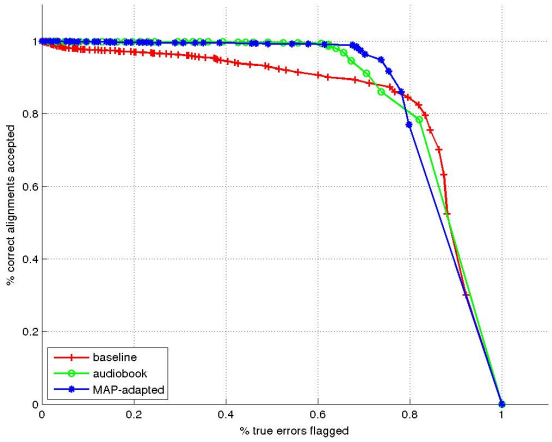


Fig. 2. DET curves for the three acoustic models at a 150ms error margin.

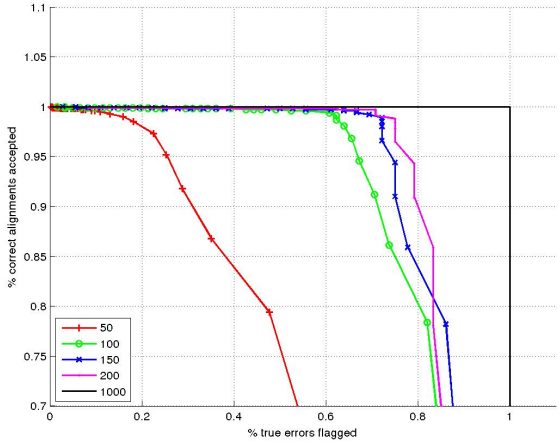


Fig. 3. DET curves for the audiobook acoustic model at different error margins.

contractions, (c) difficulty in identifying the starting position of very short (one- or two-phoneme words) and (d) a few text normalisation errors (for example, 'eenduisend negehonderd' for 'neentienhonderd').

A final observation relates to the applicability of the pronunciation dictionary used. As the alignment verification process associates a decoded phone string with each word, this produces a set of alternative pronunciations that can be considered per word. By counting the number of times the same pronunciation is observed, frequently occurring pronunciations not found in the dictionary can be added and the system retrained. In the current work, initial pronunciations were of sufficient quality that this process was not necessary to improve alignment quality, but for audiobooks that contain large numbers of unknown words (such as expected from study guides or other technical material) this may be a useful addition to the process.

VII. Conclusions

The results obtained in this study indicate that the alignments obtained by a baseline system are already good enough for practical purposes, i.e. to provide word-level mark-up for DAISY books. They also show that alignment accuracy can be improved by performing MAP adaptation on the baseline models – a fast and efficient solution requiring minimal computation. The best results are obtained with acoustic models trained on the target audiobook.

We have also shown that dynamic programming can be used to align the freely decoded and forced aligned phone strings associated with each chapter to yield an automatic measure of alignment accuracy. Error margins are defined in terms of the difference between estimated starting positions of words in the two phone strings. For an error margin of 150 ms the technique is able to accept correct alignments and flag true errors with an accuracy of 86%. For a larger error margin (of 1 second), 100% accurate alignment accuracy is achieved: all true alignment errors are rejected, and all accurately aligned words are correctly accepted.

The process will be repeated for additional audiobooks in the near future. While the voice artist spoke very rapidly, the audiobook contained few speaker errors; it would be useful to understand the extent to which a larger percentage of errors can be tolerated (and identified during alignment verification). Follow-up research will also investigate the impact of using gender-dependent baseline models on the alignment accuracy of the final systems as well as the bias of the gold standard towards the initial alignments. The results will be used to design an automated process that can be used to align large volumes of audiobooks in a fully automated way.

ACKNOWLEDGEMENTS

We would like to thank Willem van der Walt for sparking our interest in audiobook alignment and for providing information on DAISY books.

REFERENCES

[1] "Daisy," 2012, http://www.daisy.org/, Accessed in October 2012.

[2] A. Serralheiro, D. Caseiro, H. Meinedo, and I. Trancoso, "Word alignment in digital talking books using WFSTs," *Research and Advanced Technology for Digital Libraries - Lecture Notes in Computer Science*, vol. 2458/2002, pp. 508–515, 2002.

[3] I. Trancoso, C. Duarte, A. Serralheiro, D. Caseiro, L. Carrico, and C. Viana, "Spoken language technologies applied to digital talking books," in *Proceedings of Interspeech*, 2006.

[4] K. Prahallad, A. R. Toth, and A. W. Black, "Automatic building of synthetic voices from large multi-paragraph speech databases," in *Proceedings of Interspeech*, 2007.

[5] K. Prahallad and A. W. Black, "Segmentation of monologues in audio books for building synthetic voices," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 19, no. 5, pp. 1444–1449, July 2011.

[6] N. Braunschweiler, M. Gales, and S. Buchholz, "Lightly supervised recognition for automatic alignment of large coherent speech recordings," in *Proceedings of Interspeech*, 2010, pp. 2222–2225.

[7] M. Davel and F. de Wet, "Verifying pronunciation dictionaries using conflict analysis," in *Proceedings of Interspeech*, Tokyo, Japan, 2010, pp. 1898–1901.

[8] M. Davel and E. Barnard, "Pronunciation prediction with Default&Refine," *Computer Speech and Language*, vol. 22, pp. 374–393, 2008.

[9] S. J. Young, G. Evermann, M. J. F. Gales, D. Kershaw, G. Moore, J. J. Odell, D. G. Ollason, D. Povey, V. Valtchev, and P. C. Woodland, "The HTK book version 3.4." Cambridge, UK, 2006.

[10] M. Davel, C. J. van Heerden, and E. Barnard, "Validating smartphone-collected speech corpora," in *Proceedings of SLTU*, Cape Town, South Africa, May 2012, pp. 68–75.

[11] S. Paulo and L. C. Oliveira, "Automatic phonetic alignment and its confidence measures," *Advances in Natural Language Processing*, 2004.

Mean Shift Object Tracking with Occlusion Handling

B.Z. de Villiers
HyperVision Research Lab
University of Johannesburg
South Africa
Email: wheres.brett.dev@gmail.com

W.A. Clarke
HyperVision Research Lab
University of Johannesburg
South Africa
Email: willemc@uj.ac.za

P.E. Robinson
HyperVision Research Lab
University of Johannesburg
South Africa
Email: philipr@uj.ac.za

***Abstract*—An object tracking algorithm using the Mean Shift framework is presented which is largely invariant to both partial and full occlusions, complex backgrounds and change in scale. Multiple features are used to gain a descriptive representation of the target object. Image moments are used to determine the scale of the target object. A kalman filter is used to successfully track the target object through partial and full occlusions, the Bhattacharyya coefficient is used to determine the measurement noise estimation.**

I. Introduction

Object tracking is of great importance in computer vision and is used in many applications such as visual surveillance, perceptual user interfaces, augmented reality and intelligent transport systems. Mean Shift [1] is a popular method used in object tracking which is also used in commercial applications due its simple implementation, efficient and robust performance. The Mean Shift method is a non-parametric, variable step-size, statistical density estimator which iteratively determines the nearest mode of a point sample distribution using gradient ascent. The Mean Shift method has been used in a number of computer vision problems, these include line fitting [2], image segmentation [3] and object tracking [4].

A number of improvements to the traditional formulation of the Mean Shift method for object tracking have been investigated [4]. Multiple features have been investigated to gain a more descriptive representation of the target object [5,6]. In [5] various colour spaces and edge directions are used as descriptive features, feature localization weights are determined according to the similarity between background features and features present in the target model. In [6] the RGB colour space, edge directions and textural information (obtained using the discrete wavelet transform) are used as descriptive features, feature localization weights are determined according to the similarity between target candidate features and features present in the target model. Scale space theory was adopted in order to successfully determine the target object's scale during tracking [7]. The Mean Shift method was applied to Gaussian kernels at various scales to determine the target object's scale. Image moments have been used with the similarity weights (between the target model and candidate) to determine the scale and orientation of the target object [8]. Multiple ellipsoidal, asymmetric kernels with asymmetric centres have been used to effectively track target position, scale and orientation simultaneously [9]. In order to remove background features from the target model and candidate a level set function has been used along the contour of the target object [10]. The level set function defines an asymmetric kernel over the target region which does not contain any background features. Mean Shift is used to track the target object's position, scale and orientation.

This paper proposes a tracking algorithm using the Mean Shift framework which is largely invariant to both partial and full occlusions, complex backgrounds and change in scale. Multiple features are used to gain a more descriptive representation of the target object, these features include colour, edges and texture. An adaptive feature weighting method is used to maximize the feature weights of features which better localize the target object. Image moments are used in conjunction with the similarity weights (between the target model and candidate) to determine the scale of the target object. A kalman filter is used to improve the tracking performance during partial and full occlusions, a measurement noise estimation is determined using the Bhattacharyya coefficient [11].

The paper is arranged as follows. Section II provides an overview of the Mean Shift tracking algorithm [4]. Section III provides a description of the various features used to describe the target object. Section IV provides details on the tracking algorithm including scale selection, kalman filter implementation and a brief overview of the tracking algorithm. Section V provides experimental results which describe the performance of the tracking algorithm. Section VI concludes the paper.

II. Mean Shift Tracking Algorithm

A. Target Representation

A target is typically defined by an ellipsoidal region or patch surrounding a region of interest in an image. A feature space is chosen (typically the RGB feature space is used) to determine a histogram of the pixel distribution in the target region. The

histogram is represented by target model q. The target model is used to describe the appearance of the object located in the target region. The target model q is comprised of m normalized bins [4].

Target model:

$$\hat{q} = \{\hat{q}_u\}_{u=1...m} \tag{1}$$

$$\sum_{u=1}^{m} \hat{q}_u = 1 \tag{2}$$

Let $\{x_i^*\}_{i=1...n}$ denote the n normalized pixel locations in the target region which are centred around 0. Let $k(x)$ denote a convex, monotonically decreasing, isotropic kernel. Let b: $R^2 \rightarrow \{1...m\}$ be a function which determines the histogram bin $b(x_i^*)$ associated with the pixel location x_i^*. The probability of the feature $u = 1...m$ in the target models histogram is determined by

$$\hat{q}_u = C \sum_{i=1}^{n} k(\| x_i^* \|^2)\delta[b(x_i^*) - u] \tag{3}$$

Where δ is the Kronecker delta function. The normalization constant C is derived by imposing the condition (2), normalization constant C can therefore be represented by

$$C = \frac{1}{\sum_{i=1}^{n} k(\| x_i^* \|^2)} \tag{4}$$

B. Candidate Representation

Typically the target model is formed from the target region in the first frame of a video sequence. The target model is compared to candidate regions in the current frame to determine the location and scale of the target in the current frame. A target candidate $p(y)$ is defined by a histogram of the pixel distribution of a region in the current frame. The target candidate $p(y)$ is comprised of m normalized bins [4].

Target candidate:

$$\hat{p}(y) = \{\hat{p}_u(y)\}_{u=1...m} \tag{5}$$

$$\sum_{u=1}^{m} \hat{p}_u(y) = 1 \tag{6}$$

Let $\{x_i\}_{i=1...n_h}$ denote the n_h normalized pixel locations in the candidate region which are centred around y. Let $k(x)$ denote the same convex, monotonically decreasing, isotropic kernel used with the target model only with a different size (based on the scale of the target object) specified by bandwidth h. The probability of the feature $u = 1...m$ in the target candidates histogram is determined by

$$\hat{p}_u(y) = C_h \sum_{i=1}^{n_h} k(\| \frac{y - x_i}{h} \|^2)\delta[b(x_i) - u] \tag{7}$$

where

$$C_h = \frac{1}{\sum_{i=1}^{n_h} k(\| \frac{y-x_i}{h} \|^2)} \tag{8}$$

C. Similarity Model

In order to determine the similarity between the target model and the target candidate a similarity function is determined. The similarity function used is the sample estimate of the Bhattacharyya coefficient [11] between the distributions $\hat{q}$ and $\hat{p}(y)$. The similarity function is defined by

$$\hat{\rho}(y) = \rho[\hat{p}(y), \hat{q}] = \sum_{u=1}^{m} \sqrt{\hat{p}_u(y)\hat{q}_u} \tag{9}$$

Due to the conditions imposed by (2) and (6) the similarity function has a minimum value of 0 (distributions are orthogonal) and a maximum value of 1 (distributions are equal).

D. Mean Shift Vector

The Mean Shift algorithm iteratively samples target candidate locations in an effort to find the local maximum of the similarity function $\hat{\rho}(y)$. By taking the Taylor expansion around the target candidate probability values $\hat{p}_u(\hat{y}_0)$ (where the target candidate $\hat{p}(\hat{y}_0)$ is centred around $\hat{y}_0$) the estimated linear approximation of the Bhattacharyya coefficient [4] can be described by

$$\rho[\hat{p}(y), \hat{q}] = \frac{1}{2} \sum_{u=1}^{m} \sqrt{\hat{p}_u(\hat{y}_0)\hat{q}_u} + \frac{1}{2} \sum_{u=1}^{m} \hat{p}_u(y) \sqrt{\frac{\hat{q}_u}{\hat{p}_u(\hat{y}_0)}} \tag{10}$$

The first term of (10) is independent of position y, therefore to maximize $\rho[\hat{p}(y), \hat{q}]$ it is necessary to maximize the second term of (10), using (7) the second term of (10) denoted by $\rho[\hat{p}(y), \hat{q}]_2$ can be described by

$$\rho[\hat{p}(y), \hat{q}]_2 = \frac{C_h}{2} \sum_{i=1}^{n_h} \omega_i k(\| \frac{y - x_i}{h} \|^2) \tag{11}$$

where

$$\omega_i = \sum_{u=1}^{m} \sqrt{\frac{\hat{q}_u}{\hat{p}_u(\hat{y}_0)}} \delta[b(x_i) - u] \tag{12}$$

The Mean Shift vector is determined in order to maximize the similarity function $\hat{\rho}(y)$ by maximizing (11). The Mean Shift vector is determined by

$$Y_1 = \frac{\sum_{i=1}^{n_h} (x_i - \hat{y}_0)\omega_i g(\| \frac{\hat{y}_0 - x_i}{h} \|^2)}{\sum_{i=1}^{n_h} \omega_i g(\| \frac{\hat{y}_0 - x_i}{h} \|^2)} \tag{13}$$

Where $g(x) = k'(x)$. If we choose $k(x)$ to use the Epanechnikov profile [12] described by

$$k(x) = \begin{cases} \frac{1}{2} c_d^{-1}(d+2)(1-x) & \text{if } x \leq 1 \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

the computation of (13) can be simplified as $g(x)$ becomes a constant. Different kernel profiles may be used, they however have little impact on the localization accuracy of the Mean Shift algorithm. These kernel profiles have a higher computational cost as the kernel derivative $g(x)$ must be determined

for each computation of the Mean Shift vector. Using the Epanechnikov profile the Mean Shift vector can be described by

$$Y_1 = \frac{\sum_{i=1}^{n_h}(x_i - \hat{y}_0)\omega_i}{\sum_{i=1}^{n_h}\omega_i} \tag{15}$$

The updated position of the target candidate position $\hat{y}_1$ is simply described by

$$\hat{y}_1 = \hat{y}_0 + Y_1 \tag{16}$$

The Mean Shift algorithm is run recursively until convergence, convergence occurs when the Mean Shift vector is lower than a tolerance ϵ. The tolerance is usually chosen to be the width of a single pixel.

III. Image Features

Multiple Image features were used during tracking in order to better describe the appearance of the target object.

A. Local Binary Pattern Features

The local binary pattern [13,14] is an image operator which transforms an image into an array of integer labels which describe the small scale appearance of the image [14]. The LBP (local binary pattern) is an efficient texture classification method which is invariant to monotonic grey level changes. The local binary pattern was used to provide useful textural descriptive information of the target object.

The basic LBP [13] was initially designed for texture description. The basic LBP operator assigns a label to each pixel in the image. Let $z(x, y)$ describe the 3 $\times$ 3 neighbourhood surrounding a pixel. $z(x, y)$ is described by

$$z(x, y) = I(x, y) - I(x_c, y_c) \tag{17}$$

Where $I(x, y)$ represents the pixel values in the 3 $\times$ 3 neighbourhood and $I(x_c, y_c)$ represents the centre pixel in the 3 $\times$ 3 neighbourhood. Let $s(z(x, y))$ be the thresholding step function where

$$s(z(x,y)) = \begin{cases} 1 & \text{if } z(x,y) \geq 0 \\ 0 & \text{if } z(x,y) < 0 \end{cases} \tag{18}$$

The pixels surrounding the centre pixel in $s(z(x, y))$ form a binary number which is used as a label to describe the pixel. Fig. 1 shows an illustration of the basic LBP operator. A histogram of these labels can be used to describe the image.

Traditionally the histogram describing a texture or image is determined by separating uniforms patterns (such as 00000000 or 11001111) into bins. Where each unique uniform pattern has a preallocated bin and all non-uniform patterns are grouped in a single bin. There are 58 unique uniform patterns in the basic LBP and 198 non-uniform patterns [14]. In order to improve the rotational invariance of the LBP, the binary label for each pixel is circularly bit-shifted to find a minimum binary value which describes the pixel for eight possible orientations of the LBP operator. This is shown by

$$LBP^{r,i}_{P,R} = \min_i ROR(LBP_{P,R}, i) \tag{19}$$

Where $LBP^{r,i}_{P,R}$ denotes the output rotationally invariant binary label, $ROR(x, i)$ denotes the circular bitwise right rotation of bit sequence x by i steps and $LBP_{P,R}$ denotes the original basic LBP binary label.

Performing this rotation invariance step is useful in that it allows the LBP to perform robustly when rotation occurs as well as limiting the number of possible unique uniform patterns. The unique uniform patterns are reduced to the following 9 patterns 00000000, 00000001, 00000011, 00000111, 00001111, 00011111, 00111111, 01111111, 11111111 after the rotation invariance step.

The basic LBP operator with the rotation invariance step was used for each channel in the RGB colour space. A 3-dimensional RGB-LBP histogram with 10 bins per channel was formed from the 3 channels R, G and B.

B. Edge Features

Edges describe the structure of an image, edges provide beneficial descriptive information in object tracking when objects in a scene have similar colour yet different structure. A 2-dimensional edge histogram of size $N_e \times N_e$ with one channel for edge magnitude and the other for edge direction is used to describe the edge features in the target object. The simple Scharr operator [15] was used to find edges in the image as it provides efficient, robust and rotational invariant edge detection. The gradients $D_x(x, y)$ and $D_y(x, y)$ are represented by

$$D_x(x, y) = S_x \bigotimes I(x, y) \tag{20}$$

$$D_y(x, y) = S_y \bigotimes I(x, y) \tag{21}$$

Where $D_x(x, y)$ is the gradient in the x direction, $D_y(x, y)$ is the gradient in the y direction, S_x is the simple Scharr gradient operator in the x direction and S_y is the simple Scharr gradient operator in the y direction, $\bigotimes$ is the convolution operator and I(x,y) represents the intensity values in the image. The edge magnitude denoted by $D(x, y)$ and the gradient direction denoted by $\theta(x, y)$ are represented by

$$D(x, y) = \sqrt{D_x(x, y)^2 + D_y(x, y)^2} \tag{22}$$

$$\theta(x, y) = \arctan(\frac{D_y(x, y)}{D_x(x, y)}) \tag{23}$$

Where $\theta(x, y)$ is determined between edges directions $0^\circ \leq \theta(x, y) < 360^\circ$. Edges were filtered such that only edges with magnitudes above a threshold t_e were considered in the edge feature histogram.

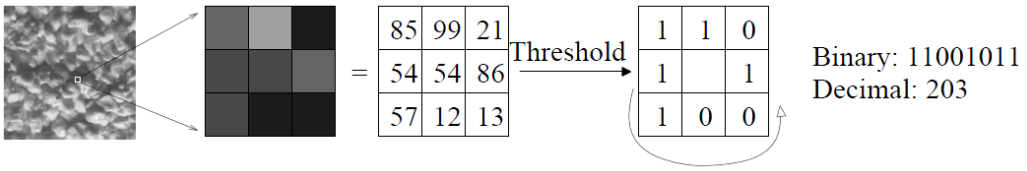


Fig. 1: Local Binary Pattern Operator

C. Colour Features

Colour histograms are most commonly used in conjunction with the Mean Shift algorithm as they are robust to partial occlusion and change in scale and rotation. They perform well under non-rigid deformations of the target object and changing complex backgrounds [4,12]. Colour histograms do however fail when other objects or background features have the same or similar colour. A 3-dimensional RGB colour histogram of size $N_c \times N_c \times N_c$ was used to describe the RGB colour distribution of the target object. A 1-dimensional Hue (from the HSV colour space) colour histogram of size N_h was used to describe the Hue colour distribution of the target object. The Hue histogram is useful as it is largely illumination invariant.

D. Colour and Edge Features

Colour and Edge features where combined in an effort to combine structural and colour information in a single histogram. Edges were found using the simple Scharr operator. The greyscale gradient magnitude $D(x, y)$ was determined for each pixel in the target object region. The pixel value $I_i(x, y)$ for each RGB channel is determined by.

$$I_i(x,y) = \begin{cases} I_i(x,y) + D_i(x,y) & \text{if } D(x,y) \leq t_e \\ I_i(x,y) - D_i(x,y) & \text{if } D(x,y) > t_e \end{cases} \quad (24)$$

Where $I_i(x, y)$ is i'th RGB channel value for the pixel $I(x, y)$ and $D_i(x, y)$ is the gradient magnitude for the RGB channel i. A 3-dimensional colour-edge histogram of size $N_c \times N_c \times N_c$ was used to describe $I_i(x, y)$.

Let σ denote the scale of the target object. Due to the elliptical shape of the target region, typically both background and object features are present in the target region of scale σ [10]. Background features in the target model can have an effect on the localization accuracy of the tracking algorithm. In order to minimize this effect 3 colour-edge histograms were used to describe the target object. The 3 colour-edge histograms were determined for target regions of scales σ, 0.8σ and 0.6σ. Histograms formed from target regions smaller than the scale of the object are less likely to contain background features.

E. Background Weighted Colour Features

If some background features are present in the target model and candidate, the localization performance would be improved if the background feature information in the target model and target candidate was suppressed. This is done by weighting the target model and target candidate with a background model at each frame such that the target object has a more salient description relative to the background [4].

Let $\hat{o}(y)$ denote the background model centred around y. Let $\{x_i\}_{i=1...n_h}$ denote the n_h normalized pixel locations in the background model region which are centred around y. Let $a(x)$ denote a concave, monotonically increasing, isotropic kernel with a size (based on the scale of the target object) specified by bandwidth h. The probability of the feature $u = 1...m$ in the background model histogram is determined by

$$\hat{o}_u(y) = C_h \sum_{i=1}^{n_h} a(\| \frac{y - x_i}{h} \|^2) \delta[b(x_i) - u] \quad (25)$$

where

$$C_h = \frac{1}{\sum_{i=1}^{n_h} a(\| \frac{y-x_i}{h} \|^2)} \quad (26)$$

The background kernel used is described by

$$a(x) = \begin{cases} 0 & \text{if } x \leq 1 \\ x - 1 & \text{if } 1 < x \leq 2 \\ 0 & \text{otherwise} \end{cases} \quad (27)$$

Where 1 represents the boundary of the target model or candidate region and 2 represents the boundary of the background model region. The kernel $a(x)$ assigns weights to pixels such that features further from the object boundary have a higher weighting. Let $\hat{o}^*$ denote the smallest non-zero histogram bin in the background histogram $\hat{o}(y)$. The scaling array v_u [4] used to minimize similar features between the background model and the target model and candidate is described by

$$\{v_u = \min(\frac{\hat{o}^*}{\hat{o}_u}, 1)\}_{u=1...m} \quad (28)$$

The background weighted target model $\hat{q}_u$ and target candidate $\hat{p}_u(y)$ are represented by

$$\hat{q}_u = C v_u \sum_{i=1}^{n} k(\| x_i^* \|^2) \delta[b(x_i^*) - u] \quad (29)$$

where

$$C = \frac{1}{\sum_{i=1}^{n} k(\| x_i^* \|^2) \sum_{u=1}^{m} v_u \delta[b(x_i^*) - u]} \quad (30)$$

$$\hat{p}_u(y) = C_h v_u \sum_{i=1}^{n_h} k(\| \frac{y - x_i}{h} \|^2) \delta[b(x_i) - u] \quad (31)$$

where

$$C_h = \frac{1}{\sum_{i=1}^{n_h} k(\| \frac{y-x_i}{h} \|^2) \sum_{u=1}^{m} v_u \delta[b(x_i^*) - u]} \quad (32)$$

A 3-dimensional background weighted colour histogram of size $N_c \times N_c \times N_c$ was used to describe a more salient RGB colour representation of the target object.

IV. Target Object Localization

A. Feature Localization Weights

Each feature determines an updated target object position $\hat{y}_1$ using the Mean Shift localization algorithm. To determine the best estimation of the target object's updated position, a weighted average is determined of the updated target object positions determined by the various features. The updated target object position $\hat{y}_1$ is determined by

$$\hat{y}_1 = \sum_{j=1}^{K_f} \omega_j \hat{y}_{1_j} \quad (33)$$

Where ω_j denotes the localization weight for feature j, $\hat{y}_{1_j}$ denotes the updated target object position for feature j and K_f denotes the number of features. The feature weights are determined from 3 global weights, The global weights consist of predetermined feature weights, model-candidate similarity feature weights and model-background similarity feature weights.

The global model-candidate similarity feature weight determines a weight based on the similarity function between the target model and target candidate. The higher the similarity, the higher the weight associated with the feature. The model-candidate similarity feature weight ω_c is described by

$$\omega_{c_j} = \frac{1}{(1 - \rho[\hat{p}_j(y), \hat{q}_j])(C_c)} \quad (34)$$

where

$$C_c = \sum_{j=1}^{K_f} \frac{1}{(1 - \rho[\hat{p}_j(y), \hat{q}_j])} \quad (35)$$

Where ω_{c_j} denotes the model-candidate similarity feature weight for feature j, $\hat{q}_j$ denotes the target model for feature j and $\hat{p}_j(y)$ denotes the target candidate for feature j. The global model-background similarity feature weight determines a weight based on the similarity function between the target model and background model. The higher the similarity, the lower the weight associated with the feature. The model-background similarity feature weight ω_b is described by

$$\omega_{b_j} = \frac{\omega_{p_j}}{(\rho[\hat{o}_j(y), \hat{q}_j])(C_b)} \quad (36)$$

where

$$C_b = \sum_{j=1}^{K_f} \frac{\omega_{p_j}}{(\rho[\hat{o}_j(y), \hat{q}_j])} \quad (37)$$

Where ω_{b_j} denotes the model-background similarity feature weight for feature j, $\hat{o}_j(y)$ denotes the background model for feature j and ω_{p_j} denotes predetermined feature weight for feature j. The localization weight ω_j for the feature j is determined by

$$\omega_j = \alpha\omega_{c_j} + \beta\omega_{b_j} + \gamma\omega_{p_j} \quad (38)$$

where

$$\alpha + \beta + \gamma = 1 \quad (39)$$

Where α, β and γ are constants which specify the relationship between the various global weights and the feature weights. The features weights are normalized such that $\sum_{j=1}^{K_f} \omega_j = 1$.

B. Scale Selection

It is necessary to determine the scale of the target object to effectively track it through out a video sequence. Image moments [16,17] are used to determine the scale of the target object in this algorithm, a similar approach is used by [8] and [18]. In [18] (CAMSHIFT) the scale and orientation is determined using image moments on a skin probability back projection. In [8] (SOAMST) the traditional kernel-based Mean Shift object tracking algorithm is used, the similarity weights ω_i (12) are used as a probability back projection. Image moments are used with the similarity weights to determine the scale and orientation of the target object. A similarity area estimation is used to correctly determine the target object's scale.

In this algorithm the similarity weights ω_i are determined for each pixel in the target region with a scale 1.2σ. Image moments are then used in conjunction with the similarity weights to determine the scale of target object in the current frame. The similarity weights ω_i are determined by

$$\omega_i = \sum_{j=1}^{K_f} \omega_j \omega_{i_j} \quad (40)$$

Where ω_{i_j} denotes the similarity weight determined by (12) for feature j. The zeroth order moment denoted by M_{00} is determined by

$$M_{00} = \sum_{i=1}^{n_h} \omega_i \quad (41)$$

Where n_h is the number of pixels in the target region with a scale 1.2σ. The second order moments denoted by M_{20}, M_{02} and M_{11} are determined by

$$M_{20} = \sum_{i=1}^{n_h} \omega_i x_{i,1}^2 \quad (42)$$

$$M_{02} = \sum_{i=1}^{n_h} \omega_i x_{i,2}^2 \quad (43)$$

$$M_{11} = \sum_{i=1}^{n_h} \omega_i x_{i,1} x_{i,2} \quad (44)$$

Where $x_{i,1}$ denotes the i'th x value in the target region with a scale 1.2σ and $x_{i,2}$ denotes the i'th y value in the target region with a scale 1.2σ. The second order central moments denoted by μ_{20}, μ_{02} and μ_{11} are determined by

$$\mu_{20} = \frac{M_{20}}{M_{00}} - \bar{x}_1^2 \tag{45}$$

$$\mu_{02} = \frac{M_{02}}{M_{00}} - \bar{x}_2^2 \tag{46}$$

$$\mu_{11} = \frac{M_{11}}{M_{00}} - \bar{x}_1\bar{x}_2 \tag{47}$$

Where $\bar{x}_1$ is the target object's centre x position and $\bar{x}_2$ is the target object's centre y position. The second order central moment covariance matrix donated by Cov is represented by

$$Cov = \begin{bmatrix} \mu_{20} & \mu_{11} \\ \mu_{11} & \mu_{02} \end{bmatrix} \tag{48}$$

The eigenvalues of the covariance matrix represent the size of the axis a and b of the target object region. Half the height of the target object is determined by b and half the width is determined by a, they are represented by

$$a = \frac{\mu_{20} + \mu_{02}}{2} - \frac{\sqrt{4\mu_{11}^2 + (\mu_{20} - \mu_{02})^2}}{2} \tag{49}$$

$$b = \frac{\mu_{20} + \mu_{02}}{2} + \frac{\sqrt{4\mu_{11}^2 + (\mu_{20} - \mu_{02})^2}}{2} \tag{50}$$

It is assumed that the scale change between frames is relatively small, to get a more accurate and smooth scale change between frames the target height and width is determined by

$$a = (\zeta)a_p + (1 - \zeta)a_n \tag{51}$$

$$b = (\zeta)b_p + (1 - \zeta)b_n \tag{52}$$

Where (ζ) denotes a constant which determines the rate at which the target object's scale should change, a_p denotes half the object width determined in the previous frame, a_n denotes half the object width determined in the current frame, b_p denotes half the object height determined in the previous frame and b_n denotes half the object height determined in the current frame.

C. State Estimation

The Mean Shift algorithm is not well suited for tracking objects in the presence of full occlusions. In order to improve the performance of the Mean Shift tracking algorithm in the presence of partial and full occlusions a kalman filter [19,20] is used. A kalman filter is a state estimation algorithm which compares state prediction against state measurements to get an accurate estimation of the true state.

The state prediction matrix F in $X_k = FX_{k-1} + v_k$ is determined using simple equations of motion for position, velocity and acceleration. The state prediction matrix also called the system matrix is represented by

$$F = \begin{bmatrix} 1 & 0 & 1 & 0 & \frac{1}{2} & 0 \\ 0 & 1 & 0 & 1 & 0 & \frac{1}{2} \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{53}$$

For the kalman filter to perform accurately the measurement noise n_k [20] needs to be estimated. The measurement noise n_k is used to estimate how reliable the measurements are in $z_k = HX_k + n_k$. An accurate estimation of the measurement noise is necessary in order to minimize the effect of inaccurate target object localization during occlusion. The measurement noise n_k is determined relative to the similarity between the target model and the target candidate, the more similar the target model and candidate, the more accurate the measurement. The measurement noise n_k is described by

$$\{n_k = 10^l(1 - \rho[\hat{p}(y), \hat{q}]_c) \quad \text{if } \epsilon_{l+1} < \rho[\hat{p}(y), \hat{q}]_c \leq \epsilon_l\}_{l=0\ldots3} \tag{54}$$

where

$$\rho[\hat{p}(y), \hat{q}]_c = \sum_{j=1}^{K_f} \rho[\hat{p}_j(y), \hat{q}_j]\omega_j \tag{55}$$

Where $\epsilon_{l\,\{l=0\ldots3\}}$ are constants which specify the bounds of the piecewise measurement noise estimation function. It is assumed that a target object's velocity is constant during occlusion. Using this assumption in order to improve the tracking performance during occlusion, the current state matrix velocity is updated every frame with the target object's weighted average velocity V_{a_k} represented by

$$V_{a_k} = 0.85V_{a_{k-1}} + 0.15((1 - V_{n_k})V_k + V_{n_k}V_{a_{k-1}}) \tag{56}$$

Where V_{n_k} is the velocity noise at frame k determined by

$$\{V_{n_k} = 0.2l \quad \text{if } \epsilon_{l+1} < \rho[\hat{p}(y), \hat{q}]_c \leq \epsilon_l\}_{l=0\ldots3} \tag{57}$$

The state matrix velocity X_{k_v} is updated with the weighted average velocity such that $X_{k_v} = 0.85V_{a_k} + 0.15X_{k_v}$.

D. Tracking Algorithm Overview

Using the methods described in sections II, III and IV, the tracking algorithm can be summarized as follows

1) Determine target model $\hat{q}_j$ for features $1...j$
2) Initialize iteration number $k_i \leftarrow 0$
3) Initialize position y_0 of candidate target in current frame
4) Determine candidate target $\hat{p_j}(y_0)$ for features $1...j$
5) Calculate feature localization weights w_j for features $1...j$
6) Calculate similarity weights ω_{i_j} for features $1...j$
7) Calculate combined similarity weights ω_i

8) Determine updated target object position y_1
9) If $\| y_1 - y_0 \| < \epsilon$ (where $\epsilon < 1$) or if $k \geq N$ (where N is chosen to be 20) stop. Go to step 10)
   Otherwise $k_i \leftarrow k_i + 1$ and $y_0 \leftarrow y_1$. Go to step 4)
10) Determine height $2b$ and width $2a$ of target object
11) Update target object states using kalman filter, this includes updating object position. Determine y_0 for the next frame using state prediction matrix F
12) Load next frame, go to step 2)

V. Experimental Results

The proposed algorithm's performance is compared to the original Mean Shift tracking algorithm with variable scale selection in [4] and the SOAMST algorithm in [8]. These algorithms were selected to use $64 \times 64 \times 64$ RGB colour histograms, the algorithms in [4] and [8] were implemented using the same kalman filter implementation used in the proposed tracking algorithm. The algorithms were tested on a complex scene (video sequence: motinas_multi_face_frontal.avi, frames: 1 - 300, target: Emilio) [21]. A persons face (Target: Emilio) was tracked in a complex environment with partial and full occlusions, change in scale, change in illumination and slight change in the appearance of the target object. During the video sequence the target's face is fully occluded by the face of a person (target: Joe, frames: 88 - 95). There is a rapid change in scale of the target (frames: 250 - 300) and a change in illumination experienced by the target (frames: 196 - 275).

The tracking performance of the algorithms can be observed from Fig. 2 (visual description of tracking performance for proposed algorithm, original Mean Shift tracking algorithm and SOAMST algorithm) and Fig. 3 (graphs describing position and scale selection error from ground truth). The original Mean Shift object tracking algorithm shows good performance in tracking the target object, however once occlusion occurs the tracker diverges, the algorithm does not benefit greatly from the kalman filter implementation. The SOAMST algorithm shows good performance in tracking the target object, however the algorithm selects the scale of the target object abruptly and inaccurately. Like the original Mean Shift algorithm the SOAMST algorithm diverges when occlusion occurs and does not benefit greatly from the kalman filter implementation. The proposed algorithm shows good performance in tracking the target object through out the video sequence. The algorithm localizes the target object inaccurately during occlusion, however the algorithm does not diverge during occlusion. The proposed tracking algorithm benefits greatly from the kalman filter implementation in minimizing the effect of object occlusion.

VI. Conclusion

A tracking algorithm using the Mean Shift framework is presented which performs robustly in complex scenes where occlusion occurs. The algorithm uses multiple features to uniquely describe objects, image moments to effectively determine the target object's scale and a kalman filter to aid the localization algorithm during occlusion. The algorithm has shown superior tracking performance in complex scenes when compared to the original Mean Shift tracking algorithm and the scale adaptive SOAMST algorithm.

References

[1] K. Fukunaga and L. Hostetler, "The Estimation of the Gradient of a Density Function, with Applications in Pattern Recognition," in *IEEE IT*, vol. 21, no. 1, pp. 32 - 40, 1975.

[2] Y. Cheng "Mean Shift, Mode Seeking, and Clustering," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, pp. 790 - 799, 1995.

[3] D. Comaniciu and P. Meer, "Mean Shift Analysis and Applications," in *International Conference on Computer Vision*, vol. 2, pp. 1197 - 1203, 1999.

[4] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-Based Object Tracking," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, pp. 564 - 577, May 2003

[5] J. Wang and Y. Yagi, "Integrating Shape and Color Features for Adaptive Real-time Object Tracking," in *IEEE International Conference on Robotics and Biomimetics*, pp. 1 - 6, 2006

[6] A. Babaeian, S. Rastegar, M. Bandarabadi and M. Rezaei, "Mean Shift-Based Object Tracking with Multiple Features," in *41st Southeastern Symposium on System Theory*, pp. 68 - 72, March 2009

[7] R. T. Collins, "Mean-shift blob tracking through scale space," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 234 - 240 2003

[8] J. Ning, L. Zhang1, D. Zhang and C. Wu, "Scale and Orientation Adaptive Mean Shift Tracking," in *Computer Vision, IET*, vol. 6, iss. 1, pp. 52 - 61, 2012

[9] S. Zhang and Y. Bar-Shalom, "Robust Kernel-Based Object Tracking with Multiple Kernel Centers," in *12th International Conference on Information Fusion*, pp. 1014 - 1021, July 2009

[10] A. Yilmaz, "Object Tracking by Asymmetric Kernel Mean Shift with Automatic Scale and Orientation Selection," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1 - 6, 2007

[11] F. Aherne, N. Thacker, and P. Rockett, "The Bhattacharyya Metric as an Absolute Similarity Measure for Frequency Coded Data," in *Kybernetika*, vol. 34, no. 4, pp 363 - 368, 1998.

[12] D. Comaniciu and P. Meer, "Mean Shift: A Robust Approach Toward Feature Space Analysis," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603 - 619, May 2002.

[13] T. Ojala, M. Pietikinen, and D. Harwood, "A comparative study of texture measures with classification based on feature distributions," in *Pattern Recognition*, vol. 29, no. 1, pp. 51 - 59, 1996.

[14] M. Pietikinen, A. Hadid, A. Zhao and T. Ahonen "Local Binary Patterns for Still Images" in *Computer Vision using Local Binary Patterns*, 2011, 2011, XV, 209 p. 87 illus., 56 in color, pp 13 - 43

[15] B. Jhne, H. Scharr, and S. Krkel, "Principles of filter design," in B. Jhne, H. Hauecker, and P. Geiler, editors *Handbook of Computer Vision and Applications*,, vol. 2, pp 125 - 151. Academic Press, 1999.

[16] F. Chaumette, "Image Moments: A General and Useful Set of Features for Visual Servoing," in *IEEE Transactions on Robotics*, vol. 20, no. 4, pp 713 - 723. August 2004

[17] R. Mukundan and K. R. Ramakrishnan, *Moment Functions in Image Analysis: Theory and Applications*, World Scientific, Singapore, 1996.

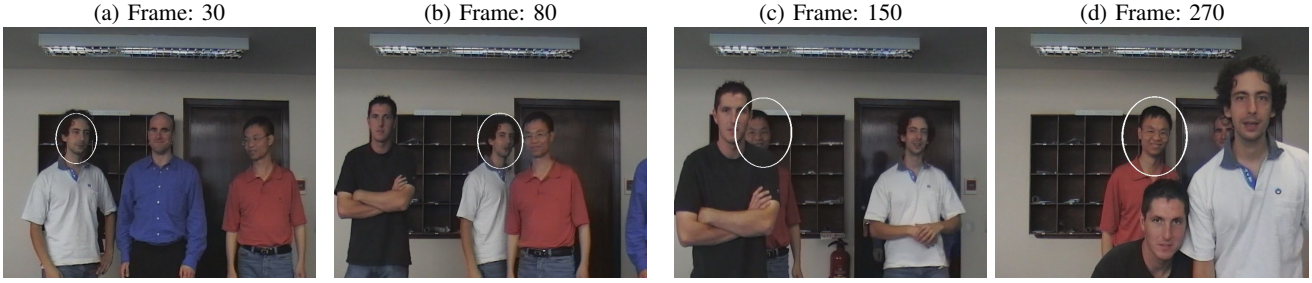
[18] G. Bradski, "Computer Vision Face Tracking for Use in a Perceptual User Interface," in *Intel Technology Journal*, 2(Q2), pp. 1-15, 1998.

[19] G. Welch and G. Bishop, SIGGRAPH 2001, Course 8, Topic: *An Introduction to the Kalman Filter*, University of North Carolina at Chapel Hill, Department of Computer Science, Chapel Hill, NC 27599-3175, 2001

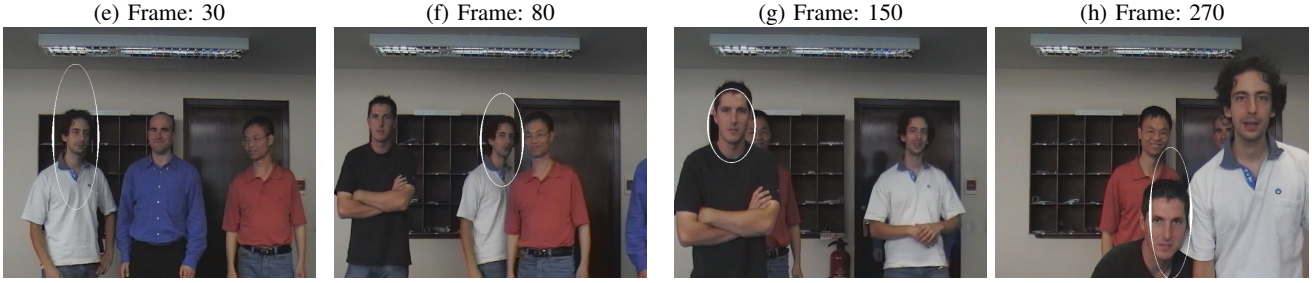
[20] K. Nickels and S. Hutchinson, "Estimating Uncertainty in SSDBased Feature Tracking," in *Image and Vision Computing*, vol. 20, pp. 47-58, 2002.

[21] E. Maggio, E. Piccardo, C. Regazzoni and A. Cavallaro, "Particle PHD filter for multi-target visual tracking," in *IEEE International Conference on Acoustics, Speech and Signal Processing*,(ICASSP 2007), Honolulu (USA), April 15-20, 2007

(a) Frame: 30 (b) Frame: 80 (c) Frame: 150 (d) Frame: 270

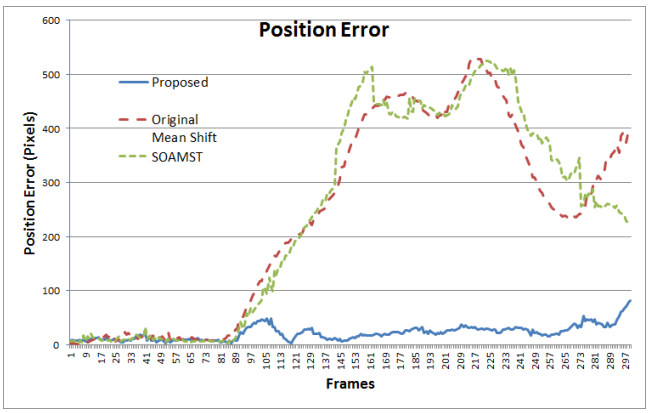


(e) Frame: 30 (f) Frame: 80 (g) Frame: 150 (h) Frame: 270

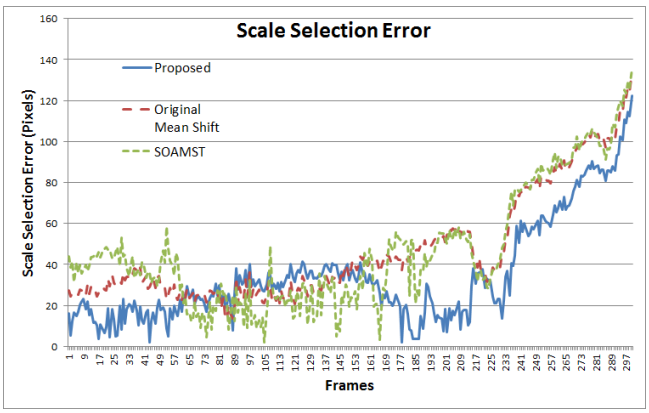


(i) Frame: 30 (j) Frame: 80 (k) Frame: 150 (l) Frame: 270

Fig. 2: Proposed algorothm (a - d), Original Mean Shift algorothm (e - h), SOAMST (i - l)



(a) Position Error from Ground Truth



(b) Scale Selection Error from Ground Truth

Fig. 3: Tracking Error from Ground Truth

A comparison of image features for registering LWIR and visual images

Jaco Cronje
Council for Scientific and Industrial Research, Pretoria, South Africa
Email: jcronje@csir.co.za

Jason de Villiers
Council for Scientific and Industrial Research, Pretoria, South Africa

***Abstract*—This paper presents a comparison of several established and recent image feature-descriptors to register long wave infra-red images in the 8–14 μm band to visual band images. The feature descriptors were chosen to include robust algorithms, SURF and SIFT — and fast algorithms, BRISK and BFROST. To evaluate the feature-descriptors a ground truth was created by determining the intrinsic and extrinsic camera calibration parameters for the cameras and using this to photogrammetrically relate pixel positions between the images. The inlier results of each feature descriptor for the top 20%, 50% and 100% of the matches (based on match strength) were used to create a homography. The average pixel error between the homography reprojected feature points and the photogrammetric reprojection was used as the error. The results show that none of the descriptors perform well in standard form, with BFROST faring slightly better than the other algorithms. This suggests a need to modify the algorithms to detect physical/structural features and de-emphasise textural features.**

I. Introduction

A. Relevance of cross spectral registration

Long Wave Infra Red (LWIR) imagery in the 8–14 μm wavelength band, also known as thermal imagery, has several advantages over visual band imagery [1]. Among these are decreased sensitivity to atmospheric aerosols and scintillation, superior performance in low (visual) light conditions and easy detection of many objects of interest such as vehicles with an internal combustion engine. This is due to the majority of light in this spectrum being emitted by the objects being surveyed rather than being reflected light.

There are several disadvantages to LWIR imagery too. Of particular interest is that intensity of objects in LWIR imagery is solely due to their surface temperature and emissivity, this implies that distinguishing marks such as colour, insignia and serial/licence numbers are generally not visible. In addition, current LWIR cameras typically have significantly lower resolution than visual cameras (e.g. see Sections III-A and III-B) yet cost significantly more. To illustrate these phenomena Figure 1 shows LWIR photos of the authors, it is much more difficult to distinguish between them.

Registering the images of the two bands, that is determining the pixel correspondence between a LWIR and visual image, would allow both the easy determination of objects of interest (using the LWIR band) and their identification (in the visual band). Other benefits may be found such as the haze mitigation of visual images via incorporating a Near Infra-Red (NIR) channel [2].

B. Related Work

Many examples of image feature detector/descriptors have been developed for matching features between visual images. The Geographical Information Systems (GIS) field yields some papers on cross-spectral feature detection. Firmenich *et al.* [3] describe how the Scale-Invariant Feature Transform (SIFT) [4] was modified to perform better in matching between the visual and NIR channels by making it insensitive to reversal in the image gradient. Hasan *et al.* [5] also improved upon SIFT for visual-NIR matching by constraining the portion in the second image on which a match for a feature in the first image is searched. This was done by using two strong matches — which include both spatial and orientation information — to predict where each other feature will be and their scale. Teke and Temezel [6] applied this scale restriction method to the Speeded Up Robust Features (SURF) [7] algorithm. Their results show a worst case matching between the NIR and Blue channels, with results of between 77% and 85% depending on the implementation of SURF and whether or not the scale restriction is applied. Equivalent results for red channels are 86% through 91%.

Brumby *et al.* [8] investigate the supervised evolution of feature extraction kernels by combining primitive image processing operations in order to extract the desired features (such as roads, crop types and rivers) from pre-registered hyperspectral images extending from the visual to short wave infra red (SWIR).

This work is different from that described above in that LWIR is used instead of NIR, a difference of over tenfold in wavelength. This results in a further decrease in feature mapping performance due to the greater dissimilarity between the bands.

C. Axis and notation definition

The mathematical notation used in this paper is as follows: A 3D vector, V_{bac}, is a vector from point a directed towards point b expressed in terms of its projections on orthogonal coordinate system c's axes. V_{bac} is used when the magnitude of the vector is unknown or unimportant. T_{bac} represents the translation or displacement of point b relative to point a.

(a) Author 1 (b) Author 2

Fig. 1. LWIR images of the authors

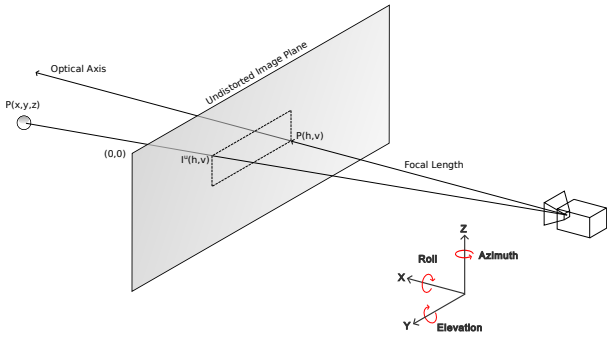


Fig. 2. Axis definition.

R_{ab} is a 3-by-3 Euler rotation matrix expressing the rotation of an orthogonal axis system a relative to (and in terms of its projections on) an orthogonal axis system b. Individual elements of 3 dimensional vectors are referred to as x, y or z whereas 2 dimensional (2D) vector's elements are referred to as horizontal (h) and vertical (v) to avoid confusion. Figure 2 defines the axis system used and the directions of positive rotation.

D. Paper organisation

The rest of this paper is organised as follows: Section II describes the basic workings of the feature detectors. Section III describes the equipment used in this comparison. Section IV details procedure used to objectively compare the different feature metrics. Section V provides the results of the comparison. Section VI summarises the results and places them in context.

II. Feature Descriptor

This section describes the feature detectors used in this comparison. Two floating point and two binary feature point descriptors were evaluated.

A. Scale-Invariant Feature Transform

The SIFT [4] detector searches for stable features across multiple scales by searching for local extrema features over a set of Difference-of-Gaussian (DoG) images. An orientation histogram is constructed by sampling gradient orientations around the feature. The highest peak in the histogram is used as the feature orientation.

The region around the feature is divided into 4 by 4 sample areas. An orientation histogram is calculated for each of the sampling areas. A Gaussian weighting is then applied to the magnitudes before they are accumulated into the histogram. The values of all the histograms are placed into the feature vector. The normalised feature vector forms the 128 floating point value feature descriptor. SIFT is robust to almost all common image transformations.

The match strength between two SIFT features is defined as the L2-Norm: i.e. the length of the difference between the two feature vectors. Smaller values are better.

B. Speeded Up Robust Features

SURF [7] was inspired by SIFT [4], with the main goal to improve the execution speed of the detector and descriptor. SURF depends mainly on an integral image to approximate and speed-up the execution time.

The detector relies on the determinant of the Hessian matrix. The Hessian matrix is approximated by sampling rectangular regions that approximate the Gaussian derivatives. The local extrema from the approximate determinant of the Hessian matrix is located across different scales. Haar wavelets are used to calculate the orientation of sampling points around the feature. The feature orientation is detected by examining the magnitude of the orientations within a sliding arc window. The arc direction with the highest resulting magnitude is chosen as the dominant orientation.

The region surrounding the feature is divided into 4 by 4 sub-regions. Haar wavelet responses for each sub-region are accumulated to form the 64 element floating point feature vector.

The match strength between two SURF features is also defined as the L2-Norm: the length of the difference between the two feature vectors.

C. Binary Robust Invariant Scalable Keypoints

Binary Robust Invariant Scalable Keypoints [9] (BRISK) is a binary feature extractor, the feature detection part uses the

improved version of the Features from Accelerated Segment Test [10] (FAST) detector, namely Adaptive and Generic Accelerated Segment Tests [11] (AGAST) to detect keypoints. The feature detection phase tries to detect features by searching in different scale-spaces. Local image gradients are calculated between sampling point pairs surrounding the feature. The sum of all gradients is used as the feature rotation.

The binary descriptor is built by comparing pairwise, smoothed pixel intensities from sampling points surrounding the feature. Each bit is set when the first pixel intensity is greater than the second pixel intensity. The resulting bits are concatenated to form the 512 bit descriptor.

The match strength between two binary features is defined by the number of elements that differ between the two binary vectors, i.e. the Hamming distance. Smaller values are better.

D. Binary Features from Robust Orientation Segment Tests

Binary features from robust orientation segment tests [12] (BFROST) is a fast feature extractor designed for the Graphics Processing Unit (GPU). BFROST uses the same continuous pixel-set criteria as the FAST detector to detect features with an additional 16 possible feature rotation estimations based on the median of the continuous pixel-set segment.

The feature descriptor describes an area around a detected feature point with a 256 bit binary vector. The descriptor is built by comparing the average pixel intensities of regions surrounding the feature. An integral image is used to speed-up the intensity calculations performed on the sampling pattern.

BFROST is scalable, rotation and translation invariant and robust to noise. The match strength between two features is also defined as the Hamming distance.

III. Equipment

One visual and one LWIR camera, as described below, were rigidly mounted relative to each other. Their intrinsic and extrinsic parameters were then determined (see Section IV-A) to allow for photogrammetric registration.

A. Visual Cameras

Prosilica GT1920 cameras, which have a 3MP resolution of 1936×1456, were used in this work. Pentax lenses with 8mm focal length were used, and provided a field of view (FOV) of $\pm 50^\circ$ horizontally by $\pm 40^\circ$ vertically.

B. Long Wave Infra Red Cameras

Xenics Gobi 640GigE microbolometers were used in this comparison. The cameras have a large 10.88mm by 8.17mm Charge Coupled Device (CCD) offering a resolution of 640×480 pixels. Combined with a 10mm lens, this provided an FOV of $\pm 60^\circ$ horizontally by $\pm 48^\circ$ vertically.

IV. Experimentation Methodology

A. Generating the ground truth

In order to quantifiably compare the different feature descriptors, a ground truth registration was sought. This was obtained by photogrammetrically calibrating the cameras. The lens distortion and inverse distortion was determined as described de Villiers *et al.* [13] using five radial, three tangential parameters and the optimal distortion center. The focal length and the extrinsic parameters of the camera were then determined as per de Villiers [14].

Once these parameters are known, the position that a pixel from Camera B should be placed in Camera A's image is determined by first calculating the the point where the distortion-corrected vector associated with each pixel of Camera B meets the stitching surface (assumed here to be a sphere [14]). This point is then back projected through to Camera A's image plane, where it was redistorted and scaled to determine the pixel position.

In order to calculate the point on the stitching sphere associated with each pixel, one first recalls the cosine rule:

$$a^2 = b^2 + c^2 - 2bc\cos\theta_{bc} \tag{1}$$

where:

$a, b, c =$ the lengths of the side of a triangle, and

$\theta_{bc} =$ the angle between sides b and c.

Now for a pixel i of Camera B, assign the corners of a triangle to be the known center of the sphere in some reference system (i.e. T_{SRR}), the position of camera B expressed in the same reference system (i.e. T_{C_BRR}) and the point where the pixel's vector intersects the sphere. This then infers that side a is equal to the stitch radius (R), and that side b is the distance between the camera and sphere center, or $\|T_{SC_BR}\|$ where $T_{SC_BR} = T_{SRR} - T_{C_BRR}$. All that is required is to determine the vector associated to each pixel and the cosine between it and T_{SC_BR}.

First one creates a vector in Camera B's axis using the focal length and intrinsic distortion parameters:

$$\begin{aligned} I_i^u &= f_B^{undistort}(I_i^d), \\ V_{P_iBB} &= \begin{bmatrix} FLen_B \\ (P_h^B - I_{i_h}^u)pix\_w_B \\ (P_v^B - I_{i_v}^u)pix\_h_B \end{bmatrix}, \\ U_{P_iBB} &= \frac{V_{P_iBB}}{\|V_{P_iBB}\|}, \\ U_{P_iBR} &= R_{BR}U_{P_iBB} \end{aligned} \tag{2}$$

where:

$I_i^d =$ the image coordinate of pixel i,

$f_B^{undistort} =$ the predetermined lens undistortion characterization function [14] for camera B,

$(P_h^B, P_v^B) =$ the principal point of camera B,

$(I_{i_h}^u, I_{i_v}^u) =$ the undistorted pixel position of pixel i,

pix_w_B = the width of the pixels on camera B's CCD,
pix_h_B = the height of the pixels on camera B's CCD,
R_{BR} = rotation of camera B relative to the ref. axis (known from the extrinsic parameters), and
U_{P_iBR} = desired pixel unit vector in reference axis.

Now, recalling that the dot product of two vectors is equal to the product of their magnitudes multiplied by the cosine of the angle between them, Eq. 1 can be rewritten as:

$$R^2 = \|T_{SC_BR}\|^2 + c^2 - 2c \times T_{SC_BR} \bullet U_{P_iBR} \quad (3)$$

which can be rewritten as:

$$0 = c^2 + c(-2 \times T_{SC_BR} \bullet U_{P_iBR}) + \|T_{SC_BR}\|^2 - R^2 \quad (4)$$

This is a quadratic in standard form, and if the camera is inside the stitch sphere will yield a positive and a negative real solution. The positive solution is the desired answer, which yields the point on the stitch radius as

$$T_{iRR} = T_{C_BRR} + c \times U_{P_iBR} \quad (5)$$

Once this point is known it is projected onto camera A's image plane, scaled to the pixel domain and then converted from the undistorted to distorted pixel domains to determine the corresponding pixel from Camera A. This process is exactly the same as that described in Sections III-B through III-D of de Villiers [14].

B. Creating the homography

OpenCV [15] was used to perform the homography calculation using the specified top percentage of the matches. The Random Sample Consensus option was selected to reject outlier matches. The percentage of inlier matches was recorded and used as further indication of the robustness of the homography determined with that particular feature descriptor and match strength.

C. Comparison metric

The metric used is the average error of the inlier features used to create the homography as described in Section IV-B. The error is the distance in pixels between the features in camera B reprojected onto camera A as determined by the homography of Section IV-B and photogrammetric calibration of Section IV-A. This is expressed mathematically as:

$$Error = \frac{1}{N} \sum_{j=0}^{j<N} \left(\|P_j^H - P_j^P\| \right) \quad (6)$$

where:

N = the number of inlier features used,
P_j^H = homography based pixel position of feature j, and
P_j^P = photogrammetrically based coordinate of feature j.

D. Image Scenes

Figure 3 shows the first scene used for this evaluation, it is an urban outdoor scene containing man-made structures with strong edges and texture. Figure 4 shows the outdoor scene used which contains natural vegetation. Both scenes appear, subjectively, to contain rich texture in the visual band.

V. Results

A. Intra-band registration

Table I provides the results of registering between visual images, the values are the number of inliers that agree with the best fit homography. Each scene is registered three times using only the top 20%, 50% or 100% of the matches respectively. The inlier percentage is the percentage of these top matches that were used. Table II provides the same results for registering the LWIR images.

The high percentage of agreement gives confidence on the correctness of the implementations of the four feature-detector algorithms. This is further supported by Figures 5 and 6, which show features correctly being matched within each band. Inlier matches are shown with a green line, while outliers are shown by the blue lines.

The BRISK algorithm performs poorly when 50% or 100% of the matches are used as many of the matches are weak and erroneous. It performs comparably to SIFT and BFROST when only the top matches are used. BFROST performs poorly on the LWIR Urban scene, but is comparable to SIFT in terms of performance when only the top 20% of the matches are used. SURF is consistently worse than SIFT and only marginally better than BRISK.

TABLE I
VISUAL TO VISUAL REGISTRATION INLIER PERCENTAGES

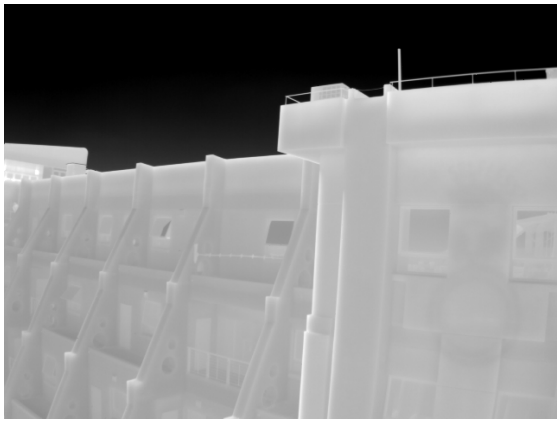
Feature Descriptor	Scene 1			Scene 2		
	20%	50%	100%	20%	50%	100%
SIFT	95.00	99.34	88.15	96.01	99.26	78.86
SURF	65.05	73.60	73.84	94.44	91.26	75.50
BRISK	96.89	85.33	51.75	91.61	71.59	44.12
BFROST	94.11	93.02	86.58	98.92	86.69	65.23

TABLE II
LWIR TO LWIR REGISTRATION INLIER PERCENTAGES

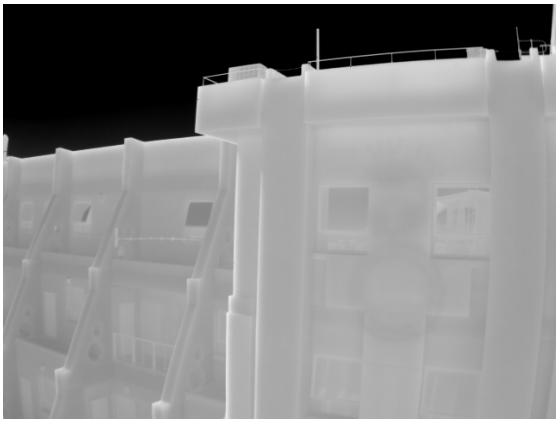
Feature Descriptor	Scene 1			Scene 2		
	20%	50%	100%	20%	50%	100%
SIFT	100.00	100.00	81.48	92.30	94.11	66.17
SURF	75.00	68.47	66.30	87.17	85.71	75.00
BRISK	91.48	81.19	64.25	100.00	69.29	50.78
BFROST	71.42	77.35	81.13	100.00	80.00	70.37

B. Inter-band registration

Table IV provides the results of registering the LWIR images onto the visual images, the values are as per Eq. 6. Table III provides the percentage of inlier features from generating the best fit homography. Figure 7 helps put these numbers

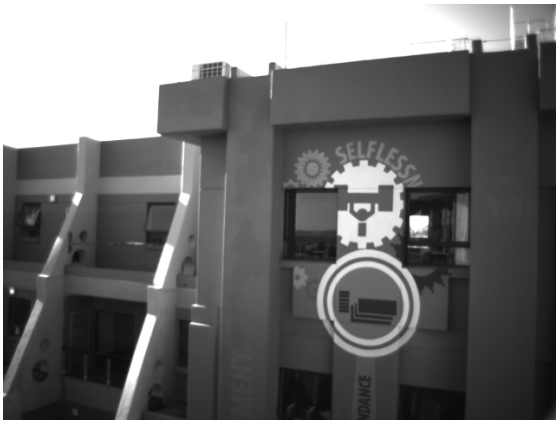


(a) LWIR image 1

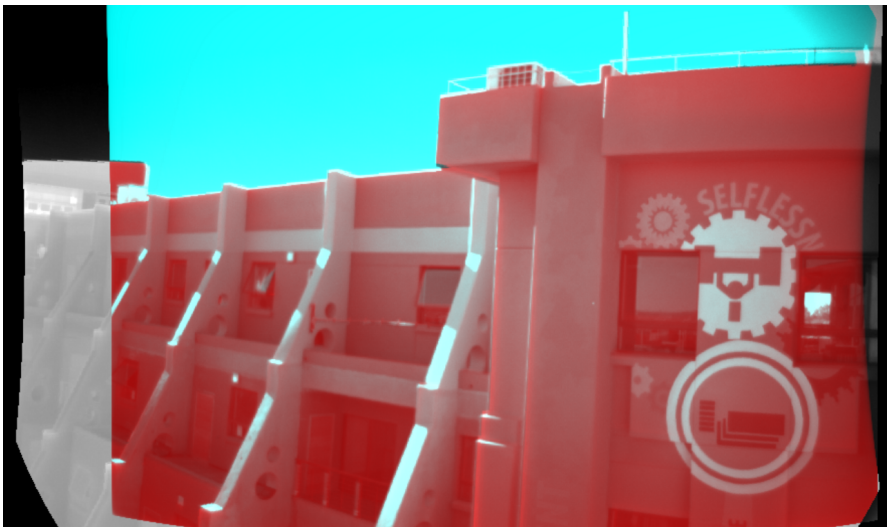


(b) LWIR image 2

(c) Visual image 1

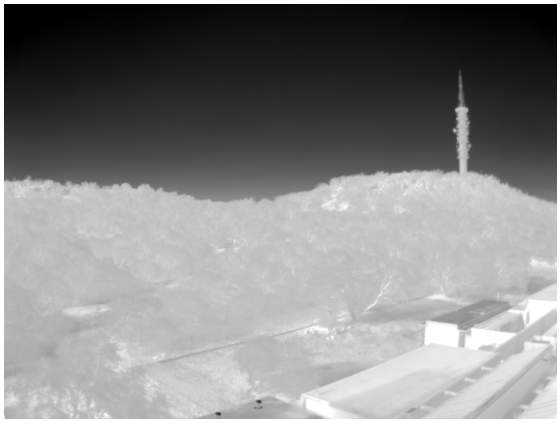


(d) Visual image 2

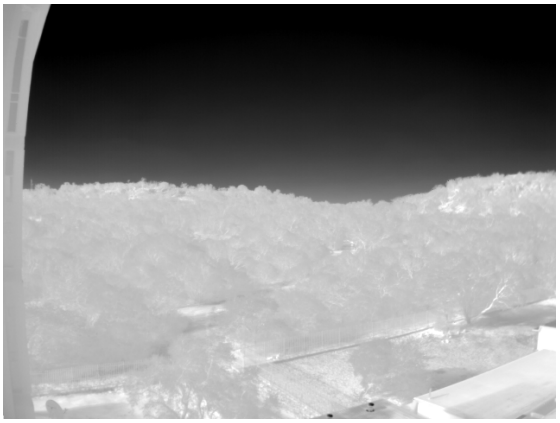


(e) Photogrammetrically stitched image

Fig. 3. Scene 1, Urban landscape



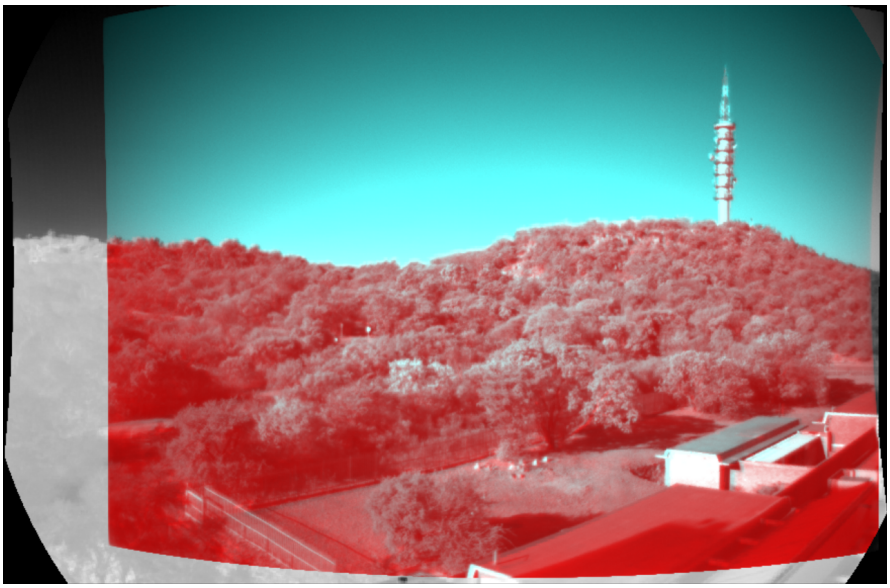
(a) LWIR image 1



(b) LWIR image 2

(c) Visual image 1

(d) Visual image 2



(e) Photogrammetrically stitched image

Fig. 4. Scene 2, Natural landscape

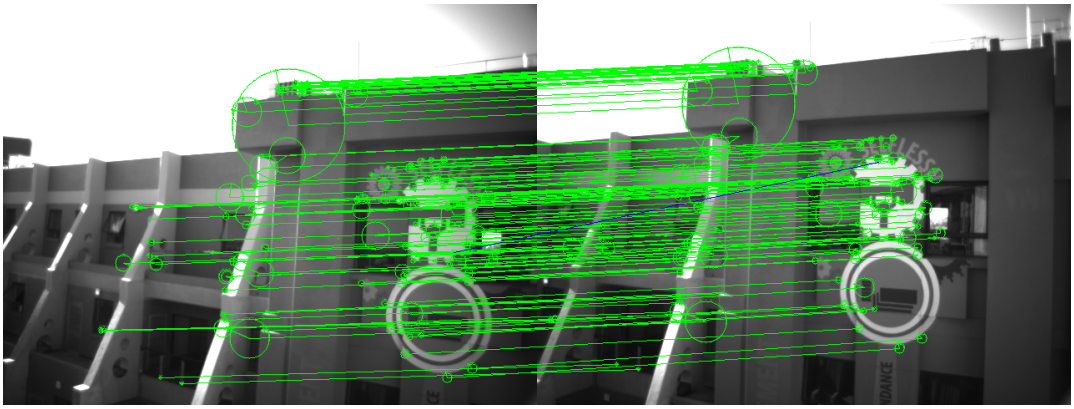


Fig. 5. SIFT feature matches between visual and visual of Scene 1.

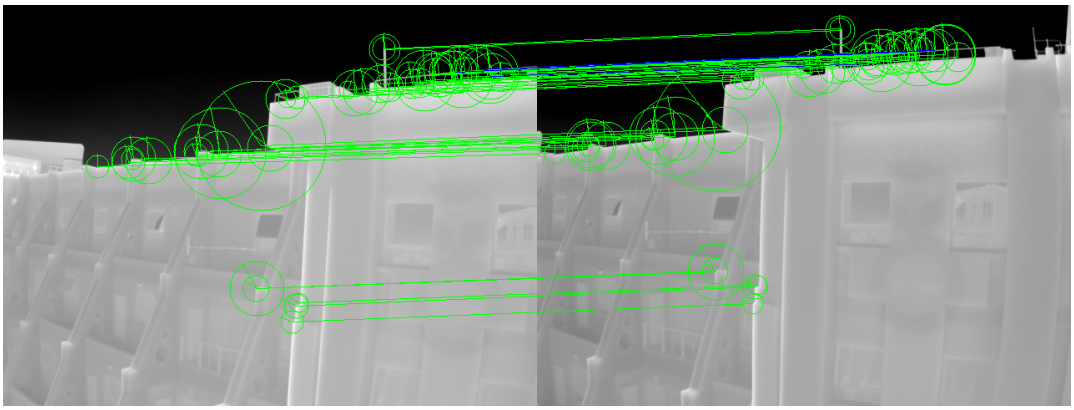


Fig. 6. BRISK feature matches between LWIR and LWIR of Scene 1.

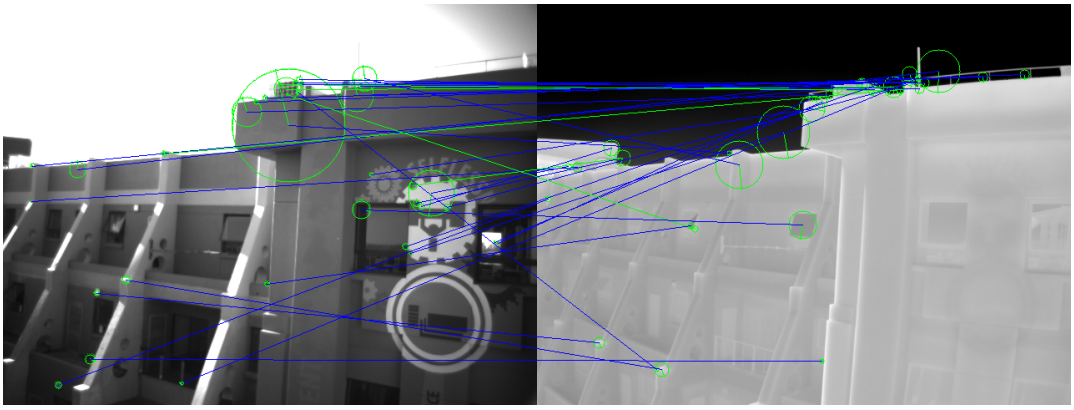


Fig. 7. SIFT feature matches between visual and LWIR of Scene 1.

in context by displaying the features matched between the thermal and visual bands.

Inspection of the values clearly shows that none of the descriptors were able to successfully register the LWIR and visual images. All the algorithms obtained errors of several hundred pixels (expressed in the 3MP AVT Camera's image space) in Table IV, this is further shown by the extremely low agreement percentages in Table III. Often it was not possible for OpenCV to find more than 6 points (the minimum is 4) that agreed to create a consensus homography.

SIFT and BFROST had the greatest number of inliers in the urban and natural scenes respectively, and the second greatest agreement in the other scene. However BFROST had, in almost all the tests, a noticeably lower error than all the other feature descriptors.

A final verification of the correctness of the photogrammetric procedures (in addition to generating Figures 3(e) and 4(e)) was performed. Ten points were crudely selected in each band in each scene, and their equivalent error was calculated. These results are given in the final row of Table IV and confirm the correctness of the photogrammetric procedures. These errors being in the order of 10 pixels, are due to the non precise manual feature selection (which is magnified by the difference in resolutions) and the poor image quality of the Pentax lenses, whose soft focus in the peripheries of the FOV adversely affected the calibrations.

LWIR–visual registration based on canonical features does not perform well. This is due to different keypoints being identified in each band which is compounded by the descriptions of correctly identified matching keypoints frequently being different too.

Further work on feature based matching ma focus on contour alignment and modification of feature descriptors to better cater for cross band matching.

VI. Conclusions

This paper tested four popular feature descriptors for the purpose of registering LWIR and visual imagery. The feature descriptors were used in unmodified canonical form6 to facilitate the selection of which descriptor should modified for LWIR-visual registration. In addition to the standard calculation of number of inlier matches, a quantified error based on comparison to photogrammetric calibration and stitching was performed.

It was found that none of the algorithms were able to register across the two bands, although all the algorithms registered well within either of the bands. This finding is consistent with Firmenich *et al.* [3] who speculated that a new feature extractor may need to be developed for LWIR imagery registration.

SIFT and BFROST significantly outperformed SURF and BRISK for inter band registration. BFROST was significantly quicker than SIFT, and so it is recommended for future modification for LWIR-visual registration.

VII. Acknowledgements

This work was supported by the Armaments Corporation of South Africa.

TABLE III
LWIR to visual registration inlier percentages

Feature Descriptor	Scene 1			Scene 2		
	20%	50%	100%	20%	50%	100%
SIFT	57.14	27.77	16.21	33.33	15.21	7.60
SURF	23.07	12.30	5.38	12.00	4.80	1.99
BRISK	28.57	17.75	9.81	25.80	23.22	12.25
BFROST	44.44	22.72	13.33	37.50	17.07	8.43

TABLE IV
LWIR to visual registration errors

Feature Descriptor	Scene 1			Scene 2		
	20%	50%	100%	20%	50%	100%
SIFT	926.6	488.6	638.2	604.8	812.2	611.4
SURF	743.1	599.1	798.7	471.8	810.1	574.2
BRISK	888.8	765.3	781.9	531.3	763.4	741.0
BFROST	684.6	438.0	620.7	929.2	371.3	441.0
Manual	11.0			11.0		

References

[1] L. Biberman, *Electro-Optical Imaging: System Performance and Modeling*. SPIE Press, 2000.

[2] L. Schaul, C. Fredembach, and S. Susstrunk, "Color image dehazing using the near-infrared," in *Image Processing (ICIP), 2009 16th IEEE International Conference on*, nov. 2009, pp. 1629 –1632.

[3] D. Firmenich, M. Brown, and S. Ssstrunk, "Multispectral interest points for rgb-nir image registration." in *ICIP*, 2011, pp. 181–184.

[4] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[5] M. Hasan, X. Jia, A. Robles-Kelly, J. Zhou, and M. Pickering, "Multispectral remote sensing image registration via spatial relationship analysis on sift keypoints," in *Geoscience and Remote Sensing Symposium (IGARSS), 2010 IEEE International*, july 2010, pp. 1011 –1014.

[6] M. Teke and A. Temizel, "Multi-spectral satellite image registration using scale-restricted surf," in *Pattern Recognition (ICPR), 2010 20th International Conference on*, 2010.

[7] H. Bay, T. Tuytelaars, and L. V. Gool, "SURF: Speeded up robust features," in *In ECCV*, 2006, pp. 404–417.

[8] S. Brumby, P. Pope, A. Galbraith, and J. Szyinanski, "Evolving feature extraction algorithms for hyperspectral and fused imagery," in *Information Fusion, 2002. Proceedings of the Fifth International Conference on*, vol. 2, 2002, pp. 986–993.

[9] S. Leutenegger, M. Chli, and R. Y. Siegwart, "BRISK: Binary robust invariant scalable keypoints," in *International Conference on Computer Vision 2011 - ICCV2011*, 2011.

[10] E. Rosten, R. Porter, and T. Drummond, "Faster and better: A machine learning approach to corner detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 32, no. 1, pp. 105–119, 2010.

[11] E. Mair, G. Hager, D. Burschka, M. Suppa, and G. Hirzinger, "Adaptive and generic corner detection based on the accelerated segment test," *Computer Vision–ECCV 2010*, pp. 183–196, 2010.

[12] J. Cronje, "BFROST: binary features from robust orientation segment tests accelerated on the GPU," in *Proceedings of the 22nd Annual Symposium of the Pattern Recognition Society of South Africa*, ser. PRASA2011, 2011.

[13] J. P. de Villiers, F. W. Leuschner, and R. Geldenhuys, "Centi-pixel accurate real-time inverse distortion correction," in *Proceedings of the 2008 International Symposium on Optomechatronic Technologies*, ser. ISOT2008, vol. 7266, 2008, pp. 1–8.

[14] J. P. de Villiers, "Real-time stitching of high resolution video on COTS hardware," in *Proceedings of the 2009 International Symposium on Optomechatronic Technologies*, ser. ISOT2009, vol. 9, 2009, pp. 46–51.

[15] G. Bradski, "The opencv library," *Dr. Dobb's Journal of Software Tools*, 2000.

FIGURE DETECTION AND PART LABEL EXTRACTION FROM PATENT DRAWING IMAGES

Jaco Cronje

Council for Scientific and Industrial Research, Pretoria, South Africa
Email: jcronje@csir.co.za

ABSTRACT

The US Patent and Trademark Office, together with the NASA Tournament Lab, launched a contest to develop specialized algorithms to help bring the seven million patents presently in the patent archive into the digital age. The contest was hosted by TopCoder.com, the largest competitive online software developer community. The challenge was to detect, segment and recognize figures, captions and part labels from patent drawing images. The solution presented in this work was the winning submission.

Index Terms— Image analysis, Character recognition, Image segmentation, Document image analysis

1. INTRODUCTION

Around seven million patents are presently stored in the US Patent and Trademark Office (USPTO) patent archive. Many of these patents are originally created before the digital age. Images of the scanned versions of these old dated patents are stored in the patent archive. These documents contain descriptive information as well as drawings about the patent. Most of the drawings are mechanical drawings which contain a lot of parts. Each part is labeled such that it can be referenced from the text description. The figures also contain captions that are used to identify and reference each specific figure.

The USPTO, together with the Harvard-NASA Tournament Lab launched an innovation challenge to invite developers and academics to develop specialized algorithms to detect and label figures and parts from the USPTO patent archive. The evaluation and submission interface to the challenge were hosted by TopCoder.com. TopCoder [1] hosts the world's largest competitive community for software developers and digital creators with a community of over 380,000 members around the world. Up to $50,000 of prizes were distributed to contest winners. The challenge ran for four weeks from mid December 2011 to mid January 2012.

Harvard University concurrently ran a research project about a study on how competitors work together within such contests. All registered competitors were divided into teams of two. The protocol used to match competitors to form teams is described in [2]. Each week during the contest, competitors had to complete a survey about their progress and their teammates progress. The strategic behavior of TopCoder contestants has been analyzed in [3].

Section 2 describes the problem statement. The algorithm evaluation method, implementation restrictions and limitations are described. Related work is reviewed in section 3. The method used by the author to solve the problem is presented in section 4. Section 5 provides some results produced by the proposed method. Finally section 6 concludes the article.

2. PROBLEM STATEMENT

The problem is to extract useful information from patent drawing pages. Each patent drawing page contains one or more figures. There can also be additional data that do not belong to any of the figures. Each figure has a caption and consists of many parts. Each part is labeled with text (typically a number). Some parts may have multiple labels. The task is to extract the location and caption for each figure and to extract the location and text for each part label.

Figure 1 illustrates the useful information of a patent drawing page for the challenge. It contains 3 figures namely $2A$, $2B$ and $2C$. Each figure has 14, 8 and 8 part labels, a total of 30 part labels for the whole drawing page. The figures are indicated by the blue polygons and the part labels by the red polygons.

The input to the algorithm consists of a raw input image and the patent text data if available for the particular patent. Patent text pages contain text that describes the patent and their drawings, the text usually contain references to figures and part labels. The ground truth of a set of 306 patent drawing pages were created for the purpose of evaluating the algorithms. 178 of these drawing pages were provided as a training set. 35 drawing pages were used for preliminary online testing. The remaining 93 drawing pages were used for the final evaluation to determine the prize winning solutions.

The output of an algorithm is evaluated against the ground truth data. The score for each drawing page is determined by the correctness (S_{corr}) and performance score (S_{perf}). The performance score is based on the run-time (T in seconds)

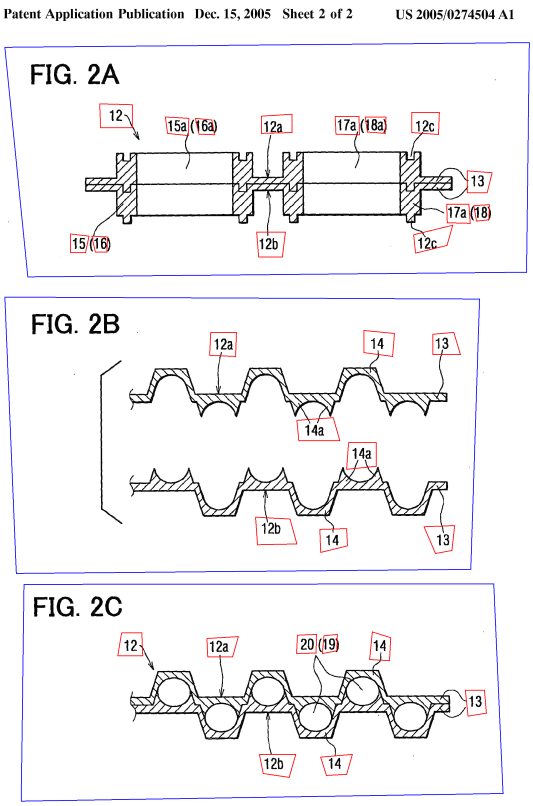


Fig. 1. Example of a patent drawing page with ground truth data. Figures are marked with blue and part labels marked with red polygons.

of the algorithm and calculated by equation 1. No penalty is applied if the run-time is less than a second, but anything slower than that can result up to a 10% penalty.

$$S_{perf} = 0.9 + 0.1 * (\frac{1}{max(T, 1)})^{0.75} \quad (1)$$

The correctness score is calculated by finding the intersection between the bounding boxes of the ground truth data and the algorithms output. For each correctly matched intersection the intersection score is incremented with 0.25 and incremented with another 0.75 if the text for the label or part matches. The intersection score is then used to calculated the precision and recall measurements, which are combined by the harmonic mean 2 to form the final correctness score for the given patent drawing page.

$$S_{corr} = \frac{2 * precision * recall}{precision + recall} \quad (2)$$

The score for an individual test case is given by 3. The overall score is then the sum of scores over all the individual test cases.

$$Score = 1000000 * S_{corr} * S_{perf} \quad (3)$$

Competitors were allowed to program in C++, C#, Visual Basic, Java or Python. The source code size limit was set to 1 MB. No access to external files were allowed. The time limit for each test case was 1 minute and the memory limit 1024 MB.

3. RELATED WORK

An overview of the benefits, requirements and challenges involved in the development of a patent image retrieval framework is provided in [4]. Furthermore, a patent search engine called PatMedia was developed based on the proposed framework. The framework segments the patent drawings into figures, extract their captions and perform feature extraction on each detected figure. The extracted figure features are used to index patent drawings and to search for similar drawings within the patent database. Information extracted from the associated patent text pages are merged with the image based information to improve the performance and resolve ambiguities.

The PATSEEK [5] application is a content-based image retrieval search engine for the US patent database. Just like PatMedia [4], PATSEEK [5] detects the figures from patent drawings and extracts a feature vector for each figure to be used for retrieval purposes. Both of them use slightly different techniques. PATSEEK do not make use of the information in the patent text pages and is outperformed by PatMedia.

The work presented in [6] focus on the extraction of features from patent or technical drawings for retrieval purposes. Lines and their attributes are detected from the drawings. The set of lines is transformed into a nearest neighbor graph and the graph attributes are converted into a 2-Dimensional histogram for fast image comparisons.

The use of angular and radial distribution information for figure feature description was used in [7]. The work in [7] focused thus more on 2-Dimensional shape features in patent drawings.

A method to detect alphanumeric labels from figures is described in [8]. The work doesn't focus specifically on patent drawings, but focuses on documents that contain a mixture of text and figures.

Captions and part labels are extracted from patents in [9] to create a user friendly browser interface. Their approach used an unsupervised clustering algorithm to classify connected components as characters or not. It is assumed that the font used across multiple drawings of the same patent remains the same. The same authors presented a patent drawing segmentation algorithm in [10]. The segmentation algorithm performs Delaunay triangulation to segment the drawing into

a graph. The graph is then further reduced and segmented such that document layout constraints are not violated.

The method presented in this article use similar techniques used in [4], [8] and [9] to extract figure captions and part labels. PatMedia used a commercial Optical Character Recognition (OCR) library where as it was not allowed for the USPTO challenge.

4. METHOD

The method presented in this work was the top submission for the USPTO innovation challenge. Patent drawings usually consist of a header, a typical header can be seen at the top of the drawing page in figure 1. The figures on the drawing may be orientated horizontally or vertically. Section 4.1 describes how the page orientation is detected.

Firstly a margin around the border of the image is cleared to eliminate the header from further image processing steps. The gray scale image is then converted to a binary image by applying a fixed threshold.

Many old patent images contain a lot of salt and pepper noise. A connected component algorithm is performed and if the number of very small components detected are more than 30% of the total number of components, a dilate and erode process are performed to reduce the noise.

4.1. Page orientation

In order to recognize the text from captions and part labels, the orientation of the page needs to be detected. All the connected components that could possibly be a character are used to determine the page orientation. Figure 2 illustrates a patent drawing which is vertically orientated along with its detected connected components.

A voting system classifies the page to be horizontal or vertical. For each character, a vote is cast for a horizontal layout if the width of the character is greater than the height, otherwise a vertical vote is counted. Also, for each character the nearest neighboring character is found. A vote is then cast depending on whether the two characters are more horizontally or vertically aligned to each other.

The dominant orientation with the most votes wins.

4.2. Text extraction

The image is segmented through connected component labeling. Each connected component can be a character, part of a figure or image noise. Each connected component needs to be classified into one of the categories before the figure extraction and part labeling process can proceed.

Components with a width and height smaller than 13 or greater than 250 pixels are regarded as not characters. The resolution of the images were typically 2560 by 3300 pixels. The remaining components are marked as possible characters

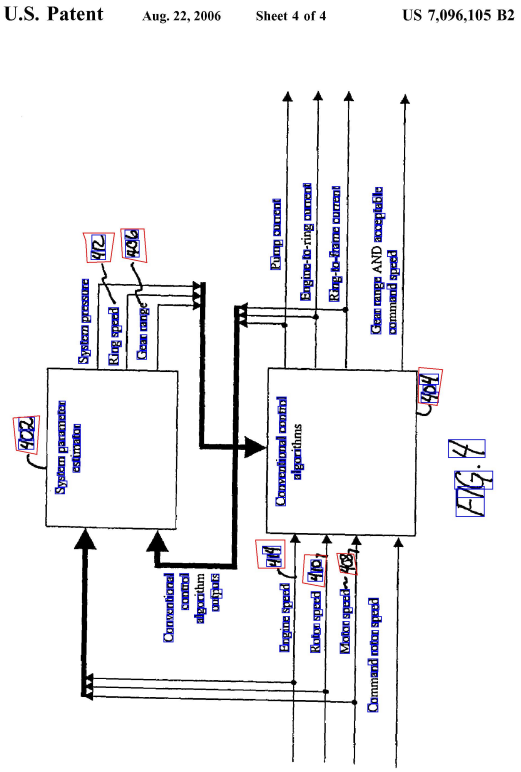


Fig. 2. Vertical page orientation. The connected components that could be characters are indicated with blue rectangles.

if they do not contain any other component within the characters axis aligned bounding box.

Components marked as characters are then sorted from left to right. Groups of character components are created based on the same merging metric described in [8]. The metric merges two components if their horizontal spacing is small and they overlap significantly in the vertical direction.

The group of characters are then recognized. Each character is separately processed by the character recognition system explained in section 4.3.

4.3. Character recognition

A simplistic template matching algorithm is used to perform optical character recognition. Patches containing known characters were manually extracted from the set of training images. Only the ten numerical characters and the characters f, g, a and c were used as templates. The characters f and g had to be recognized to detect the figure captions. The characters a and c mostly appear at the end of part labels and within figure captions. The character b was not recognized because of

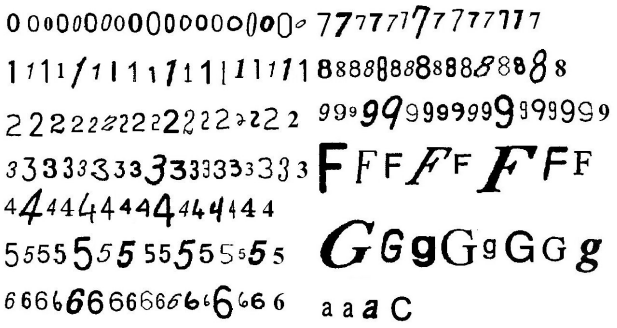


Fig. 3. Characters manually extracted from the set of training images.

the similarities between b and 6. Figure 3 shows the template patches.

The connected component under recognition is firstly scaled to fit an area of 16 by 32 pixels. All the pixels that belong to a hole in the character are marked by using a flood fill algorithm. The scaled image and hole information are compressed into 32 unsigned integers to form the component descriptor.

To find the best matching character, each template is compared with the input component descriptor. The number of matching pixels P and mismatched pixels F are counted. A matching score is calculated by $(P - F)/(P + F)$ and the best scoring template is used as the recognized character.

4.4. Figure extraction

The bounding box and caption of each figure within the drawing need to be extracted and recognized. Firstly the components are extracted as described in section 4.2. Text components that contain the pattern $f1g$ are removed from the component list and added in a list of possible detected figure captions.

A different method is used to segment the figures when no figure caption was detected. Components with an area less than 300^2 pixels are merged with their nearest neighboring component. Larger components are merged only with their intersecting components. Merging two components mean that their axis aligned bounding boxes are merged into one bounding box that contains both of the original bounding boxes. The merging process continues until no more components are merged. Figure 4 shows the components before the merging process.

Each component is initially assigned to their nearest figure caption if captions were detected. Figure 5 shows the components after the merging process. Note that the three components below figure 2B should all be assigned to figure 2B and a simple nearest neighbor assignment will not work in this case and needs to be refined. A segmentation score is calculated by taking into account the bounding box intersecting

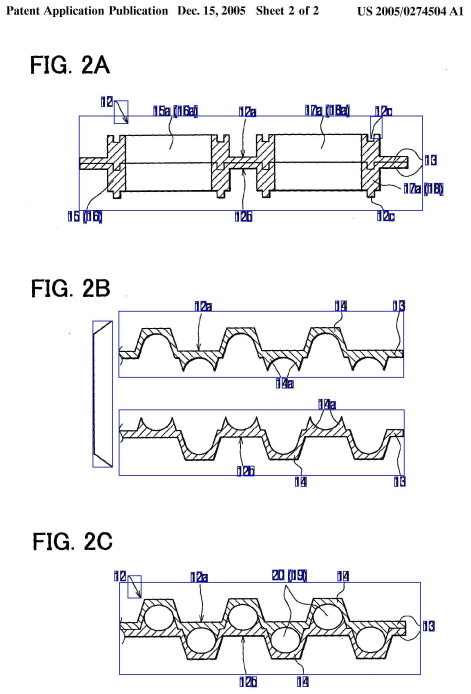


Fig. 4. Components before the merging process begins.

area of the segmented figures. The score is penalized when none or more than one figure caption intersects the bounding box assigned to a figure. The components assignment are randomly shuffled for 1000 iterations and the best scoring segmentation is used.

The header of a patent usually contains text that indicates the current sheet number and the total number of sheets. These sheet numbers are extracted and used to refine the recognized figure captions.

Possible figure captions are extracted from the patent text data and sorted numerically. The recognized figure captions are matched with the captions from the text. The best matching sequence is used for the figure captions in the drawing, taking into account the sheet numbers. For example the last sheet should contain the last figures.

The bounding boxes returned in the output are shrunk such that they minimize their intersection with each other.

4.5. Part labeling

The part labeling process firstly extracts text components described in section 4.2. Patent drawings can contain tables or graphs, usually they do not contain any part label inside their boundaries. The border of each component is examined. If the border is more than 25% filled, the component is considered to be a table or a graph and all the intersecting text

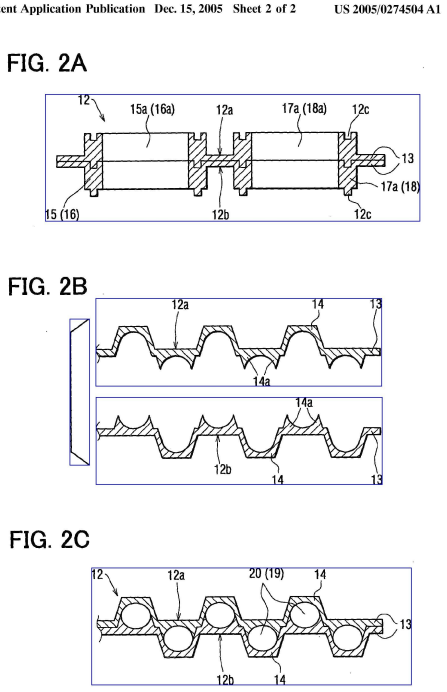


Fig. 5. Components after the merging process.

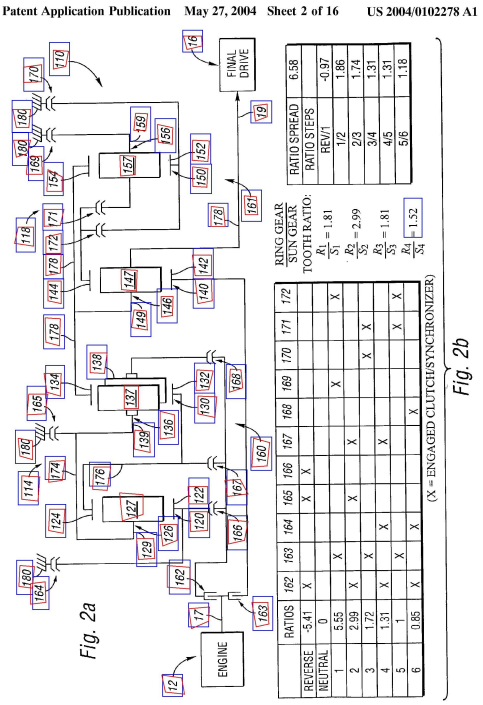


Fig. 6. Patent drawing that contains a table. Red rectangles show the ground truth data and blue rectangles show the detected part labels by the algorithm.

components are removed. Figure 6 shows a patent drawing that contains a table.

Text components containing one of the following characteristics do not classify as part labels:

- The width or height is smaller than 10 pixels.
- The component contains more that 4 characters.
- Figure captions are removed.
- Character recognition matching score below zero.
- No numbers occur within the text.
- The text contains more than one alphabetic character.
- The border surrounding the text is more than 4% filled.

Words that contain numbers are extracted from the patent text data. The recognized text from the remaining text components are corrected by finding the best matched word from the patent text data. The correction only takes place if the character recognition matching score is below 0.5. The text component is removed if the best match from the patent text changed more than half of the original recognized text.

Finally the average height and area of the remaining text components are computed. Any text component where the height or area of which differs significantly from the average is removed from the output. The bounding boxes of the parts are shrunk such that they minimize their intersection with each other.

Table 1. *Training set performance.*

	Correct	Total	Percentage
Figures detected	234	285	82.1
Captions recognized	213	234	91.0
Part labels detected	2875	3752	76.6
Labels recognized	2424	2875	84.3

5. RESULTS

Table 1 shows the performance on the training set. The percentage of correctly segmented figures, recognized captions, part label locations detected and part label text recognized are shown. The running time of the algorithm was below 1 second for all cases, thus avoiding any time penalty. The average recall and precision measurements on the training set is shown in Table 2.

The overall score was 275 million out of a possible 356 million based on the USPTO challenge scoring metric on the training set.

Table 2. *Recall and precision measurements.*

	Recall	Precision
Figures	0.8534	0.8537
Part labels	0.7533	0.7358

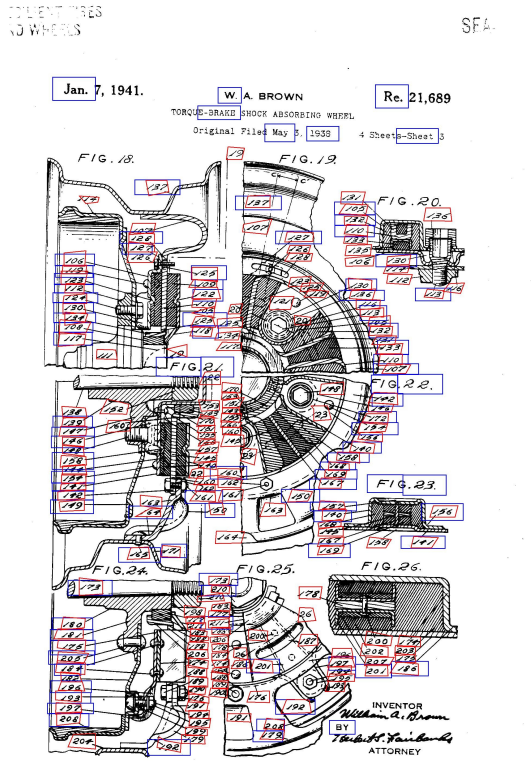


Fig. 7. Patent drawing that contains hand written characters and figures that are difficult to label.

6. CONCLUSION

The work presented in this paper provides a way to segment and label figures from patent drawing pages. A method for part label extraction has been described. The algorithm was tested on a set of real patent drawings and the results look promising as the algorithm scored at the top within the challenge.

There is still room for improvements to the algorithm due to the limited duration of the USPTO innovation challenge. A more sophisticated character recognizer could be integrated. Figure 7 shows a drawing with hand written characters and figures that are difficult to segment.

The USPTO challenge[1] was an interesting challenge and drawn the attention of many top problem solvers around the world. Hopefully more challenges will be launched in the future to promote and encourage academics and developers to solve real world problems together on a global scale.

[1] `http://community.topcoder.com/longcontest/stats/?module=ViewOverview&rd=15027`

7. REFERENCES

[1] KR Lakhani, D. Garvin, and E. Lonstein, "Topcoder (a): Developing software through crowdsourcing," *HBS Case*, pp. 610–032, 2010.

[2] J.J.M. Tan, "A necessary and sufficient condition for the existence of a complete stable matching," *Journal of Algorithms*, vol. 12, no. 1, pp. 154–178, 1991.

[3] N. Archak, "Money, glory and cheap talk: analyzing strategic behavior of contestants in simultaneous crowdsourcing contests on TopCoder.com," in *Proceedings of the 19th international conference on World wide web*. ACM, 2010, pp. 21–30.

[4] S. Vrochidis, S. Papadopoulos, A. Moumtzidou, P. Sidiropoulos, E. Pianta, and I. Kompatsiaris, "Towards content-based patent image retrieval: A framework perspective," *World Patent Information*, vol. 32, no. 2, pp. 94–106, 2010.

[5] A. Tiwari and V. Bansal, "PATSEEK: content based image retrieval system for patent database," in *Proceedings of International Conference on Electronic Business, Beijing, China*, 2004, pp. 1167–1171.

[6] B. Huet, N.J. Kern, G. Guarascio, and B. Merialdo, "Relational skeletons for retrieval in patent drawings," in *Image Processing, 2001. Proceedings. 2001 International Conference on*. IEEE, 2001, vol. 2, pp. 737–740.

[7] Z. Zhiyuan, Z. Juan, and X. Bin, "An outward-appearance patent-image retrieval approach based on the contour-description matrix," in *Frontier of Computer Science and Technology, 2007. FCST 2007. Japan-China Joint Workshop on*. IEEE, 2007, pp. 86–89.

[8] M. Worring and A.W.M. Smeulders, "Content based hypertext creation in text/figure databases," *Series on software engineering and knowledge engineering*, vol. 8, pp. 87–96, 1997.

[9] L. Li and C.L. Tam, "A graphics image processing system," in *Document Analysis Systems, 2008. DAS'08. The Eighth IAPR International Workshop on*. IEEE, 2008, pp. 455–462.

[10] L. Li and C.L. Tan, "Associating figures with descriptions for patent documents," in *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems*. ACM, 2010, pp. 385–392.