Grasping Objects from a User's Hand using Time-of-Flight Camera Data

Natasha Govender and Jonathan Claassens

Abstract— **For a robotic platform to be able to assist/interact in human environments, the platform must be able to perform some fundamental tasks. This includes interacting with humans by grasping or releasing objects as or when required by the human. This paper presents a system which allows a robotic arm manipulator to grasp any moving object from a user's hand and releases the object when indicated to do so. Data from a Time-of-Flight camera is fused with an ordinary laboratory camera to create a robust method of rapidly tracking a target object and providing data of possible obstacles. A basic experiment is used to illustrate to the system.**

I. INTRODUCTION

For a robot manipulator to collaborate with a user in completing an arbitrary task, it is necessary for both participants to pass objects to each other. From the perspective of the robot, taking a provided object involves tracking, grasp planning and 'safe' execution. The latter is not a focus of this paper.

There has been some research conducted on this topic [3] and there is a massive body of knowledge on grasp planning and object tracking [1]. Our emphasis is on leveraging the Time-of-Flight (ToF) camera data and fusing it with a standard laboratory camera to provide a means to rapidly track a target object and provide data of potential obstacles. The aim is to improve the robustness of a 'taking' maneuver. This is an important step in creating a robot solution where a user requires a reliable, quick responding robot in a collaborative task. The paper represents an initial solution at solving the problem with a vision system. The kinematic and planning aspects are kept simple.

The ToF camera emits an infrared pulse and measures return phase change at every pixel to estimate depth over an image. We used a Mesa Imaging SR4000 which, if conditions are right, provides impressively accurate point cloud data with associated intensities. The resolution of 176x144 is low, but if the point cloud data can be calibrated to data from a ordinary lab camera an excellent depth map estimate of a scene can be measured. The ToF camera provides a frame rate of roughly 30 frames per second (fps) and combined with a fast implementation of SIFT, the system can be used to locate and track a moving object which the robot is required to take.

This paper will describe all the components required to complete the 'taking' maneuver. The structure of the paper is as follows: Section II will describe the system architecture and provide details of the sensor fusion, robot planning and object recognition. Section III will present the experimental results and finally Section IV will describe the conclusions and future work.

Natasha Govender and Jonathan Claassens are with the Mobile Intelligent Autonomous Systems, CSIR Pretoria, RSA jclaassens@csir.co.za

II. SYSTEM

The equipment used in the proposed system consists of a Barrett Whole Arm Manipulator^{tm} (WAM), a ToF camera and a Point Grey black and white lab camera. These two camera are mounted fixed to each other. Our WAM has a Barrett Hand end effector, a variant of the Salsbury hand, with sufficient prehensile capability to grasp larger, less delicate day to day objects. The lab camera is used to recognize the target object. The approach used will be presented in Subsection B. To obtain the 3D pose of the object, the ToF data is fused with the vision information. The fusion process is described in the next section.

A. Time-of-Flight Range Data

The ToF camera's software provides a 3D point cloud with the origin set just in front of the lens. To label pixels in the lab camera's image plane the following method is used. Pictures are taken simultaneously with both cameras of a chessboard. In the ToF case, the camera's infrared return image is used. This image has the modest resolution of 176x144, but is still sufficiently high to calibrate the camera using OpenCV.

The camera center of the ToF's infrared image is not at the origin of the 3D point cloud. This would thwart any attempt to calibrate the two images directly with something like OpenCV's stereo calibration algorithm. To correct for this the intensity labeled ToF data is reprojected to a virtual camera which shares an origin with the point cloud. An arbitrary focal length of 260 is chosen. The method of K-Nearest Neighbors (KNN) is used to interpolate the pixels values across the 176x144 resolution virtual camera.

After determining the intrinsic parameters of the lab camera, OpenCv's stereo callibration algorithm was applied to the virtual camera and Point Grey camera images to determine the cameras' extrinsic parameters. Finally depth-labeled point cloud data is projected onto the Point Grey camera image plane. Because the ToF camera may see points behind objects in the lab camera image, it is necessary to remove points that are occluded from the perspective of the lab camera. This is done by dividing the lab camera image into cells (10x10 in the experiments) and assigning all pixels within a cell with the minimum depth of a point from the ToF camera that project into the cell.

Fig. 1. SIFT features detected and extracted from an object used in the training set

The ToF camera has a number of sources of error **??**. There is a smoothing effect on the boundaries of objects so that the discontinuity of an occlusion appears as a gradual change. Color and illumination dependent noise sum with typical white noise to further roughen the measurement. In the proposed system, a simple averaging filter was used to the remove the white noise. Any point that measured a variance higher than some threshold over three frames was removed.

B. Object Recognition

For each object in the dataset, training images are captured. These are used to create a model for each object, which enables the system to recognize the object at some later stage. For the training set, objects are spun in front of the system's cameras against a white background. Approximately 40 images were captured for each object. The Scale Invariant Feature Transform (SIFT)[4] detector and descriptor was used to extract relevant features from all the training images captured. SIFT was sued as it is robust to changes in illumination and affine transformations. Figure 1 is an example of SIFT features which were detected and extracted from an object in the training set.

Most state-of-the-art object recognition systems combine multiple training images to produce a single model representation of an object. This has the advantage of allowing the system to recognise an object from any viewpoint, especially if the object is in a cluttered environment or partly occluded. We used Davide Lowe's [5] view clustering algorithm to combine multiple training views to create 2.5D models of each object. The idea is that similar image views of an object are clustered into a single model view. The 2.5D representation of the object consists of a set of these model views which represents views from a range of significantly different locations around the view sphere of the object.

The first training image is used to build an initial model. This consists of the all SIFT features extracted from the training view, as well the location, orientation and scale of each feature in that image. We then use SIFT matching followed by the Hough transform [7] and a least-squares geometric verification to match subsequent images.

The matches obtained using SIFT matching are inputted into the Hough transform. The Hough transform assists in removing ambiguous SIFT matches. This is achieved by allowing each match to vote for an approximate location, scale and orientation of the object as described in [6]. Only features that agree on a specific object location, scale and orientation are kept. We use a bin size of 30 degrees for orientation, a factor of 2 for scale and 0.25 times the maximum model dimension for location. The large bin sizes allow us to cluster images even in the presence of substantial geometric distortion. Figure 2 displays features that have voted for a particular location,scale and orientation of an object.

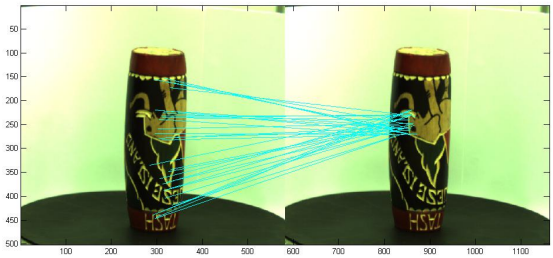

Fig. 2. Features that agree on a particular object location,scale and orientation

Only bins with at least 3 entries are considered. These are then sorte into decreasing order of size. A geometric verification using a similarity transform is then performed on each bin. This enables us to calculate if there exist parameters that allow the similarity transform from the reference image to the matched point.

The similarity transform gives the mapping of a model point *[x y]* to an image point *[u v]* in terms of image scaling s, image rotation,θ, and image translation $[t_x, t_y]$.

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} s & cos\theta & -s & sin\theta \\ s & sin\theta & s & cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

Defining $m = s\ cos\ \theta$ and $n = s\ sin\ \theta$

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} m & -n \\ n & m \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

The above equation can then be written in a linear form collecting the unknown similarity transform parameters into a vector[7].

$$\begin{bmatrix} x & -y & 1 & 0 \\ y & x & 0 & 1 \\ \cdots & & & \end{bmatrix} \begin{bmatrix} m \\ n \\ t_x \\ t_y \end{bmatrix} = \begin{bmatrix} u \\ v \\ \vdots \end{bmatrix}$$

This equation describes a single feature match, but any number of further matches can be added, with each contributing two more rows to the first and last matrix. We can write this linear system as:

$A_x = b$

The least-squares solution for the parameters x can be determined by solving the corresponding normal equations,

$X = [A^T A]^{-1} A^T b$

which minimises the sum of square distances from the projected model locations to the corresponding image locations. We can then use this solution to calculate the error *e* between the projected model feature and the image feature.

$e = \sqrt{\frac{2\|Ax-b\|^2}{r-4}}$

where r is the number of rows in matrix A from which we extract the 4 degrees of freedom of the similarity transform. The factor 2 in the numerator accounts for the fact that the squared errors in 2 rows must be summed to measure a squared image distance [5].

The error *e* is then compared to a pre-defined threshold *T* to determine if the new training image should be clustered with an existing model view. *T* is selected to be 0.05 times the maximum dimension of the training image, which results in clustering views that differ by less than roughly 20 degrees rotation in depth. When a new training image is inputted into the system, it is matched to previous models views and depending on whether there is a match and/or the value of *e* one of three cases can occur:

1) The training image matches an existing model view but $e > T$. In this case a new model view is created using the training image
2) The training image matches an existing model view and $e \leq T$. Here the training image is then clustered with the existing model view. All features from the training image are transformed into the coordinates of the model view using the similarity transform solution.
3) The training image does not match an existing model view and a new cluster model is created.

The clustering algorithm combines multiple object views into a single representation of the object which allows us to robustly recognize the object from various viewpoints even if occlusion occurs.

C. Object Tracking

SIFT is a particular slow combination of a detector and descriptor. To improve the performance of the system without GPU implementations of SIFT, a simple optimization was used. If the target object was seen in the previous image, only the region of the image it was located in plus a border of 50 pixels was stored. Only this region of the next image was processed with SIFT. If the object was lost the region was allowed grow to the full resolution of the camera.

The approach assumes that the object is moving slowly. This is a fair assumption when an object is being handed to the robot.

D. Robot Control

During training of the object, the 3D coordinate of the object's center was recorded. For each feature extracted on the object, its depth and location from the object's center was associated the feature.

When the robot is made to grasp an object, the following steps are taken.

1) The WAM manipulator is moved to a default position with the arm and end-effector pointing directly up.
2) An image is taken from both cameras and the lab camera image is labelled with depth information.
3) The object recognition component is applied to the image and a set of matched features are output. Each matched feature proposes an object center. The proposed centers are adjusted by the depth information provided by the ToF camera. They are then to estimate the objects position in space.
4) An inverse kinematic solution is found which brings the Barrett Hand to a point 5 centimeters above the object. The destination pose is set with end-effector palm face down. Because the WAM is a redundant manipulator there will be a 1 DoF set of solution. The solution closest in joint space, in a Euclidean sense, to the WAM's current pose is selected. This ensures motion is smooth.
5) The Barrett is moved a fix percentage toward this point from a pose of the arm pointing directly up.
6) The system will return to step 2 unless the end-effector has arrived above the object and the object's position has been stable for T loops.
7) The end-effector fingers are spread and closed.

The robots approach from the top is to reduce the likelihood of collision with the user and to keep the end-effector from obscuring the vision component's view of the target object.

III. Experiment

Figure 3 illustrates the accuracy of the calibration between the lab camera and ToF camera. Points from the ToF camera were labelled by intensity and projected using the estimated cameras' extrinic parameters to show what the lab camera would see. This is compared to actual lab camera picture. To make the project image more visible it was convoluted with a 2x2 box filter. The accuracy of the calibration process is clearly acceptable for the grasping goal.

To obtain inverse kinematic solutions for the robot controller the method described in [2] was used. It requires the location of the cup in the frame of the robot. To determine this, a chessboard was attached to the side of the Barrett WAM which OpenCV was able to locate. The displacement between the center of the chessboard and the reference frame of the robot was hand tuned. Figure 4 shows the result.

The system was trained on the object shown in Figure 1 using the above methods. The grasping component was executed in the absence of obstacles and the result is shown in Figure 4. The limited dexterity of the Barrett Hand requires the user to help the robot to a minimal degree. It is thus difficult to quantify the performance of the system as a whole without introducing an element of subjectivity.

The variant implementation of SIFT used for the proposed system is libSIFTFast [9].

IV. Conclusions and Future Work

A system was developed to grasp an object from a user's hand. A simple fusion process was used to combine time-of-flight (ToF) and normal camera data to locate and track

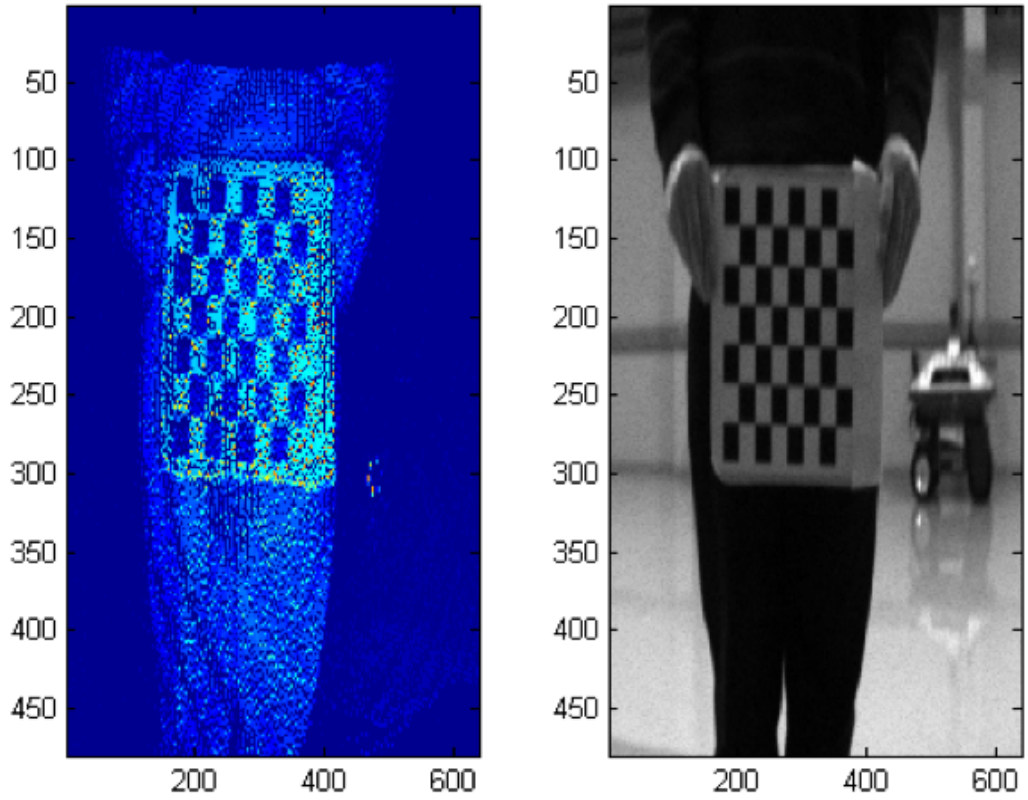

Fig. 3. On the left is the result of projecting the point cloud from the ToF camera into the estimated lab camera image plane. The right image shows the image from the lab camera.

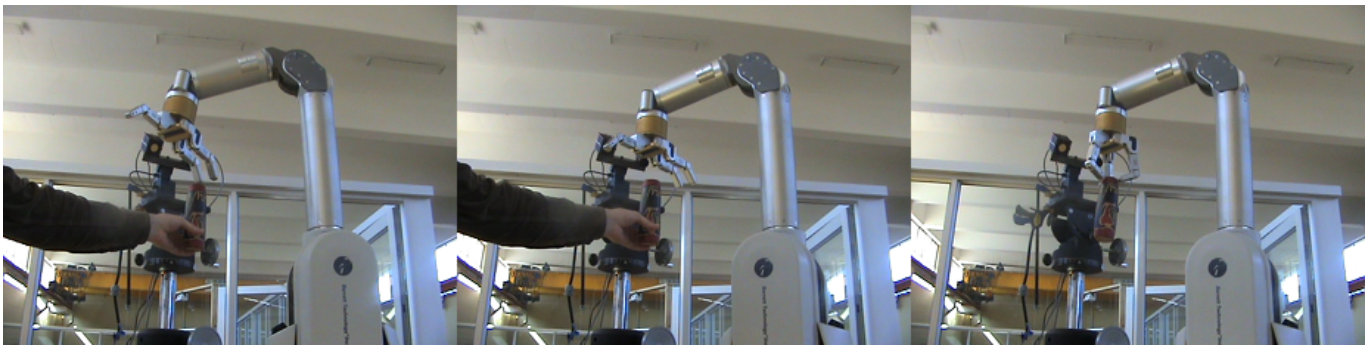

Fig. 4. The Barrett WAM executing the grasping program.

the target object. The object is recognized using the normal camera data using view clustering.

It is a first step in tackling this difficult problem and requires some assistance from the user because the dexterity of the robot hand is limited and there are no tactile sensors installed. Future work will be adding tactile information in the loop and replacing the simple grasp planner with a robust grasp planning system. This planner will require finger locations which is also a separate research problem.

REFERENCES

[1] B. Siciliano, O. Khatib, "Springer Handbook of Robotics," Springer, 2008.

[2] G. K. Singh, J. A. Claassens, "An Analytical Solution for the Inverse Kinematics of a Redundant 7DoF Manipulator with Link Offsets, " IEEE/RSJ International conference on Intelligent Robots and Systems, Taiwan, Taipei, 2010.

[3] A. Edsinger, C. C. Kemp, "Human-Robot Interaction for Cooperative Manipulation: Handing Objects to One Another, " The 16th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN), 2007.

[4] D.G Lowe, "Distinctive Image Features from Scale Invariant Keypoints", *International Conference on Computer Vision*,vol.60 (2004), pp- 91-110.

[5] D.G Lowe, "Object Recognition from Local Scale-Invariant Features", *International Conference on Computer Vision*,1999, pp-1150-1157.

[6] D.G Lowe, "Local Feature View Clustering for 3D Object Recognition ", *Computer Vision and Pattern Recognition*,2001, pp- 682-688.

[7] D.H Ballard, "Generalising the Hough Transform to detect arbitary patterns ", *Pattern Recognition*,vol.13 (1981), pp- 111-122.

[8] A.R Pope and D.G Lowe, "Probabilistic Models of Appearance for 3D Object Recognition ", *International Journal of Computer Vision*,vol.40 (2000), pp- 149-167.

[9] SIFT Fast, http://sourceforge.net/projects/libsift/, last accessed on the 8/10/2010.

[10] S. May, S. Fuchs, D. Droeschel, D. Holz and A. Nchter, "Robust 3D-Mapping with Time-Of-Flight Cameras," International Conference on Intelligent Robots and Systems, 2009.