Cylindrical Pellet Pose Estimation in Clutter using a Single Robot Mounted Camera

Punit Tiwan
Electrical Engineering
Indian Institute of Technology Delhi
New Delhi
tiwanpunit@gmail.com

Riby Abraham Boby
Mechanical Engineering
Indian Institute of Technology Delhi
New Delhi
ribyab@gmail.com

Sumantra Dutta Roy
Electrical Engineering
Indian Institute of Technology Delhi
New Delhi
sumantra@ee.iitd.ac.in

Santanu Chaudhury
Electrical Engineering
Indian Institute of Technology Delhi
New Delhi
santanuc@ee.iitd.ac.in

S.K.Saha
Mechanical Engineering
Indian Institute of Technology Delhi
New Delhi
saha@mech.iitd.ac.in

ABSTRACT
Pose estimation of cylindrical pellet using a single camera-in-hand configuration of a robot is discussed in this paper. Approaches to estimate pose in both isolated and an occluded environment is discussed. The pellet contour from the segmented image of the scene was compared with contours in the database to ascertain the matching orientation. For occluded pellets, a multiple-view based pose recognition system is proposed. Later, the estimated pose was communicated to the robot to enable it to pick-up the pellet. This has been experimentally implemented for cylindrical pellets and the performance is discussed. The algorithm enables online pellet pose determination and pick-up using KUKA KR5 robot.

Categories and Subject Descriptors
I.4.9 [Image Processing and Computer Vision]: Applications. I.2.9 [Artificial Intelligence]: Robotics - Commercial robots and applications. I.4.8 [Image Processing and Computer Vision]: Scene analysis- object recognition. J.6 [Computer Applications]: Computer-aided engineering - Computer-aided manufacturing (CAM).

General Terms
Pose estimation using monocular vision, curve based matching,

Keywords
computer vision, robotics, pose

1. INTRODUCTION
Manufacturing industries require higher levels of automation to attain higher productivity. One of the task which is difficult to automate is picking and assembling of objects. This is because pose estimation becomes difficult because of the fact that many of engineering components like cylindrical objects are without prominent features. Another issue is that many of the manufactured components are gathered in bins which give rise to mutual occlusion between objects. Popular approaches of pose estimation include the use of stereo-vision [1] and range sensors[2]. Stereo setup can be used to reconstruct the scene by finding correspondence between two images. However, this can become problematic if objects possess fewer features or landmarks to produce the point correspondence and thus the disparity map. Another approach is the use of range sensors to get the 3D point cloud of the scene to estimate the object's pose. Though this is the most common method of implementation of bin picking, it is time consuming and computationally intensive. Another suitable alternative is to use multiple views from a camera to estimate the pose. Later texture extractors like SIFT[7, 8] etc can be used to determine features for matching. In this type of technique, images of object are taken from different angles and its features are saved in database. At the time of pose estimation, features are extracted and matched with those from the database to find the matching pose. But objects like cylindrical pellets (many industrial applications uses cylindrical bar stocks for manufacturing) are featureless and thus above technique will fail in this case. Added to it, this is a very costly solution in terms of both price and time. This paper proposes an algorithm which will estimate the pose of a cylindrical object online using monocular vision.

The paper is organized as follows. Section 2 discusses the pose estimation problem for a pellet constrained to a plane. Section 3 explains pose estimation in occlusion. Set up for robot performance evaluation is presented in Section 4. Results are discussed in Section 5. Finally conclusions are discussed in Section 6.

2. POSITION AND ORIENTATION ESTIMATION
In the case of previous research in the area of estimating pose using monocular vision [3, 4], the trend has been to use a database to learn about the model and then to estimate the position and pose of a pellet. In contrast, we propose a slightly different approach based on matching the database contours with the contours of segmented image.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
Request permissions from Permissions@acm.org.
AIR '13, July 04-06 2013, Pune, India
Copyright 2013 ACM 978-1-4503-2347-5/13/07...$15.00.
http://dx.doi.org/10.1145/2506095.2506149

2.1 Position estimation

If an attempt is made to map a 2D image point to 3D, then the result obtained is a line, instead of a unique point [1]. To obtain a unique point, one value among either the x, y or z coordinates is required. The fact that all pellets are lying on a plane, i.e., the point on the pellet that touches the planar surface will have a fixed z axis coordinate value can be exploited to find other two values using the following equation

$$A_{2\times2}P_{2\times1} = B_{2\times1} \quad (1)$$

where

$P_{2x1} = [X, Y]^T$ is the unknown world coordinate

$$A_{2\times2} = \begin{bmatrix} M_{1,1} - xM_{3,1} & M_{1,2} - xM_{3,2} \\ M_{2,1} - yM_{3,1} & M_{2,2} - yM_{3,2} \end{bmatrix}$$

$$B_{2\times1} = [M_{3,4} + M_{3,3}Z - M_{1,4}, M_{3,4} + M_{3,3}Z - M_{2,4}]^T$$

$[x,y]^T$ is image coordinate,

$M_{i,j}$ are the values from the camera matrix with index $[i,j]^T$ obtained from the calibration information and

Z is known value of Z coordinate of the plane in world coordinate system.

To fix $[x,y]^T$ in image, the point on the object, where it touches the table is considered.

2.2 Segmentation

We have used S.K.Nayar's [5] approach of relative reflectance ratio for segmentation. It finds connection between two neighbouring pixels on the basis of reflectance ratio and distinguishes pellets from background. This approach gives nearly consistent results for a wide variety of illumination.

2.3 Orientation estimation

This section will describe different steps used for estimating orientation of the pellet. We subdivide the process into following sections namely

1. Database Creation
2. Curve based matching
3. Hierarchical search

2.3.1 Database creation

Using a 3D model of the pellet, we create a database of its appearance in 2D when viewed from camera. This can be done offline. The camera matrix which is needed for this step can be obtained by using any camera calibration technique. Additionally, since the workspace is small, the perspective effect on 2D appearance, due to change in position of pellet will be very small and can be neglected. This will reduce the size of the database. Firstly the 3D position of pellet in the workspace was fixed and then, the orientation was changed to create the database. Using 3D transformation, points on pellets in 3D can be easily transformed according to the required orientation. This helps in making the algorithm translation and rotation invariant. Since the size of object is within a given range the algorithm does not consider scale variations. With the help of following equation, the projection of a 3D point in image plane can be estimated

$$\lambda p_i = MP_w \quad (2)$$

where,

p_i is 3×1 matrix of homogeneous image coordinate,

P_w is 4×1 matrix of homogeneous world coordinate ,

M is3×4 matrix of camera parameters and λ is a scaling constant.

2.3.2 Curve based matching

The decision about match is made (about matching of two curves) by calculating the average distance between pixels of two curves.

$$weight = \left(md - \frac{\sum_{i=0}^{n}\|P_{Ai} - P_{Bi}\|_2}{n*md}\right) * 100 \quad (3)$$

where,

P_{Ai} = i^{th} pixel in curve A

P_{Bi} = i^{th} corresponding pixel of curve A in curve B

n = total number of pixels in curve A

md = maximum distance between two pixels, fixed before running algorithm

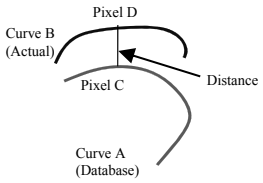

Figure 1. Corresponding pixel of C in Curve A on curve B

To find the correspondence between pixels of two curves, we draw a normal on a pixel of reference curve and find its intersection point in the other curve. If no corresponding point is found, then, instead of distance, we give penalty (some high value) for that pixel. To illustrate the process, consider two curves A and B such that Curve B is the edge image of the real pellet and curve A is the edge image of the database pellet (see Figure 1). For finding corresponding point of pixel C in curve B, we draw a normal on curve A at point C and the point at which this line intersects curve B is the required point.

2.3.3 Hierarchical search

A hierarchical search was used to ascertain orientation of pellet. In hierarchical searching, discrete data are initially taken and some filtration is used to localize the area of interest. Using this, the search can be continuously refined to find the desired result. This search gives results equivalent to exhaustive searching but has very low computational complexity. For example, let the pose of pellet be 37.From 360 angular values, we initially reduce the solution space to 20 angular values (by taking samples with step size of 20) and then to 5.Finally we select a single angular value which is the estimated pose. The algorithm with curve based matching is explained in Figure 2.

1. SegmentImage()
2. FindContour() to obtain contour edges and pick anyone of the contour's edge, say *curveA*
3. Iterate through database pellet with different orientation angles and step size of 20, say *curve B*

$$\text{weight} = md - \frac{\sum_{i=0}^{n}\|P_{Ai} - P_{Bi}\|_2}{n*md}$$

AddWeightInList(*weight,curveB*)

GO TO STEP 3 till all permutation of angle is referenced

4. IF all permutations are referenced then
   temppellet = FindMinWeightedPellet()
5. Iterate through database pellet with angle differed to temppellet by ±10 and with step size of 5. Repeat process as done in step 3.
6. After all permutations are referenced then
   temppellet = FindMinWeightedPellet()
7. Again iterate through database pellet with angle differed to temppellet by ±3 and with step size of 1. Repeat process done in step 3.
8. The pellet which has the minimal weight will be the final pellet.

Figure 2. Curve matching algorithm with hierarchical search

3. POSE ESTIMATION OF OCCLUDED PELLETS

SURF, SIFT and linear moments [7-9] have been used for pose estimation and identification of objects with prominent features. For pose estimation of objects without features (like cylindrical objects) this cannot be applied. We propose an approach for cylindrical objects which are feature-less, i.e., they do not have prominent point features.

Figure 3(a) represents one scenario where occlusion is present and Figure 3(b) represents segmentation of the given workspace. Numbering on segmented regions of image has been done manually. As one can see in Figure 3(b), some segmented contours contain information of two or more pellets. Segmentation is unable to differentiate pellets which are touching each other. Another observation which can be made through Figure 3(b) is that single view is not enough to estimate pose of pellet. This is because the pellet may either be surrounded or hidden by other pellets and sometimes shape information of pellet extracted from a scene is not enough for estimation of pose. Hence one has to take another view to confirm hypothesis. Our intuition of taking multiple views to recognize pose comes from paper on multiple view object recognition[9],where the object is identified using multiple views. Process of calibration, segmentation and database creation will remain the same as discussed before. Explanation of pose estimation in occlusion is as follows.

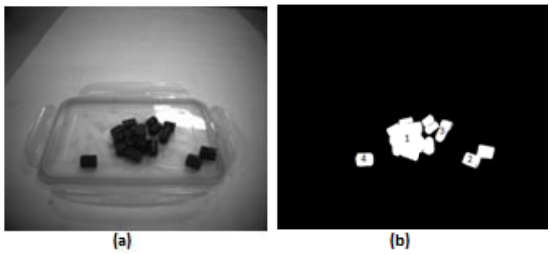

Figure 3.(a)Scenario of occlusion (b)Segmented image of scene

Initially, position of pellet is found. As explained earlier since pellets are touching the table, one can find its position by finding lowermost point of contour in image and then find 3D world coordinate using camera matrix and fixed Z coordinate(Section 2.1). Next step is to find orientation of pellet. As discussed earlier, in some cases one view is not enough to correctly identify orientation of pellet. Thus one has to make a list of hypothesis and verify it from another view. Since camera is mounted on robot, one can move robot dynamically to second location to confirm correctness of hypothesis. The camera pose corresponding to two different views are shown in Figure 4. First the robot takes the camera to pose 1 and then to pose 2 as shown in Figure 4.

3.1 Formulation

Use of this approach raises many issues. Step by step solution of these issues will complete the description of algorithm. Issues and their solutions are as follows.

3.1.1 Finding same object in both views

To solve this problem we relied on the high positioning repeatability of the robot. We calibrated camera on the second location corresponding to the second view. KUKA KR5 robot has high positioning repeatability, and thus it was assumed that the camera always attains the same pose again. This ensured that camera matrix remained the same once calibration is done for the particular pose.

We still have the problem of finding a correspondence. To solve this we take into consideration the position of the pellet estimated in first view. We re-project a virtual pellet placed on a location estimated before and then using camera matrix compute a second view. After getting the contour, we take the image of the scene in the second view and segment it. The segmented contour containing the re-projected contour will be the required contour.

3.1.2 Creation of database to match objects in next view

Solution to this problem again depends on the high positioning repeatability of the movements of the robot. If we move the robot such that its distance from the center of the workspace remains same but only its orientation changes, we can easily use the previous database. If distance between camera and center remains same then

$$\theta_{new} = \theta_{old} + \theta_{rotated} \quad (6)$$

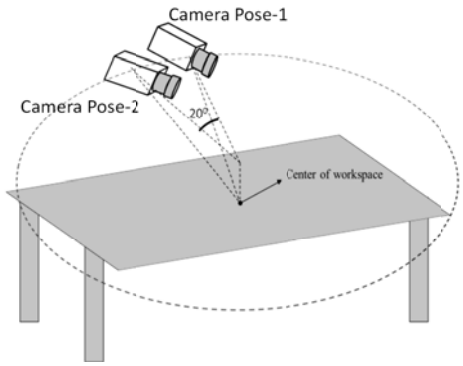

Figure 4. Schematic of camera pose for multiple views.Camera Pose-1 and Camera Pose-2 corresponds to first view and second view respectively

For example, we move robot in a circular motion about the center of the workspace. We move it by -20 degrees in Z-axis as shown in the Figure 4. So if we want to match pellet having angle 100o then we will pick pellet with orientation 100-20 = 80o from the database.

The change in angular orientation was fixed as a small angular value (20° in this case) due to the following reasons:

- Problem in correspondence: If movement of robot becomes too large then problem in correspondence occurs. It may happen that other pellets will completely hide the interested pellet in the new pose of the robot.
- Intuitive behavior: This is similar to the intuitive behavior of humans. To disambiguate an pellet from its surrounding, instead of moving our body we just tilt our head. The motion of camera is comparable to that motion.
- Saving time: Movement of robot consumes time. Through small movements one can reduce pick up time since picking up time is an important criteria in bin picking applications.

3.1.3 Hypotheses formulation

In hierarchical search as stated before, we initially matched the segmented contour with the database contours with angles varying from 0 degree to 180 degree with step size of 20°.Later the area of search was reduced by reducing the step size. Therefore if due to error of segmentation or occlusions, wrong assumption is made at the first stage, then final result will be wrong.

Hence we make use of tree structure during creation of list of hypothesis. In the first step, database contours are matched by varying angles with step size of 20°. Instead of taking one value, top three matching values are taken into consideration. These three values will become node of the search tree and further search will be around these values. Then search is begun around these three values by considering angles±10 of estimated value. Again top three matching values are considered and search is further continued by keeping these values as center of angular area of search. In the end, the results consist of values of different orientation and its matching percentage. These are the leaf node of the tree structure. With the help of insertion sort, hypothesis with same angles are removed from the list. If top value is above threshold and next value has matching percentage below threshold then top value will be considered as pose of pellet and robot is commanded to pick that pellet. If top value is not above a given threshold then this list is then verified in the next view.

3.1.4 Degree of Occlusion

Till now we have created the hypotheses list and verified it. We commanded the robot to pick it up. When the robot reaches the pellet's position to grab it, other pellets (which were touching the pellet) might get damaged by the gripper. This problem will arise when pellets are placed close to one another. Due to this even if we detect pellet accurately, we cannot grip it because of the surrounding pellets.

To know how many sides of pellets are open, we have tweaked the curve-fitting algorithm. The curve of the contour is found by using any standard edge detection algorithm. Then we try to fit the verified pellet into that curve using curve-fitting algorithm. But instead of calculating weight (matching percentage) by considering both curves we will consider only curve formed by re-projection of verified pellet. Equation to find matching percentage is given below

$$weight = \frac{\sum_{i=0}^{n} \|P_{Ai} - P_{Bi}\|_2}{n * md} * 100 \qquad (4)$$

where, P_{Ai} = i^{th} pixel in curve A
P_{Bi} = i^{th} corresponding pixel of curveA in curveB
n = total number of pixels in curve A
md = maximum distance between two pixels (fixed before running algorithm)
Weight will give us an estimation of how many edges are open. If weight is high then that many pixels of edges are visible on the image and hence three or four edges are open. Whereas if weight is less then lesser number of pixel of edges are visible on the image and hence one or two edges are open. Using this method one can find an pellet's pose and possibility of it being picked up safely, even in an environment having occluding pellets.

One can also quantify occlusion using following method.

To quantify occlusion one can make use of following equation

$$weight = \left(md - \frac{\sum_{i=0}^{n} \|P_{Ai} - P_{Bi}\|_2}{n * md}\right) * 100 \qquad (5)$$

where, P_{Ai} = i^{th} pixel in curve A
P_{Bi} = i^{th} corresponding pixel of curve A in curve B
n = total number of pixels in curve A
md = maximum distance between two pixels, fixed before running algorithm
Curve A represents curve stored in database and curve B represents curve extracted from segmented image. Method of finding corresponding pixel and distance is as explained in Section 2.1.

On the basis of computed weight we have empirically quantified occlusion in following manner according to the ranges in weight.

- If weight is in the range**[100,90]→ No Occlusion**
- If weight is in the range**(90,70] → Low Occlusion**
- If weight is in the range **(70,55]→ Medium Occlusion**
- If weight is **below 55 → High Occlusion**

4. REPEATABILITY MEASUREMENT

Since the discussed methods assume that the robot reaches the taught position with very less error, the robots repeatability has to be measured. For repeatability measurements, a single camera was mounted on the end-effector of the robot and oriented such that the axis of rotation of the last link is parallel to the camera optical axis. The orientation of the end-effector was orthogonal to the planar surface. The two fold advantage of this orientation is that the object is always picked up from this orientation and the external camera parameters will remain the same. With regards to position, four 2D points were fixed such that they form the four extremities of the workspace where the pellet will be placed for pick up by the robot. In this experiment a square area of approximately 200 mm×200 mm was considered as the workspace. The fifth point was considered as the point of intersection of the diagonals of the square. The home point was considered as the pose of the robot where the camera has been calibrated for pellet pose estimation. No extra load was placed on the gripper and the experiments were done as per the details in [10, 11].

A blank sheet was pasted on a planar surface, for e.g. a writing desk. On the paper, a square was sketched using a pencil. The robot end-effector was taken to the position while maintaining a constant z value and orientation. The camera calibration was done using the edges of a calliper for the given z axis value and orientation (implying internal camera parameters are estimated while external camera parameters are assumed to be constant). This was maintained throughout the experiment. The robot path was fixed over each of the corners. Later, a hand to eye measurement was made using the same pose of robot to measure the position of the robot end-effector from the camera images. Then the robot was taken in a cycle across the five points as per [10]. At each point, the images were taken and then, processed and segmented. A Hough transform [12] based approach was used to obtain the lines in the images. The displacement of the point of intersection with respect to the camera position was measured. Since the camera was rigidly mounted on the end-effector, the measurements made with respect to the camera correspond to the original position of the robot end-effector. The set-up is shown in Figure 5.

5. RESULTS AND DISCUSSIONS

A BASLER PILOT camera was mounted on the end-effector of KUKA KR5 robot. Robot Sensor Interface (RSI) was used to communicate with the robot through Ethernet. The computer system used was based on Intel Xeon processor. Pellet's diameter and height were approximately 12 mm and 14mm respectively. The diameter and height vary within a range of 2 mm and 10mm respectively. The pellets were kept on the planar surface at random orientations in a non-occluding fashion at the first stage of experiments. Direct Linear Transformation (DLT) is used for camera calibration [6].

Firstly camera calibration was done for a fixed pose of the robot. Later, cylinders of different radii and heights were considered and the database was created. This can be done offline. The image of

scene was taken when the robot reached its home position, i.e. the position at which camera was calibrated and then the image was fed to the algorithm. The output of the algorithm, i.e., the pose of the pellet was sent to the robot, so that it can could move to the desired location and pick up the pellet.

Table 1. Result of pose estimation for isolated pellets

Average error in estimating orientation (degree)	2.4
Average error in estimating position (mm)	3
Average time taken(ms)	221

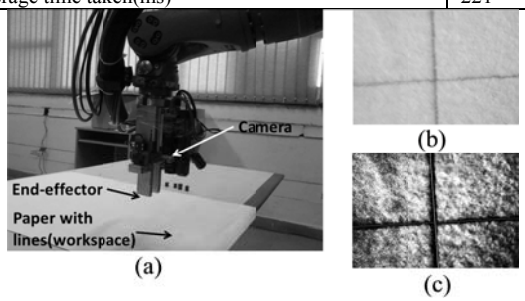

Figure 5. Set up for 2D accuracy measurement. (a)Camera in handconfiguration with optical axis perpendicular to plane, (b) Image of the line sketched on paper, (c) Lines detected using Hough transform and their point of intersection.

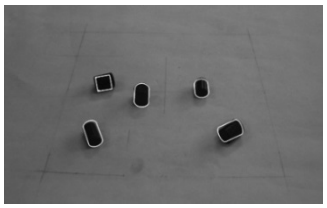

Figure 6 Final edge image. The detected contour is shown in white.

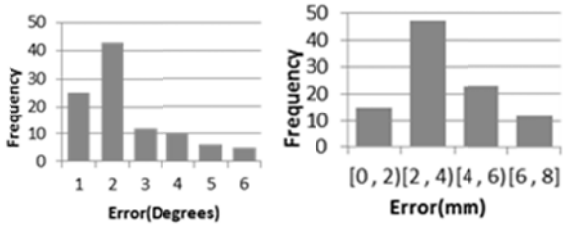

Figure 7.Graphical representation of errors for isolated pellets in (a)Orientation (b)Position

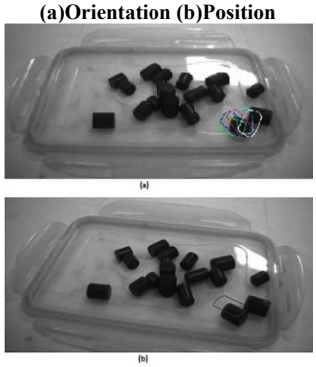

Figure 8 (a) Image of scene from first view. Different hypotheses are superimposed on the image (b) Image of scene from second view. Confirmed hypothesis is superimposed on the image

The images of different cylindrical objects and the detected contours from the database are shown in Fig. 6. Both matching algorithms perform similarly. As shown in Table 1, the average error in estimating pose of a pellet is 2.4 degrees. Sometimes due to segmentation error, some parts of shadow also get included with the contour of the pellet, thereby causing the pose estimation to become inaccurate. Figure 7a shows the frequency of distribution of errors in orientation. Figure 7b shows the frequency of distribution of errors in position.

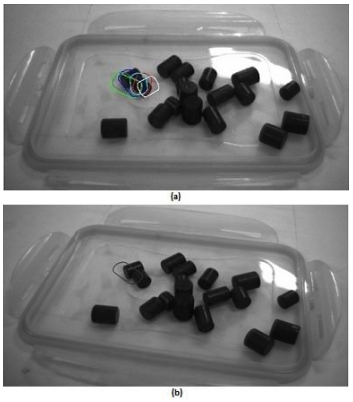

Figure 9. (a)Image of scene from first view. Different hypotheses are superimposed on the image (b) Image of scene from second view. Confirmed hypothesis is superimposed on the image

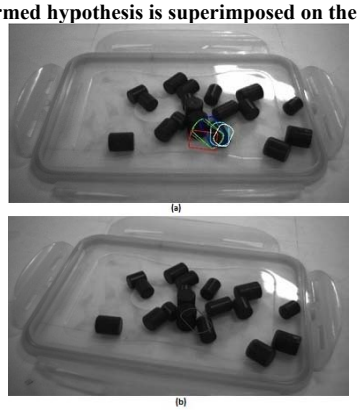

Figure 10. (a) Image of scene from first view. Different hypotheses are superimposed on the image (b) Image of scene from second view. Confirmed hypothesis is superimposed on the image

Later experiments were conducted with cylindrical pellets placed in occluding fashion. Figures 8, 9 and 10 show images where occlusions are present. Figure 8(a) shows various hypotheses created for a pellet with different colors and Figure 8(b) shows confirmed hypothesis from second view. Similar is the case for Figure 9 and 10. Once the hypothesis is verified from the second view the robot is commanded to implement the pellet pick up depending upon the degree of occlusion. Table 2 shows the results of pose estimation for occluded pellets. As shown in Table 3, pellet for which hypotheses are made in Figure 10 is surrounded by many pellets so quantification of occlusion computed is high and also after computation it is categorized as non graspable. Whereas occlusion computed for pellets shown in Figure 8 and 9 are low and after computation it is categorized as graspable.

The pose estimation is based on the fact that for a small region, the perspective effect will be negligible and one database can be used for the whole workspace area. But we do not want to restrict this method for small workspace alone. To extend this method for application in larger workspace, the workspace was divided into 100×100mm squares. For each square area we create done

database. Before estimating the pose, the position of the pellet was determined and then using a lookup table, the corresponding database suitable for that area was selected and subsequent matching was done.

Repeated tests have proved that the proposed algorithm would enable successful pick up of the cylindrical pellets. Since the pellet shape was known, the present approach is well suited and justified. The time required for estimating pellet's pose can be reduced further with the use of parallel processing, since the matching of two database curves with original one does not depend on each other.

The methodology discussed here assumes that the repeatability of robot is very high. The measured repeatability values are tabulated in Table 4. Low values of repeatability support the assumption of the calibration matrix remaining unchanged after calibrating for a given pose. This also annuls the need of a computationally complex visual servoing based algorithm to pick up pellets whose pose has been estimated.

Figure 11 shows the image of the pellet in different lighting conditions. The thresholding operation has given good results in both high illumination and low illumination. This makes it possible to implement the algorithm with fewer constraints on the lighting condition.

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

Figure 11 Results of thresholding with varying background background; (a) Image in dark background; (b) thresholded image of a; (c) image with bright background; (d) thresholded image of d.

Table 2. Result of pose estimation for occluded pellet

Average error in estimating orientation (degree)	12
Average error in estimating position (mm)	5
Average time taken (ms)	571

Table 3. Quantification of occlusion and grasping

Pellet showed in Figure	**Graspable**	**Occlusion Quantization**
8	Yes	Low
9	Yes	Low
10	No	High

Table 4. Repeatability measurements for robot

Position	**Repeatability (mm)**
1	0.52
2	0.19
3	0.41
4	0.20
5	0.22

6. CONCLUSION

Pose estimation of cylindrical pellets for pick up by industrial robots has been proposed. An orientation estimation method based on curve based matching with a database is proposed. The pellet position is obtained from the calibration information for the given camera position and orientation. This information will enable pick up of pellets using the robot's gripper. For occluded pellets a slightly varied approach based on multiple views was used. The probable poses have been estimated from one view and then verified using the next view. The possibility of successful pick up has been quantified using a measure of occlusion which can be utilized to decide whether to attempt a pick up or not in the given occluded environment. The performance of these algorithms have been tested and quantified. The performance proves that this method can be used for robot based bin picking applications and can replace computationally intensive visual servoing and laser range scanning based methods.

7. ACKNOWLEDGMENTS

The authors are thankful to the Board of Research in Nuclear Sciences, India for grant towards setting up of Programme for Autonomous Robotics in IIT Delhi.

8. REFERENCES

[1] Hartley, R. and Zisserman, A. Multiple View Geometry in Computer Vision, Cambridge University Press, Cambridge 2001.

[2] Simon, D. A. Hebert, M. and Kanade T. 1993. Real-time 3-D Pose Estimation Using a High-Speed Range Sensor, Robotics Institute, Paper 460.

[3] Amano, T. and Tamaki, T. 2009. An appearance based fast linear pose estimation, Proceedings *of* MVA. IAPR Conference on Machine Vision Applications, pp. 182-186.

[4] Brandao, M. Bernardino, A. and Victor, J. S. 2011. Image driven generation of pose hypotheses for 3D model-based tracking, Proceedings of the 12th IAPR Conference on Machine Vision and Applications, pp. 59-62.

[5] Nayar, S. K. and Bolle, R. M. 1993. Computing reflectance ratios from an image, Pattern Recognition, v-26(10), 1529-1542.

[6] AbdelAziz, I. Y. and Karara, H. M. 1971. Direct linear transformation into object space coordinates in close-range photogrammetry, Proc. Symp. Close-Range Photogrammetry, pp. 1-18.

[7] Björn, J. and Moe, A. 2005. Patch-duplets for object recognition and pose estimation, Proceedings. 2nd Canadian Conference on Computer and Robot Vision, 9-16.

[8] Changchang, Wu. 2008. 3D model search and pose estimation from single images using VIP features, Computer Vision and Pattern Recognition Workshops, CVPRW'08, 1-7.

[9] DuttaRoy, S. Chaudhury, S. and Banerjee, S. 2000. Isolated 3D object recognition through next view planning, IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans, v- 30 i1, 67-76.

[10] ISO 9283: 1998. Manipulating industrial robots — performance criteria and related test methods.

[11] ISO/TR 13309: 1995. Manipulating Industrial Robots - Informative Guide on Test Equipment and Metrology Methods of Operation for Robot Performance Evaluation in Accordance with ISO 9283.

[12] Duda, R. O. and Hart, P. E. 1972. Use of the Hough transformation to detect Lines and curves in pictures, Comm. ACM, V-15, 11–15.