EFFICIENT VISUAL GRASPING ALIGNMENT FOR CYLINDERS

NAGW-1333

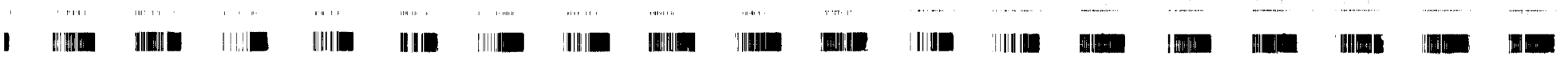
by

Keith E. Nicewarner and Robert B. Kelley

Rensselaer Polytechnic Institute
Electrical, Computer, and Systems Engineering Department
Troy, New York 12180-3590

December 1991

CIRSSE REPORT #110



Efficient Visual Grasping Alignment for Cylinders

Keith E. Nicewarner and Robert B. Kelley

Center for Intelligent Robotic Systems For Space Exploration
Electrical, Computer, and Systems Engineering Department
Rensselaer Polytechnic Institute, Troy, NY 12180

ABSTRACT

Monocular information from a gripper-mounted camera is used to servo the robot gripper to grasp a cylinder. The fundamental concept for rapid pose estimation is to reduce the amount of information that needs to be processed during each vision update interval. The grasping procedure is divided into four phases: learn, recognition, alignment, and approach. In the learn phase, a cylinder is placed in the gripper and the pose estimate is stored and later used as the servo target. This is performed once as a calibration step. The recognition phase verifies the presence of a cylinder in the camera field of view. An initial pose estimate is computed and uncluttered scan regions are selected. The radius of the cylinder is estimated by moving the robot a fixed distance toward the cylinder and observing the change in the image. The alignment phase processes only the scan regions obtained previously. Rapid pose estimates are used to align the robot with the cylinder at a fixed distance from it. The relative motion of the cylinder is used to generate an extrapolated pose-based trajectory for the robot controller. The approach phase guides the robot gripper to a grasping position. The cylinder can be grasped with a minimal reaction force and torque when only rough global pose information is initially available.

1 INTRODUCTION

A proposed construction of the NASA space station involves a large truss structure composed of 2–5 meter struts and reconfigurable nodes. At the Center for Intelligent Robotic Systems for Space Exploration (CIRSSE), we are interested in automating the assembly of these struts and nodes. The CIRSSE testbed consists of 2 9-DOF robots, 2 robot grippers equipped with force and cross-fire sensors, 2 force-torque sensors for each robot wrist, a pair of cameras mounted on each robot, 2 stationary cameras, and a laser scanner. The stationary cameras can give rough global pose information of the struts in the assembly area. These pose estimates would be insufficient for such operations as grasping or inserting a strut. The arm cameras provide a means for refining the global pose estimates of struts.

Figure 1 shows the mounting of the cameras on a robot. Although the vision-guided grasping algorithm discussed in this paper uses only one camera, the two cameras on the arm allow for future research with stereo vision. Currently, the two cameras allow us to select a camera with an unobstructed view. Also, the visual servoing algorithm can be used on two simultaneous images to match the pose of two cylindrical objects, as is the case for vision-guided insertion of a strut.

Solutions to the visual servoing problem can be grouped into a hierarchy. At the top of the hierarchy is the placement of the cameras: stationary cameras as opposed to an eye-in-hand approach. Using stationary cameras

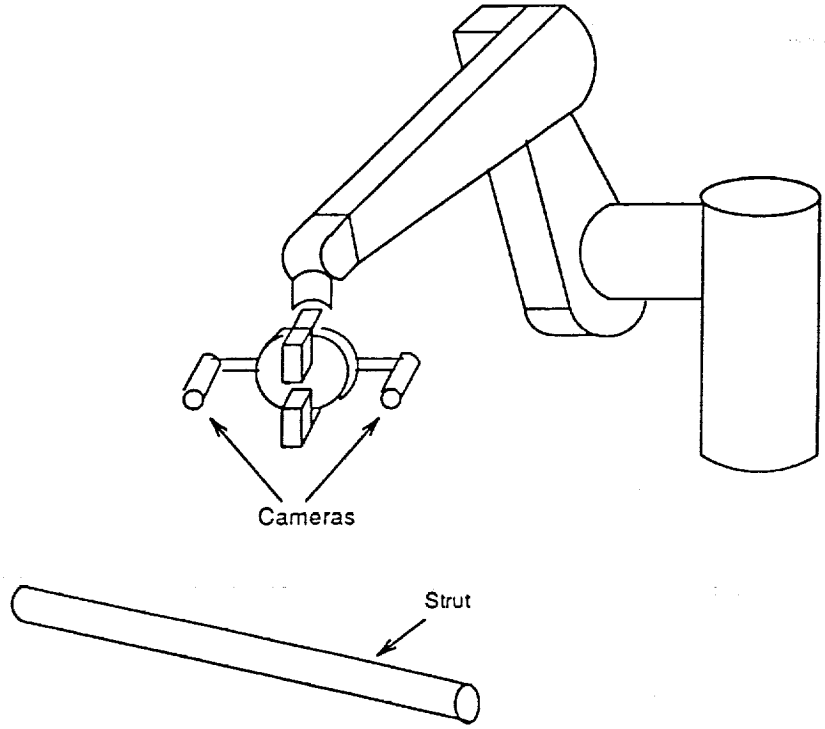


Figure 1: Dual cameras mounted on the gripper of a PUMA robot.

simplifies control because we can assume that any observed motion is due to the object alone and not to the robot's motion. However, we sacrifice either field of view or accuracy because a large field of view will imply reduced accuracy and *vice-versa*. In addition, we must contend with the fact that the robot will appear in the view. With a gripper-mounted camera pair, we aggravate the control problem but we retain accuracy while having a field of view limited only by the workspace of the robot.

Another grouping in the visual servoing hierarchy is feature-based servoing versus pose-based servoing. In the feature-based case, the control target is in the image plane while in the pose-based case, the target is in either local or global cartesian coordinates. Lee and Lin[1] give a comparison of these two methods. The most important difference is this: If path planning or obstacle avoidance is needed, the feature-based servoing cannot give a direct meaning of the trajectory in task space, whereas pose-based servoing can.

Any type of visual servoing requires the ability to process visual information as rapidly as possible. An inherent problem in implementing vision guidance is that conventional vision systems usually have processing cycles that are in the range of hundreds of milliseconds since they are usually tied to the camera frame rate of 33.3 ms (30 Hz for NTSC) or 40 ms (25 Hz for PAL). Robot control systems, on the other hand, usually run with update intervals in the millisecond range. Even if the vision information can be processed instantaneously, the frame rate of the camera is still the limiting factor.

Thus, to provide accurate and useful information to the robot motion control system, the visual processing system must be able to not only interpolate between frames, but also anticipate the pose trajectory until the next frame is acquired. To achieve rapid image processing, the area of processing and the amount of processing must be reduced to the minimum. These two ideas, feature extrapolation and information reduction, are essential to real-time visual servoing.

In this paper, we concentrate on a particular instance of visual servoing: guiding a robot to grasp a strut. Monocular information from a gripper-mounted camera is used to servo the gripper to a strut. Using one camera, as opposed to stereo cameras, greatly simplifies the image processing requirements. Three-dimensional information can be obtained from the motion of the robot. A pose estimate is calculated every camera update interval and is passed to the robot controller. The pose is then extrapolated to provide rapid synchronous updates to the PID controller, thus driving the robot gripper to a grasping position. The gripper then closes with minimal reaction forces and torques.

2 STRUT POSE ESTIMATION

There are many image processing techniques which give the position and orientation (pose) of various objects. Some methods were developed to process as many sizes and shapes as possible, thereby sacrificing speed for versatility. Other methods were developed for the purpose of processing only limited types and numbers of objects, thereby increasing the speed of the image processing. Since a strut may be modeled as a cylinder, a general cylinder pose estimation method is required.

There are few pose estimation methods dealing specifically with cylinders.[2–4] These methods perform an inverse perspective transformation on the image to get an anticipated object surface, then this surface is matched with the surface of the known object. Surface matching techniques have not yet demonstrated the speed necessary for visual servoing.

We propose a rapid cylinder pose estimator which requires processing only two parallel lines in an image. This method requires that the following conditions be met:

1. There is indeed a cylinder in the field of view.
2. There is a sufficient length of the cylinder visible.
3. The edges of the cylinder can be extracted.

The first condition seems trivial. However, it is critical to establish whether there is anything in the image with which to align the robot gripper. It is also important to verify that the object is cylinder-like. The second condition requires a cylinder segment that is at least twice as long as it is wide. This insures that accurate estimates can be obtained. The third condition requires that there not be excessive noise or clutter in the image. There are a variety of edge-detection techniques, where invariably the more robust methods are more time consuming. Thus, an edge-detector should be chosen based on the current viewing conditions.

The three listed conditions form the basis of a rapid cylinder pose estimation technique. The visual-servoing procedure is dependent upon verifying these conditions. The procedure is broken into four phases: learn, recognition, alignment, and approach.

2.1 Learn Phase

Before vision-guided alignment can begin, the target pose for the strut in the robot task space needs to be defined. The target pose is defined by simply placing the strut in the gripper and noting the pose calculated by the pose estimator.

2.2 Recognition Phase

In the recognition phase, the three conditions for the rapid pose estimator must be verified. The first two conditions are verified by first looking at a thresholded image and computing the moments of inertia of any blobs. Then the moments are examined for any blobs which are basically "long and thin." Performing a Hough transform on the image can verify that the edges are straight and will indicate where the scan lines should be placed. Additional verification can be obtained through using structured light such as a laser.

The third condition is checked by trying several edge-detection methods across the estimated edges obtained previously and choosing the most appropriate one. First, a simple threshold edge detection scheme is used to check if the cylinder contrasts with the background. If there is insufficient contrast resulting from too much background clutter or image noise, the scan lines are moved along the cylinder until an acceptable edge is found. If not, then a gradient operator is used to detect high-spatial-frequency peaks. The camera is focused so that the depth of focus includes the end of the gripper, so distant clutter is usually out of focus and is thereby ignored.

Once the three conditions have been verified, the scan lines are chosen and the radius of the cylinder is estimated, if it is not known *a priori*. The scan line positions and scan ranges are chosen to minimize the noise that will be encountered during the approach phase. The scan lines are chosen to be as far apart as possible to ensure more accurate pose estimations. The pose estimation works fastest with scan lines that are aligned with either the rows or columns of the imaging plane. Hence, if the cylinder is more horizontal than vertical, vertical scan lines (columns) are used.

If not known before hand, the radius of the cylinder can be estimated by observing the change in the image induced by moving the robot a certain distance towards the cylinder. If the radius projected onto the screen at the first position is r_1 and the projected radius at the second position is r_2, the radius R can be determined by similar triangles.

$$\frac{R}{d_1} = \frac{r_1}{f}$$
$$\frac{R}{d_2} = \frac{r_2}{f}$$

where f is the focal length of the camera and d_1, d_2 are the distances from the cylinder to the camera focal point at the two positions. Recognizing that $d_2 = d_1 + \Delta d$, we can solve these equations for R,

$$R = \frac{\Delta d}{f}\left(\frac{r_2 r_1}{r_2 - r_1}\right)$$

Therefore, we can use the calibration of the robot to move a given distance and calculate an estimate of the cylinder's radius.

2.3 Alignment Phase

The alignment phase begins by processing the scan lines for edges, or *critical points*. If some unexpected noise is encountered while scanning for critical points, the scan ranges and scan line positions can be adjusted. In the next section, it will be shown that 4 of the 6 cylinder pose parameters can be determined from only 4 critical points. The pose of the cylinder is computed relative to the camera. The 4 pose parameters are:

1. θ – the clockwise rotation of the cylinder about the optical axis, relative to the image plane y-axis.
2. x_C – the horizontal displacement of the center of the cylinder from a vertical plane through the optical axis.
3. ϕ – the tilt angle of the cylinder axis out of a plane perpendicular to the optical axis.
4. d_C – the distance from the camera lens to the center of the cylinder along the optical axis.

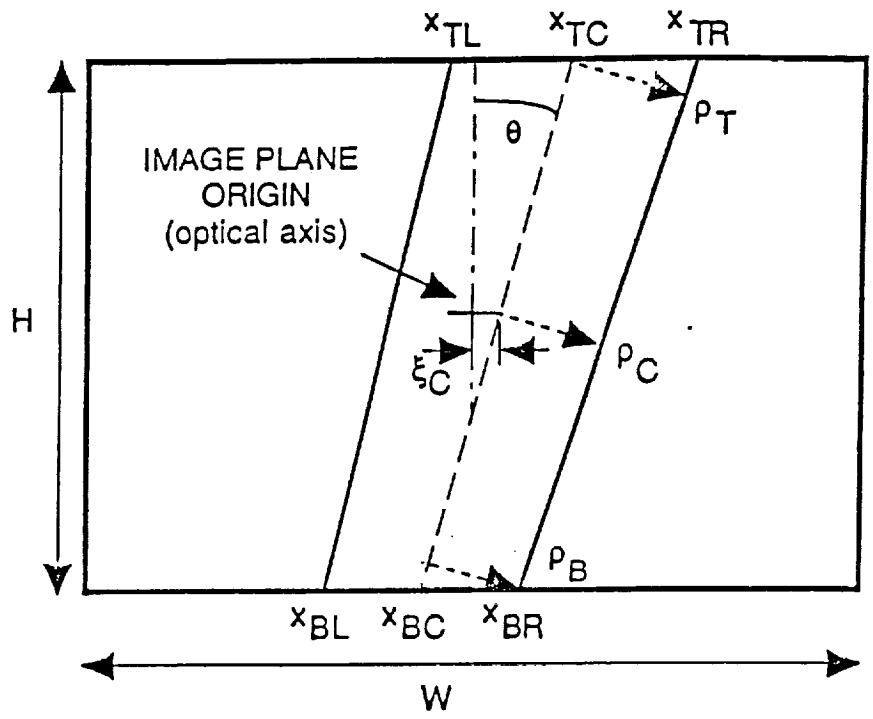


Figure 2: Image plane geometry for a cylinder.

For the alignment phase, the robot controller drives all the parameters except d_C to the desired parameters acquired in the learn phase. The distance is left out so that the cylinder remains in the field of view during the alignment. At close distances, a small motion made by the robot can drive the cylinder out of the image.

2.4 Approach Phase

Once aligned, the robot controller begins to use the distance estimate in addition to the other 3 pose parameters. The distance serves as the primary servo variable, while the other parameters serve as "guides" for the approach motion. The robot controller stops when the cylinder is between the gripper fingers, and then the gripper closes. If the cylinder is lost for any reason, the robot backs away and re-enters the recognition phase.

3 DERIVATION OF THE POSE ESTIMATE

Consider a $W \times H$ grey-scale image of a scene containing a cylinder as in Figure 2. For this discussion, it is assumed that two critical points lie on the top raster line and two lie on the bottom line, i.e., the cylinder is vertically oriented. The critical points are designated as x_{TL}, x_{TR}, x_{BL}, and x_{BR}.

Using the critical points, the orientation of the axis of the cylinder is computed using geometry. If the critical points make an angle of less than 10° with the optical axis, then the perspective distortion effects are negligible. Thus, the centerline of the cylinder image is a good approximation to the projection of the cylinder axis.

The centerline and its rotation about the optical axis are computed as follows. Define the midpoints of the top and bottom critical point pairs, x_{TC} and x_{BC}. Thus,

$$x_{TC} = \frac{1}{2}(x_{TL} + x_{TR})$$

and

$$x_{BC} = \frac{1}{2}(x_{BL} + x_{BR})$$

The clock-wise rotation of the cylinder about the image's z-axis, relative to vertical, is

$$\theta = \tan^{-1} \frac{x_{TC} - x_{BC}}{H}$$

Within the image, the apparent center of the cylinder has an x-component ξ_C given by the mid-point between the top and bottom center points. Therefore, the apparent horizontal displacement of the center of the cylinder from the optical axis is

$$\xi_C = \frac{1}{2}(x_{TC} + x_{BC})$$

The apparent radius of the cylinder at the top of the image is given by

$$\rho_T = (x_{TR} - x_{TC}) \cos\theta$$

and similarly at the bottom of the image,

$$\rho_B = (x_{BR} - x_{BC}) \cos\theta$$

These parameters are used to calculate the apparent radius of the cylinder in the center of the image:

$$\rho_C = \frac{1}{2}(\rho_T + \rho_B)$$

The tilt angle ϕ is the angle the cylinder is rotated towards the viewer relative to a plane perpendicular to the optical axis. The tilt angle geometry is shown in Figure 3. Assuming a constant optical magnification factor μ (i.e., $d_C \gg f$), the following expressions are obtained:

$$\mu = \frac{R}{\rho_T} = \frac{d_C - l\sin\phi}{f}$$

$$\mu = \frac{R}{\rho_B} = \frac{d_C + (L-l)\sin\phi}{f}$$

$$\mu = \frac{R}{\rho_C} = \frac{d_C}{f}$$

where R is the actual radius of the cylinder, L is the length of the visible portion of the cylinder, l is the length above the optical axis, and f is the focal length of the camera. Using similar triangles, we get a second set of identities:

$$\frac{H}{2f} = \frac{fl\cos\phi}{\rho_T R} = \frac{f(L-l)\cos\phi}{\rho_B R}$$

where H is the distance between the scan lines. Simplifying and solving for ϕ,

$$\phi = \tan^{-1}\left(\frac{2f}{H}\left(\frac{\rho_T - \rho_B}{\rho_T + \rho_B}\right)\right)$$

The distance to the center of the cylinder is then given by

$$d_C = \frac{Rf}{\rho_C}$$

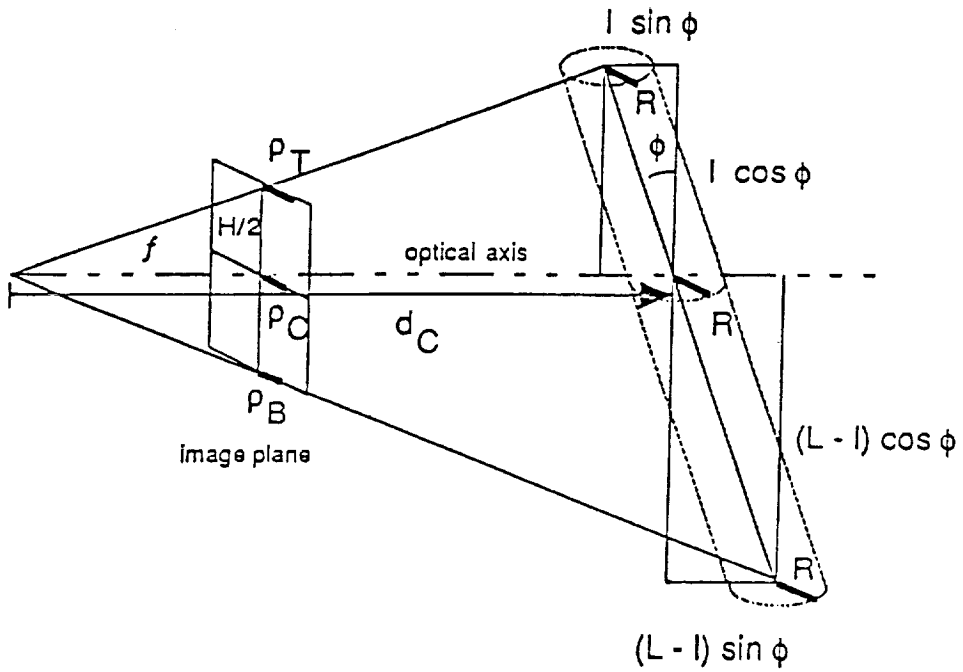


Figure 3: Tilt angle geometry.

The angle made by the cylinder axis and the optical axis is given by

$$\psi = \tan^{-1}\left(\frac{\xi_C}{f}\right)$$

And finally, the lateral displacement x_C is given by

$$x_C = d_C \sin\psi$$

The orientation angles θ and ϕ are determined from these expressions without knowledge of the actual radius of the cylinder. However, the position parameters d_C and x_C of the cylinder are expressed in terms of the actual radius of the cylinder. Therefore, with only two parallel scan lines from the image and *a priori* knowledge of the radius of the cylinder, 4 of the 6 pose parameters of the cylinder can be determined.

4 ROBOT CONTROL

The end effector of the robot is aligned with the axis of the strut in a continuous error-reducing fashion in the manner of,[5,6] as opposed to the traditional look-and-move approach.[7] This allows for slight inaccuracies in the pose estimate due to image noise, quantization errors, lens distortion, and camera parameter uncertainties. The estimate is constantly being refined, and the final pose (with the cylinder vertical and centered in the image) is chosen to minimize distortion effects.

A block diagram of the closed-loop system is shown in Figure 4. The desired pose is obtained in the learn phase. The KOH is a k^{th}-order-hold which is the model of the linear extrapolator used. The transmission delay is modeled because in our case, there is a significant delay in transmitting the pose information from the vision

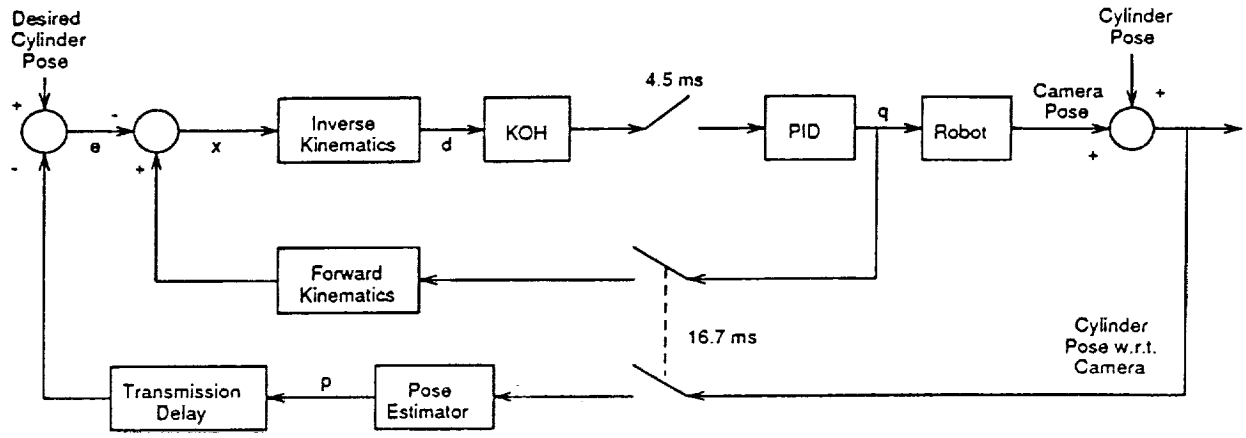


Figure 4: Block diagram of the visual servo system.

system to the robot control system. This delay ranges from 0.1 ms to 2.3 ms, depending on the local computer network traffic.

The vision updates are being calculated every 33.3 ms in the ideal case. This interval may be decreased to 16.7 ms, provided that the cylinder is vertical and the camera acquisition hardware allows access to individual fields within a frame, in which case, the scan lines both reside on either the even field or the odd field. The actual time may vary if, for example, the cylinder is lost and must be re-acquired.

For each vision update, the pose parameters are computed and in our case, transmitted to the robot controller. The pose parameters are then subtracted from the desired pose parameters, giving a set of error terms:

$$\Delta\theta = \theta_d - \theta$$

$$\Delta\phi = \phi_d - \phi$$

$$\Delta d_C = d_{Cd} - d_C$$

$$\Delta x_C = x_{Cd} - x_C$$

These errors are used to modify the current pose of the end-effector, which is represented as a homogeneous transform matrix, M. This transform matrix is obtained by using the forward-kinematic transform on the current joint angles of the robot, the vector $\mathbf{q}$:

$$\mathbf{M} = \mathbf{K}(\mathbf{q})$$

The robot pose modifications are a set of cartesian transformations on M:

$$\mathbf{M}_{new} = \mathbf{M}\mathbf{R}_z(-\Delta\theta)\mathbf{R}_x(-\Delta\phi)\mathbf{T}_x(-\Delta x_C)\mathbf{T}_z(-\Delta d_C)$$

where $\mathbf{R}_i$ is the rotation transformation matrix about axis i and $\mathbf{T}_i$ is the translation transformation matrix along axis i. The new joint angles are then computed by the inverse-kinematic transform,

$$\mathbf{q}_{new} = \mathbf{K}^{-1}(\mathbf{M}_{new})$$

The inverse-kinematic approach was chosen over a Jacobian approach such as that used by,[5,6] because first of all, it was well within our hardware capabilities. The inverse-kinematic transform takes approximately 2 ms on our system and is performed in parallel on a separate processor so as not to interfere with the robot controller. Second, the inverse-kinematic approach can alert us if the desired robot pose is out of the workspace of the robot

because positions are expressed in absolute terms. With Jacobians, positions are expressed as increments, thus giving little meaning to the singularities that will arise.

The robot controller is running at a fixed rate, 4.5 ms in our case. Thus, we need to interface the asynchronous vision updates with the robot controller. Note also that 33.3 is not a multiple of 4.5; thus aliasing will occur and must be considered. By taking the current joint positions along with the prior joint positions, a linear approximation for the robot trajectory can be constructed. Using this approximation, we generate set-points every 4.5 ms along an anticipated path for the robot. This works well if we are insured that the vision updates will be regular. If the cylinder is lost, the robot would follow the approximated path indefinitely. However, if we place an additional criterion on the approximation that the velocity along the trajectory should go to zero after, for example 66.6 ms, the robot motion halts gracefully while the cylinder is re-acquired.

The generated position and velocity set-points are fed into a PID controller. The integral term is primarily to reduce steady-state error. Because of stiction, the integral term builds up to a break-away torque and the robot jerks slightly. This jerk is observed in the imaging plane and an incorrect trajectory is computed. The visual servoing then attempts to drive the pose error to zero. The result is oscillation of the robot about the desired pose. To remedy this, a dead-zone is used around the destination. If the absolute values of the pose error terms are within a threshold, the vision is essentially ignored and the last pose estimate is used as the final set-point for the robot.

5 EXPERIMENTAL RESULTS

The experimental set-up consists of a compact camera mounted on the gripper of a PUMA robot arm and a set of white cylinders of various diameters (8 mm, 16 mm, 22 mm, and 38mm). The camera has a 24mm focal length and an imaging plane of 570×485 pixels or 6.39×4.88 mm. There are two VME cages connected via an ethernet. The vision cage consists of an MV-147 and an MV-135 processor and Datacube pipe-lined DSP boards. The robot control cage consists of 5 processors. At CIRSSE, a real-time operating system was developed to simplify robotic research in intelligent control: the CIRSSE Testbed Operating System (CTOS). Under CTOS, there are software hierarchies for each cage. The Vision Services System (VSS) resides on the vision cage and allows simple access to the powerful image processing capabilities of the Datacube. The Motion Control System (MCS) resides on the robot control cage and provides a development platform for multi-arm control and experimental controllers.

The image processing for the visual servoing was written under VSS. A standard 6-joint PID controller is used from the MCS library. The scan method used was thresholding, which was performed by the Datacube at frame rate. The Datacube has triple-ported memory, so frame acquisition can be uninterrupted. A field-complete flag signals the processing of the scan lines, which is performed in parallel by the MV-147. Thus, the vision update rate is 16.7 ms. The pose parameters are calculated and sent to the MCS via internet sockets. Upon receipt, the robot joint angles are read and converted to a transform matrix representation of the robot pose. The matrix is then modified and the inverse-kinematics are performed, giving new joint positions. The joint positions are extrapolated and passed to the robot controller. The PID controller gives torque updates to the PUMA every 4.5 ms. This process continues until the user exits the program. If the cylinder is lost at any time, the scan lines are adjusted until the cylinder is re-acquired. If no cylinder is found in the image, the robot halts and the program waits until one appears.

Approximately 70 trials were made over a variety of conditions. The robot and its own calibration are used to measure the errors. The robot is positioned and a pose estimate is computed. Then the robot is moved slightly and another pose estimate is computed. By comparing the visual change with the change in the position of the robot, the error is computed. The results from the cylinder pose estimation trials and the ranges of the parameter values are shown in Table 1. The minimum error is encountered when the cylinder is oriented vertically and centered in the screen. This is because the camera distortion is at a minimum in the center of the image. This is reflected in Table 2. The distance condition is due to the focal depth of the camera.

Table 1: Average Pose Estimation Error

Parameter	Range	Average Absolute Error
θ	$0 - 90°$	$0.28°$
ϕ	$0 - 45°$	$3.15°$
d_C	$13 - 46$ cm	1.20 mm
x_C	$0 - 10$ cm	0.13 mm

Table 2: Minimum Pose Estimation Error

Parameter	Minimum Absolute Error	Condition
θ	$0.10°$	$\theta = 0$
ϕ	$1.16°$	$\theta = 0, \phi = 0$
d_C	1.02 mm	$d_C = 15$ cm
x_C	0.10 mm	$d_C = 15$ cm, $x_C = 0$

6 SUMMARY

A method for visually guiding a robot gripper to grasp a cylinder has been presented. The pose estimation algorithm computes the cylinder pose relative to the camera using a single camera mounted on the gripper of a robot. Frame-rate updates are calculated to align the gripper with the cylinder so that a minimal reaction force and torque results when the cylinder is grasped. The method scans two parallel lines in the image and processes for high-contrast edges, called critical points. From the four critical points, the three-dimensional position and orientation of the cylinder is estimated. The pose estimates are linearly extrapolated to provide inputs to a PID controller. This process provides a means to refine rough, global information about the NASA truss assembly area obtained by stationary cameras. The vision-guided servoing makes a smooth transition from gross robotic motion to fine robotic motion in the workspace.

7 ACKNOWLEDGMENT

This research is performed at the Center for Intelligent Robotic Systems for Space Exploration at Rensselaer Polytechnic Institute and is supported in part by Grant number NAGW-1333 from NASA.

8 REFERENCES

[1] C. Lee and W. Lin. A hybrid method of visual guiding for eye-in-hand robot. In *European Control Conference*, pages 2524–2529, 1991.

[2] M.J. Korsten and Z. Houkes. Parametric descriptions and estimation, a synergetic approach to resolving shape from shading and motion. In *Third International Conference on Image Processing and its Applications*, pages 5–9. IEE, 1989.

[3] M. Richetin, M. Dhome, and J.T. Lapreste. Inverse perspective transform from zero-curvature curve points application to the localization of some generalized cylinders. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 517–522. IEEE, 1989.

[4] H. Printz. Finding the orientation of a cone or cylinder. In *Proceedings of the IEEE Computer Society Workshop on Computer Vision*, pages 94–99. IEEE, 1987.

[5] J.T. Fedemma and O.R. Mitchell. Vision-guided servoing with feature-based trajectory generation. *IEEE Transactions on Robotics and Automation*, 5(5):691–700, 1989.

[6] J.T. Fedemma and C.S. George Lee. Adaptive image feature prediction and control for visual tracking with a hand-eye coordinated camera. *IEEE Transactions on Systems, Man, and Cybernetics*, 20(5):1172–1183, 1990.

[7] Y. Shirai and H. Inoue. Guiding a robot by visual feedback in assembly tasks. *Pattern Recognition*, 5:99–108, 1973.