Vision-Guided Grasping of a Strut for Truss Structure Assembly

Keith E. Nicewarner  Robert B. Kelley

Electrical, Computer, and Systems Engineering Department
Center for Intelligent Robotic Systems for Space Exploration
Rensselaer Polytechnic Institute
Troy, New York 12180-3590

Abstract

Monocular information from a gripper-mounted camera is used to servo a robot gripper to grasp a strut. The procedure is divided into four phases: learn, recognition, alignment, and approach. In the learn phase, a strut is placed in the gripper and the pose estimate from the camera is used as a servo target. The recognition phase verifies the presence of a cylinder in the camera field of view. The alignment phase processes only selected scan regions in the image. Relative motion information is then used to generate an extrapolated pose-based trajectory to guide the gripper to a certain distance from the strut. The approach phase uses the last pose estimate and "blindly" moves to grasp the strut. By using a simple fiducial stripe on the strut, rapid and robust vision-guided grasping can be achieved.

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 (6 DOF PUMA + 3 DOF linear-track Aronson platform)
- 2 robot grippers equipped with force and cross-fire sensors
- 2 force-torque sensors for each robot wrist
- a pair of cameras mounted on one of the robot grippers
- 2 stationary cameras
- 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 global pose estimates of the 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 and vision-guided insertion of a strut into a node connector.

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 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. 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 im-
age 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 feature 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 PD controller, thus driving the robot gripper to a grasping position a certain distance from the strut. Using the last vision update, the robot then moves “blindly” through the remaining distance and grasps the strut.

2 Vision-Guided Grasping

There are many image processing techniques which give the position and orientation (pose) of various objects. Some methods are developed to process as many sizes and shapes as possible, thereby sacrificing speed for versatility. Other methods are 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 with a circumferential stripe, a general cylinder pose estimation method is required.

There are few pose estimation methods dealing specifically with cylinders [2, 3, 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 5 scan 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 on which to perform visual servoing. It is also important to verify that the object is cylinder-like. The second condition requires a segmentation of the robot design 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 be no 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 observed 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 camera 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. This procedure is typically done only once as a calibration step whenever the operating conditions of the robot change, such as camera parameters, camera location, lighting, or strut design. Since this is not a time-sensitive task, computation restrictions are not necessary for the image processing.

In a typical learn session, the strut is placed in the gripper and the gripper is closed. An image is then snapped from the camera and the strut is located in the image using the recognition algorithm described in the next subsection. The pose is then estimated and saved to a file which is from then on loaded and used as the target pose for the strut.

2.2 Recognition Phase

A simple method for strut recognition uses the fact that the cylindrical struts appear as long, thin rectangles at various orientations in the image plane. This orientation can be found using the first and second moments of inertia of the rectangles. The discrete moments of inertia for a binary image are defined as in Schalkoff [5],

$$m_{pq} = \sum_{i,j \in R} i^p j^q$$

where $R$ is a region (or “blob”) with uniform intensity of “1.” Moments of order higher than 1 are not invariant to translation. Translation of the centroid of the region to the origin gives the translation-invariant, or “central” moments,

$$m_{pq} = \sum_{i,j \in R} (i-\bar{i})^p (j-\bar{j})^q$$

where $\bar{R}$ is the region with centroid at the origin.

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.
where, using Equation 1,

\[ i = \frac{m_{10}}{m_{00}} \quad (3) \]
\[ j = \frac{m_{01}}{m_{00}} \quad (4) \]

The orientation of the principle (or major) axis is found as described by Klaus and Horn [6],

\[ \theta = \frac{1}{2} \tan^{-1} \left( \frac{2\mu_{11}}{\mu_{20} - \mu_{02}} \right) \quad (5) \]

where \( \theta \) is the angle the principle axis of the “blob” makes with the image plane X-axis.

Now that we have the orientation, we need some measure of the “long-and-thinness” of a blob. With further normalization of 2, scale invariant moments can be obtained,

\[ \eta_{pq} = \frac{\mu_{pq}}{(\mu_{00})^{1+q}} \quad (6) \]

Schalkoff lists 7 moment “features” which are invariant to rotation, scale, and translation. The first two have at most second-order terms,

\[ \phi_1 = \eta_{20} + \eta_{02} \quad (7) \]
\[ \phi_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \quad (8) \]

From statics, we will recognize \( \phi_2 \) as the square of the radius of the Mohr circle made by the moments and \( \phi_1 \) is related to the center of the Mohr circle. The “long-and-thinness” of a blob is given by the ratio of the major second moment to the minor second moment,

\[ \frac{\phi_1 + \sqrt{\phi_2}}{\phi_1 - \sqrt{\phi_2}} \quad (9) \]

The actual width and length of the strut projection rectangle can be found by “walking” along the major and minor axes. These parameters can also be used to verify the shape of the area.

Once the long and thin blobs are extracted, collinear blobs are merged together because the fiducial circumferential stripes effectively split a strut into a group of collinear cylinders. The merges are then noted as candidate marker locations, to be later verified.

If no valid struts result from this, the program fails because there are no struts it can see to be grasped. If there is more than one strut in the image, the program fails as well because there is no criterion to choose an appropriate strut to grasp. The program only continues if there is one valid strut in the image.

The information so far can be used to crudely center and align the strut vertically in the image. As stated before, vertical alignment is necessary for the pose estimation algorithm. This rough alignment is done simply by calculating the delta movement in the image plane for the marker and strut axis using the information given by the strut recognition routine.

The next verification made is that the radius of the strut is within an expected range. The radius of the strut can be estimated by observing the change in the image induced by moving the robot a certain distance towards the strut. 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.

\[ \frac{R}{d_1} = \frac{r_1}{f} \quad (10) \]
\[ \frac{R}{d_2} = \frac{r_2}{f} \quad (11) \]

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

\[ R = \frac{\Delta d}{f} \left( \frac{r_2 r_1}{r_2 - r_1} \right) \quad (12) \]

Therefore, we can use the calibration of the robot to move a given distance and calculate an estimate of the strut’s radius. If this strut is outside of an expected range, the program fails because the object most likely is a bogus object.

Once the strut has been verified, the critical processing areas of the image are chosen. These critical areas are called scan lines and are processed with a 1-D edge detector for rapid pose estimation. Five scan lines comprise a scan region (see Figure 2). These scan lines are either vertically or horizontally oriented to provide fast access to the critical areas of an image by a computer. Two scan horizontally along the top and bottom of the screen for edges. Two more scan vertically across the top and bottom edges to detect the end of the strut. The last scan line vertically crosses the fiducial marker (if present).

The scan line positions and scan ranges are chosen to minimize the noise that might be encountered during the alignment phase. The top and bottom horizontal scan lines are chosen to be as far apart as possible to ensure more accurate pose estimations. The top and bottom cross scans are used to ensure that the top and bottom horizontal scans are sufficiently far from the end of a strut (if visible).

2.3 Alignment Phase

The alignment phase begins by rapidly processing the scan lines for edges, or critical points. There are five critical points: 2 on the top horizontal scan line, 2 on the bottom horizontal scan line, and 1 representing the mid-point of the edges across the fiducial stripe. 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 5 of the 6 strut pose parameters can be determined from only 5 critical points. The pose of the strut is computed relative to the camera. The 5 pose
parameters are:

1. $R_x$ - the tilt angle of the strut axis out of a plane perpendicular to the optical axis.
2. $R_z$ - the clockwise rotation of the strut about the optical axis, relative to the image plane y-axis.
3. $T_x$ - the horizontal displacement of either the strut marker or the center of the strut from a vertical plane through the optical axis.
4. $T_y$ - the vertical displacement of the strut marker (if visible) from a horizontal plane through the optical axis.
5. $T_z$ - the distance from the camera lens to the center of the strut along the optical axis.

Note that $R_y$ is not available since the strut is rotationally symmetric. $T_y$ is only available if a stripe is visible; otherwise, only four parameters are used. Effectively, if no stripe is seen, the strut will be grasped arbitrarily along the axis.

For the alignment phase, the robot controller serves all the parameters except $T_z$ to zero. The distance is servoed to an optimal distance from the strut. This distance is determined primarily by the focal depth and field of view of the camera.

2.4 Approach Phase

Ideally, the alignment phase could be continued all the way to the target pose. As discussed in [7], the pose estimates get better as we get closer, so we should expect our best performance as we are grasping the strut. In reality, although the pose estimate errors do indeed decrease as the distance decreases, the sensitivity of the critical point extraction process increases. As the image becomes larger in the image, unavoidable minute "jerks" in the robot's movements can cause the feature extraction process to fail.

To solve this problem, the visual servo process halts when the last pose estimate is the "best." From there, the robot moves "blindly" to grasp the strut. Weighing the relative costs of completely servoing versus the loss in fault tolerance introduced by blind motion is discussed in [7].

3 Derivation of the Pose Estimate

Visual servoing requires rapid feature extraction. As mentioned before, the features for our case are the critical points in the image. These critical points are found as edges along scan lines. Five scan lines compose a scan region. Figure 2 shows a scan region in a typical scene. Note the top and bottom edge scan lines are present only to place the top and bottom horizontal scan lines away from the end of a strut, if visible. Normally, an end of the strut is not visible, but this provides both robustness and future expansion to handle the short cylindrical strut connectors.

Critical points are found by performing one-dimensional edge-detection along each scan line. 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. 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. With this method distant clutter is usually out of focus and is thereby ignored.

Once the critical points are obtained, a rapid method for estimating the strut pose is needed. We will now discuss the derivation of this pose estimate. Consider a $W \times H$ image of a scene containing a strut as in Figure 3. The top and bottom edge scan lines have been omitted for clarity. The critical points are designated as $x_{TL}$, $x_{TR}$, $x_{BL}$, $x_{BR}$, and $m$. Note the critical point $m$ is an $(x, y)$ location in the image plane, and represents the mid-point of the two edges found along the marker scan line.

Using the critical points, the orientation of the axis of the strut is computed using geometry. Due to perspective distortion, the image-plane axis of the strut and the projection of the actual axis do not necessarily coincide. However, if the critical points make an angle of less than about 10° with the optical axis, then the perspective distortion effects are negligible. Thus, the centerline of the strut image is a good approximation to the projection of the strut 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 $x_{BC}$. 

![Figure 2: Scan lines composing a typical scan region.](image-url)
Thus,

\[ x_{TC} = \frac{1}{2}(x_{TL} + x_{TR}) \]  
\[ x_{BC} = \frac{1}{2}(x_{BL} + x_{BR}) \]  

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

\[ R_z = \tan^{-1}\frac{x_{TC} - x_{BC}}{H} \]  

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

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

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

\[ \rho_T = (x_{TR} - x_{TC}) \cos R_z \]  

and similarly at the bottom of the image,

\[ \rho_B = (x_{BR} - x_{BC}) \cos R_z \]  

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

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

The tilt angle \( R_x \) is the angle the strut is rotated towards the viewer relative to a plane perpendicular to the optical axis. The tilt angle geometry is shown in Figure 4. Assuming a constant optical magnification factor \( \mu \) (i.e., \( T_x \gg f \)), the following expressions are obtained:

\[ \mu = \frac{R}{\rho_T} = \frac{T_x - l \sin R_x}{f} \]  
\[ \mu = \frac{R}{\rho_B} = \frac{T_x + (L - l) \sin R_x}{f} \]  
\[ \mu = \frac{R}{\rho_C} = \frac{T_x}{f} \]  

where \( R \) is the actual radius of the strut, \( L \) is the length of the visible portion of the strut, \( l \) 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:

\[ \frac{H}{2f} = \frac{f l \cos R_x}{\rho_T R} \]  
\[ = \frac{f (L - l) \cos R_x}{\rho_B R} \]  

where \( H \) is the distance between the scan lines. Simplifying and solving for \( R_x \),

\[ R_x = \tan^{-1}\left( \frac{2f}{H} \left( \frac{\rho_T - \rho_B}{\rho_T + \rho_B} \right) \right) \]  

The distance to the center of the strut is given by similar triangles,

\[ T_x = \frac{Rf}{\rho_C} \]  

And finally, the displacement parameters of the fiducial marker, \( T_x \) and \( T_y \), are given by

\[ T_x = \frac{T_x}{f} (m_x - \text{MIDX}) \]  
\[ T_y = \frac{T_y}{f} (m_y - \text{MIDY}) \]  

where \( m_x \) and \( m_y \) are the image-location of the fiducial mark and \( \text{MIDX} \) and \( \text{MIDY} \) are the mid-points of the image (or more precisely, the intersection of the optical axis with the image plane). If no fiducial marker is present, then only \( T_x \) can be estimated using the center of the strut in the image as a reference,

\[ T_x = \frac{Rf}{f} (\xi_C - \text{MIDX}) \]  

In summary, the orientation angles \( R_x \) and \( R_z \) are determined from these expressions without knowledge of the actual radius of the strut. However, the position parameters \( T_x \), \( T_y \), and \( T_z \) of the strut are expressed in terms of the actual radius of the strut. Therefore, with only two parallel scan lines from the image and
The calculation of this pose estimate involves relatively few math operations compared to other methods. Therefore, the calculation is much faster. On our system, running on a 68030 under VxWorks, the calculations take less than 2 ms.

4 Motion 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 [8, 9], as opposed to the traditional look-and-move approach [10]. This allows for slight inaccuracies in the pose estimate due to image noise, quantization errors, lens distortion, and camera parameter uncertainties, because the estimate is constantly being refined.

A block diagram of the closed-loop system is shown in Figure 5. The desired strut pose is obtained in the learn phase discussed previously. The current camera pose is the homogeneous transform \( T_0 \) which is computed by using the forward-kinematic transform on the current joint angles of the robot, the vector \( q \), to obtain the end-effector transform,

\[
T_0 = K(q)
\]

The camera transform is calculated by post-multiplying the end-effector transform by the end-effector-to-camera transform,

\[
T_0 = T_0' T_e^-
\]

In our case, \( T_e^\) need only be roughly approximated.

\( T_0' \) is computed only once from the joint angles at the beginning of a servo session. From then on, the transform is modified to reflect the intended motion of the robot. This is an important point because we would otherwise compute the forward kinematics each cycle. In an ideal world, there would be no difference, but in actuality, the difference is significant.

If the robot has not reached its objective by the next sampling time, its joints would be read and the camera transforms would be applied to the current end-effector transform. The result would be a "wandering" of the robot from the intended motion, making servoing difficult. By keeping track of the desired robot motion, the wandering is kept to a minimum.

For each vision update, the pose parameters are computed and transmitted to the robot controller. The transmission delay is modeled because in our case, there is a delay in transmitting the pose information from the sensor 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 pose parameters are then subtracted from the desired pose parameters, giving a set of error terms:

\[
\Delta T_x = T_{xd} - T_x
\]

\[
\Delta T_y = T_{yd} - T_y
\]

\[
\Delta T_z = T_{zd} - T_z
\]

\[
\Delta R_x = R_{xd} - R_x
\]

\[
\Delta R_y = R_{yd} - R_y
\]

Note that the rotation errors are treated as linear errors. While this linearization is valid for small errors, it might not be for large errors, in which case a quaternion approach might be used. In our case, the errors are generally small and no such approach was needed.

The pose parameter errors are used to modify the current pose of the camera, \( T_0' \). The robot pose modifications are a set of cartesian transformations on \( T_0' \):

\[
T_{0,new} = T_0' T_x (-\Delta T_x) T_y (-\Delta T_y) T_z (-\Delta T_z)
\]

\[
R_x (-\Delta R_x) R_y (-\Delta R_y)
\]

where \( R_i \) is the rotation transformation matrix about axis \( i \) and \( T_i \) is the translation transformation matrix along axis \( i \). The order of the transformations is important; the translations must be done first, followed by the rotations. Stated more simply,

\[
T_{0,new} = T_0' \Delta T_e
\]

The new joint angles are then computed by the inverse-kinematic transform,

\[
q_{new} = K^{-1}(T_{0,new})
\]

The new desired joint angles are then linearly extrapolated by the trajectory generator and sent to the joint controller. The linear extrapolation can be modeled as a 1st-order-heap (FOH).

The generated position and velocity set-points are fed into a standard PD joint-space controller which has been "tuned" to 5.4 ms. Because there is a finite amount of precision in the vision sensing, robot position sensing, and robot control torques, oscillations would occur if we were to use the raw vision estimates. 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 vision-guided grasping systems discussed in this paper was successfully implemented with the CIRSSE experimental testbed. Several experiments were performed to evaluate the performance of the vision-guided grasping system presented in this paper. In one experiment, white cylinders of various diameters (8 mm, 16 mm, 22 mm, and 38 mm) were used to test the pose estimation process with respect to the robot calibration.

Approximately 70 trials were made over a variety of conditions. The robot and its own calibration was used to measure the errors. The robot was positioned and a pose estimate was computed. Then the robot was moved slightly and another pose estimate was computed. By comparing the visual change with the change in the position of the robot, the error was 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, as
Figure 4: Projection geometry of a cylinder segment.

Figure 5: Block diagram of the visual servo system.
Table 1: Average measured pose estimation error

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Range</th>
<th>Absolute Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R_z$</td>
<td>0 - 90°</td>
<td>0.28°</td>
</tr>
<tr>
<td>$R_z$</td>
<td>0 - 45°</td>
<td>3.15°</td>
</tr>
<tr>
<td>$T_z$</td>
<td>13 - 46 cm</td>
<td>1.20 mm</td>
</tr>
<tr>
<td>$T_z$</td>
<td>0 - 10 cm</td>
<td>0.13 mm</td>
</tr>
</tbody>
</table>

Table 2: Minimum measured pose estimation error

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Absolute Error</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R_z$</td>
<td>0.10°</td>
<td>$R_z = 0$</td>
</tr>
<tr>
<td>$R_z$</td>
<td>1.16°</td>
<td>$R_z = 0, R_z = 0$</td>
</tr>
<tr>
<td>$T_z$</td>
<td>1.02 mm</td>
<td>$T_z = 15$ cm</td>
</tr>
<tr>
<td>$T_z$</td>
<td>0.10 mm</td>
<td>$T_z = 15$ cm, $T_z = 0$</td>
</tr>
</tbody>
</table>

discussed in [7]. This is reflected in Table 2. The distance condition is due to the focal depth of the camera.

Another set of experiments performed involved finding and grasping a strut, moving to “home” position, then placing the strut randomly and repeating the process. This is perhaps the best measure of performance for our system because it conveys the reliability and repeatability of the process. With the completed system, around 100 trials were made. All were handled properly, meaning that if the strut was not visible in the starting image, the program exited and if the strut was visible, it was successfully grasped.

6 Summary

A successful method for visually guiding a robot gripper to grasp a cylindrical strut has been presented. The pose estimation algorithm computes the strut 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 strut so that a minimal reaction force and torque results when the strut is grasped. The method processes five scan lines in the image and processes for high-contrast edges, called critical points. From the five critical points, the three-dimensional position and orientation of the strut is estimated. The pose estimates are linearly extrapolated to provide inputs into a PD controller. This process provides a means to refine rough, global information of the 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.

Acknowledgment

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

References


