HAND-EYE COORDINATION FOR GRASPING MOVING OBJECTS

Peter Allen
Billibon Yoshimi
Alex Timcenko
Paul Michelman

Department of Computer Science
Columbia University
500 W. 120th Street
New York, NY 10027

ABSTRACT

Most robotic grasping tasks assume a stationary or fixed object. In this paper, we explore the requirements for grasping a moving object. This task requires proper coordination between at least 3 separate subsystems: dynamic vision sensing, real-time arm control, and grasp control. As with humans, our system first visually tracks the object's 3-D position. Because the object is in motion, this must be done in a dynamic manner to coordinate the motion of the robotic arm as it tracks the object. The dynamic vision system is used to feed a real-time arm control algorithm that plans a trajectory. The arm control algorithm is implemented in two steps: 1) filtering and prediction, and 2) kinematic transformation computation. Once the trajectory of the object is tracked, the hand must intercept the object to actually grasp it. We present 3 different strategies for intercepting the object and results from the tracking algorithm.

1. INTRODUCTION

The focus of our work is to achieve a high level of interaction between a real-time vision system that is capable of tracking moving objects in 3-D and a robot arm that contains a dexterous hand that can be used to intercept, grasp and pick up a moving object. We are interested in exploring the interplay of hand-eye coordination for dynamic grasping tasks such as grasping of parts on a moving conveyor system, assembly of articulated parts or for grasping from a mobile robotic system. However, the algorithms we have developed are quite general, and applicable to a wider range of domains including the ones described above.

The system we have built addresses three distinct problems in real-time, dynamic grasping of moving objects: fast computation of 3-D motion parameters, grasp planning which entails coordination with a dexterous robotic hand and predictive control of a moving robotic arm to track a moving object. The system is able to operate in real-time, at approximately human arm movement rates, using visual feedback in an active sense for a dynamically changing (as opposed to static) environment.

The system we are building consists of two statically mounted cameras that can image a scene containing a moving object (see Figure 1). A PUMA-560 with a Utah-MIT dexterous hand attached is used to track the object with the eventual goal of stably grasping the object as it moves. The system operates as follows:

1. The imaging system performs a stereoscopic optic-flow triangulation at real-time rates.
2. The 3-D position of the moving object computed by step 1 is used to command the robotic arm to track the moving object's position
3. Once tracking is stable, the system command the arm to intercept the moving object
4. The hand is used to stably grasp the object once it is intercepted.

The focus of this paper is on steps 1,3, and 4. A related paper [3] discusses step 2 in more detail.

This work was supported in part by DARPA contract N00039-84-C-0165, NSF grants DMC-86-05065, DCI-86-08845, CCR-86-12709, IRI-86-57151, IRI-88-1319, North American Philips Laboratories, Siemens Corporation and Rockwell Inc.

2. VISION SYSTEM

While the vision system comprises only one part of our system, visual motion tracking has been the subject of much past research. In this section, we describe related work, explain our use of optic-flow for real-time motion detection, and describe our parallel-pipelined algorithm for computing optic-flow in real-time.

2.1. Related Work

Previous efforts in the areas of visual motion tracking are too numerous to exhaustively list here. We instead list some notable efforts that have inspired us or use similar approaches. Burt et al. [7] has focused on high speed feature detection and hierarchical scaling of images in order to meet the real-time demands of surveillance and other robotic applications. Related work has been reported by Lee and Wohn [22] and Wiklund and Granlund [36] who use image differencing methods to track motion. Corke, Paul and Wohn [11] report a feature based tracking method that uses special purpose hardware to drive a servo-controller of an arm-mounted camera. Goldenberg et al [12] have developed a method that uses temporal filtering with similar hardware to our own. Luo, Mullen and Wessel [23] report a real-time implementation of motion tracking in 1-D based on Horn and Schunck's method. Verghese et al. [34] report real-time, short-range visual tracking of objects using a pipelined system similar to our own. Safadi [30] uses a tracking filter similar to our own and a pyramid based vision system, but few results are reported with this system. Rao and Durrant-Whyte [29] have implemented a Kalman filter based de-centralized tracking system that tracks moving objects with multiple cameras. Miller [26] has integrated a camera and arm for a tracking task where the emphasis is on learning kinematic and control parameters of the system. Weiss et al [35] also use visual feedback to develop control laws for manipulation. Brown [6] has implemented a gaze control system that links a robotic "head" containing binocular cameras with a servo controller that allows one to maintain a fixed gaze on a moving object. Clark and Ferrier [10] also have implemented a gaze control system for a mobile robot.

2.2. Computing Optic-Flow

In a visual tracking problem, motion in the imaging system has to be translated into 3-D scene motion. Our approach is to initially compute local optic-flow fields that measure image velocity at each pixel in the image. A variety of techniques for computing optic-flow fields have been used with varying results including matching based techniques [4, 8, 32], gradient based techniques [9, 18, 27] and spatio-temporal energy methods [1, 14]. Optic-flow was chosen as the primitive upon which to base the tracking algorithm for the following reasons:

- The ability to track an object in three dimensions implies that there will be motion across the retinas (image planes) that are imaging the scene. By identifying this motion in each camera, we can begin to find the actual 3-D motion.
- The principal constraint in the imaging process is high computational speed to satisfy the update process for the robotic arm parameters. Hence, we needed to be able to compute image motion quickly and robustly. The Horn-Schunck optic-flow algorithm (described below) is well suited for real-time computation on our PIPE image processing engine.
- We have developed a new framework for computing optic-flow robustly using an estimation-theoretic framework [33]. While this work does not specifically use these ideas, we have future plans to try to adapt this algorithm to such a framework.

Our method begins with an implementation of the Horn-Schunck method of computing optic-flow [17]. The underlying assumption of this method is the optic-flow constraint equation, which assumes image irradiance at time t and $t+\delta t$ will be the same:

$$I(x+\delta x, y+\delta y, t+\delta t) = I(x, y, t). \tag{1}$$

If we expand this constraint via a Taylor series expansion, and drop second and higher order terms, we obtain the form of the constraint we need to compute normal velocity

$$I_x u + I_y v + I_t = 0 \tag{2}$$

where u and v are the velocities in image-space, and I_x, I_y and I_t are the spatial and temporal derivatives in the image. This constraint limits the velocity field in an image to lie on a straight line in velocity space. The actual velocity cannot be determined directly from this constraint due to the aperture problem, but one can recover the component of velocity normal to this constraint line as:

$$V_n = -\frac{I_t}{\sqrt{I_x^2 + I_y^2}} \tag{3}$$

While computationally appealing, this method of determining optic-flow has some inherent problems. First, the computation is done on a pixel by pixel basis, creating a large computational demand. Second, the information on optic flow is only available in areas where the gradients defined above exist. A second, iterative process is usually employed to propagate velocities in image neighborhoods, based upon a variety of smoothness and heuristic constraints.

We have overcome the first of these problems by using the PIPE image processor [5,20]. The PIPE is a pipelined computer capable of processing 256x256x8 bit images at frame rate speeds, and it supports the operations necessary for optic-flow computation in a pixel-parallel method (a typical image operation such as convolution, warping, addition/subtraction of images can be done in one cycle - 1/60 second). The second problem is alleviated by our not needing to know the actual velocities in the image. What we need is the ability to locate and quantify gross image motion robustly. This rules out simple differencing methods which are too prone to noise and will make location of image movement difficult. Hence, a set of normal velocities at strong gradients is adequate for our task, precluding the smoothing step of the algorithm.

2.3. A Real-Time Optic-Flow Algorithm

Our goal is to track a single moving object in real-time. We are using 2 static cameras that image the scene and need to report motion in 3-D to a robotic arm control program. Each camera is calibrated with the 3-D scene, but there is no explicit need to use registered (i.e scan-line coherence) cameras. Our method computes optic-flow fields in each camera and then use a triangulation to intersect the flow fields in areas of image motion in each camera. To implement our algorithm in the PIPE, we used 4 processors on the PIPE. The processors are assigned as 2 per camera - one each for the calculation of X and Y motion energy centroids in each image. We also use a special processor board (ISMAP) to perform real-time histogramming. The steps below correspond to the numbers in Figure 2 (single camera):

1. The camera images the scene and the image is sent to processing stages in the PIPE.
2. The image is smoothed by convolution with a Gaussian mask. The convolution operator is a built in operation in the PIPE and it can be performed in one frame cycle.

3-4. In the next 2 cycles, two more images are read in, smoothed and buffered, yielding smoothed images I_0 and I_1 and I_2. The ability to buffer and pipeline images allows temporal operations on images, albeit at the cost of processing delays (lags) on output. There are now 3 smoothed images in the PIPE, with the oldest image lagging by 3/60 second.

5. Images I_0 and I_2 are subtracted yielding the temporal derivative I_t.
6. In parallel with step 5, Image I_1 is convolved with a 3x3 horizontal spatial gradient operator, returning the discrete form of I_x. In parallel, the vertical spatial gradient is calculated yielding I_y (not shown).

7-8. The results from steps 5 and 6 are held in buffers and then are input to a look-up table that divides the temporal gradient at each pixel by the absolute value of the summed horizontal and vertical spatial gradients. This yields the normal velocity in the image at each pixel.

9-10. In order to get the centroid of the motion information, we need the X and Y coordinates of the motion energy. For simplicity sake we show only the situation for the X coordinate. The gray-value ramp in Figure 2 encodes the horizontal coordinate value (0-255) for each point in the image. If we threshold the computed normal velocities, and then AND the above threshold velocities with the positional ramp, we have an image which encodes high velocity with its positional coordinates in the image. In our experiments, we thresholded all velocities below 10 pixels per 60 msec. to zero velocity.

11. By taking this result and histogramming it, via a special stage of the PIPE which performs histograms at frame rate speeds, we can find the centroid of the moving object by finding the mean of the resulting histogram. Histogramming the high velocity position encoded images yields 256 16-bit values (a result for each intensity in the image). These 256 values can be read off the PIPE via a parallel interface in about 10 ms. This operation is performed in parallel to find the moving objects Y centroid (and in parallel for X and Y centroids for camera 2). The total associated delay time for finding the centroid of a moving object becomes 15 cycles or 0.25 seconds.

The same algorithm is run in parallel on the PIPE for the second camera. Once the motion centroids are known for each camera, they are back-projected into the scene using the camera calibration matrices and triangulated to find the actual 3-D location of the movement. Because of the pipelined nature of the PIPE, a new X or Y coordinate is produced every 1/60 second with this delay.

The system exhibits an interesting mix of local and global computations, separated by processors and update rates. The PIPE is able to perform the local optic-flow computation at video rates (but with delay), the ISMAP board gathers global histogram statistics, and these are then shipped via a high-speed interface to a host where the stereo triangulation and kinematic control algorithms reside, updating the arm parameters every 30 msec.

3. ROBOTIC ARM CONTROL

The second part of the system is the arm control. The robotic arm has to be controlled in real-time to follow the motion of the object, using the output of the vision system. The vision system output is not sufficient as a control parameter since its output is both noisy as well as delayed in time. The control system needs to do the following:

- Filter out the noise with a digital filter
- Predict the position to cope with delays introduced by both vision subsystem and the digital filter
- Perform the kinematic transformations which will map the desired manipulator's tip position from a Cartesian coordinate frame into joint coordinates, and actually perform the movement

We have adopted a simple, modular solution for the control subsystem where each task performs its job independently from the others. This approach, although suboptimal, exhibits a high degree of modularity and, nonetheless, simplicity; characteristics which make a complex robotic system more robust and extensible.

The 3-D values of the moving object's position determined by the vision system are both noisy and out of date due to the processing delays. To alleviate this, the system has been modeled as a standard second order system, and the input to the system is a series of filtered predictions (see [3] for details). We are using RCCL [13] to control the robotic arm (a PUMA 560). RCCL (Robot Control C Language) allows the use of C programming constructs to control the robot as well as defining transformation equations (as described in [28]). The transformation equations permit dynamic updating of arm position by generating the 4×4 transform of the moving object's position from the vision system and sending this information to the arm control algorithm (see Figure 3).

4. VISUAL-MOTOR COORDINATION

In developing a robotic hand-eye system that is capable of tracking and grasping a moving object, we have examined the psychological literature. This problem has been examined by many researchers in psychology who have tried to understand human visual-motor coordination and develop theories of human control mechanisms. In discussing the relationship between vision and motor control, we hope to find useful paradigms for robotics research.

There are several theories on the organization of skilled human motor control. Richard Schmidt [31] has proposed a theory of generalized motor programs, or movement *schemas*. In this view, a skilled action is composed of an ordered set of parametrized motor control programs of short duration (less than 200 msec), each of which accomplishes one part of the task. As one program is completed, the next one is executed. Generalized motor programs accomplish several objectives: (1) they specify *which muscle* to move in a given motion; (2) the *order of contraction* of the muscles; (3) the *phasing* within the sequence, i.e., the temporal relationships among the contractions; (4) the *relative force* of each element. At the initiation of a skilled task, the parameters of the motor control program are determined by sensory input and task demands, and then the programs are executed to completion. If the wrong program is selected for some reason, the program cannot be stopped by use of sensory information. Similarly, in playing table tennis, the motion of the racket is determined before the beginning of the swing and visual input has little effect after the initiation of motion. As an example of Schmidt's theory, the skilled task of grasping a moving object could be partitioned into

two motor control schemas: one to position the arm and a second one to control the grasping action.

The schema concept maps into Von Hofsten's ideas about the development of grasping skills in children [16]. He believes there are two separate sensorimotor systems responsible for reaching: one for approaching the target and one for grasping it. During early childhood, the precise timing between these two systems develops as the child learns how to catch. The reaching system develops first, before a child is capable of grasping. But even before he is capable of closing his hand at precisely the right moment, he has begun to develop the ability to move his hand toward a moving object and predict the location at which his hand will intercept the object. With growth, a child learns to control the timing between reaching and grasping, that is, to close his hand at the correct moment. Experimental evidence has shown that there is a window of approximately 14 msec during which the hand must begin closing. Unlike Schmidt, however, Von Hofsten does not consider vision and grasping to be two mutually exclusive tasks [15]. Visual tracking is used to guide the reaching arm *during* its motion, not only before motion. A coordinated motion is a combination of perceptual schemas and motor schemas (see Iberall and Arbib [19]).

Vision is used during the reaching phase of the task for what psychologists call "prospective control". Prospective control corresponds to predictive filtering, as used by control theorists. In grasping a moving object, it is necessary for the hand to move not to the current position of the object, but to plan ahead to where it will be shortly. Vision, rather than haptics, provides the basis of prospective control because touch cannot provide the anticipatory information required to predict the course of a moving object. There are two predominant theories about what visual schema is used to track a moving object and aid in predicting the intersection of the reaching hand and that object. Lee [21] proposes the use of vision to measure the expansion of the image on the retina in order to estimate the time until contact. The attraction of this theory is that humans would not need to compute the velocity and location of the moving object, but would calculate the more useful time-until-contact information. A person catching an object uses this image to compute when to begin the correct motion commands (usually at about 300 msec before the actual grasp). Von Hofsten disputes the use of retinal expansion information because it is clear that people are able to track targets in which there is no such expansion, such as objects that are circling or passing across the field of view. He suggested an alternative schema in which people calculate the distance to a moving object by using the vergence angle to the object. Vision seems to be used predominantly to track the moving object, but the catcher also tracks his hand during reaching to aid his nonvisual proprioceptive senses, that is, to help judge the position of his hand in relation to the environment. Finally, vision must be used during the reaching phase to orient the hand correctly in relation to the object that is being caught.

We have identified three possible control strategies for the grasping part of the system. The first is analogous to the motor control schemas of Schmidt, in which catching a moving object is considered a skill of short duration -- vision is used only to predict the location of the object, but not to control the reaching and grasping portion of the task. In the second strategy, we make use of visual information during both the object tracking and reaching portions of the task. The third strategy is similar to the second, with the exception that perceptual information is used until the final grasp.

5. STRATEGIES FOR INTERCEPTING THE MOVING OBJECT

As outlined above, we first need to bring the hand into proximity of the moving object before we can effect an actual object grasp. We refer to this phase of the task as interception. Currently, we are implementing strategy 1, but we are also exploring strategies 2 and 3 in parallel.

5.1. Strategy 1

The first strategy is an open loop "snatching" strategy, similar in spirit to the pre-programmed motor control schemas described in the psychological literature. Schmidt's schema theory holds that for tasks of short duration, perception is used to find a set of parameters to pass to a motor control program. It is not used during the execution of a task. When grasping a moving object, for example, once vision determined the trajectory of the object, the reach and grasping motor schemas take over with no interference from vision.

In our implementation of this strategy, vision is not used to continually monitor the grasping, but only to provide a final position and velocity from which the arm is directed to very quickly move to the object. This automatic movement is done by establishing coordinate frames of action for each of the components of the system and solving transformation equations (see Figure 3). The transformation equations permit dynamic updating of the arm position by generating the 4×4 transform of the moving object's position from the vision system and sending this information to the arm

control algorithm. This positional information from the vision system is used to update the **Obj** transform in figure 3. The other transforms in the equation are known, and this allows the system to solve for the **Drive** transform which is the transform used to update the manipulator's joints and develop a straight line path in Cartesian coordinates that will bring the hand into contact with the moving object. Because the movement of the hand requires a small amount of time during which the object may have moved, the object's trajectory is predicted by linearly interpolating its current trajectory slightly ahead. This linearization is simple, and unless the object makes a radical maneuver during grasping, will accurately bring the hand into contact with the object. By keeping the fingers of the hand spread during this maneuver, no actual contact takes place unless the fingers are commanded to close (assuming the object is small in comparison to the hand).

5.2. Strategy 2

A more complicated, but potentially more accurate strategy, is to visually monitor the interception and use this visual information to update the **Drive** transform at video update rates. This approach is computationally more demanding, requiring multiple moving object tracking capability. The initial vision tracking described in section 2 is capable of single object tracking only. If we attempt to visually servo the moving robotic arm with the moving object, we have introduced multiple moving objects into the scene.

We have identified 2 possible approaches to tracking these multiple objects visually. The first is to use the PIPE's region of interest operator that can effectively "window" the visual field and compute different motion energies in each window concurrently. Each region can be assigned to a different stage of the PIPE and compute its result independently. This approach assumes that the moving objects can be segmented. This is possible since the motion of the hand in 3-D is known - we have commanded it ourselves. Therefore, since we know the camera parameters and 3-D position of the hand, it will be possible to find the relevant image-space coordinates that correspond to the 3-D position of the hand. Once these are known, we can form a window centered on this position in the PIPE, and concurrently compute motion energy of the moving object and the moving hand in each camera. Each of these motion centroids can then be triangulated to find the effective positions of both the hand and object and compute the new **Drive** transform. Both computations must, however, compete for the hardware histogramming capability needed for centroid computation, and this will effectively reduce the bandwidth of position updating by a factor of 2.

A more appealing approach to us, both practically and scientifically, is to take advantage of the recent work of Bergen et al [7]. Using a simple perspective imaging model, we can compute the spatial change in image coordinates over time knowing the spatial change in 3-D world coordinates over time. Given a point in the world at time $t=0$ represented by X_0, Y_0, Z_0, its positional change over time (given constant component velocities u, v, w) is:

$$\begin{bmatrix} X_0(t) \\ Y_0(t) \\ Z_0(t) \end{bmatrix} = \begin{bmatrix} X_0 + ut \\ Y_0 + vt \\ Z_0 + wt \end{bmatrix} \tag{4}$$

If we use a standard perspective projection model where the image coordinates are x_i , y_i, the camera focal length is f, and we assume $Z \gg f$, we get:

$$\begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} \frac{fX_0}{Z_0} \\ \frac{fY_0}{Z_0} \end{bmatrix} \tag{5}$$

which yields, as a function of time:

$$\begin{bmatrix} x_i(t) \\ y_i(t) \end{bmatrix} = \begin{bmatrix} \frac{f(X_0 + ut)}{Z_0 + wt} \\ \frac{f(Y_0 + vt)}{Z_0 + wt} \end{bmatrix} \tag{6}$$

Equation 6 allows us to compute image space position as a function of time, given known 3-D velocities. Therefore, multiple motions in an image are tracked by effectively nulling out the known the motion. In the case of translating objects, we can shift an image from time t_i to t_{i+1} by the amount of pixel motion computed by (6) and then use a simple differencing method that will result in an image containing the remaining component of object motion. Since we know the 3-D motion of the robotic hand as it approaches the object (we are commanding it), we can effectively compute its optic-flow field via camera projections, and then use these velocities to null out the hand's motion as we track the moving object. Since this will require extra cycles on the PIPE to perform the image shifts, we may reduce our bandwidth in updating the moving object's position.

5.3. Strategy 3

The third approach is to use a coarse-fine hierarchical control system that uses a multi-sensor approach. As we approach the object for grasping, we can shift the visual attention from the static cameras used in 3-D triangulation to a single camera mounted on the wrist of the robotic hand. Once we have determined that the moving object is in the field of view of this camera, we can use its estimates of motion via optic-flow to keep the object to grasped in the center of the wrist camera's field of view. This control information will be used to compute the **Drive** transform to correctly move the hand to intercept the object. We have implemented such a tracking system with a different robotic system[2] and can adapt this method to this particular task.

6. HAND-OBJECT CONTACT

We first note a relevant fact for human contact and grasping of objects. The central factor to the final grasp is the time of the onset of hand closure. In early childhood (up to about 5 months), closing the hand is triggered primarily by touch. Children tend to begin grasping only when they are already in contact with the object. By the time a child is 13 months old, however, the hand closes before touch, on average as early as for adults. We take the view below that our robotic system is past early childhood - we will close the hand before actual contact is made.

The grasping portion of the task is performed as follows. The hand is initially preshaped with its four fingers extended. As soon as the arm completes its movement toward the moving object, grasping begins. Timing is critical here because the object continues to move. Therefore, the fingers close at high speed. Because the size and orientation of the moving object are not known with any accuracy, it is not possible to plan and execute a grasp. Although it seems improbable that an object can be stably grasped without precise planning, a moment's reflection will reveal that a rough knowledge of object size and orientation are all that are required. People are able to catch objects such as rotating rectangular blocks by trying to align their hands perpendicular to the axis of rotation and precisely timing the closing of the fingers. Because we are grasping a symmetrical moving object, the orientation is of less importance than the timing of the grasp.

In our experiments, we plan to use the same finger trajectory for each grasp: a three-fingered grasp, with the Utah/MIT hand's thumb and the two fingers that oppose it. The arm is moved to a position such that a line drawn between the thumb and the opposing second and third fingers is directly above the moving object. At the moment the arm ends its motion, the arm signals the hand to begin grasping. Because of the compliance of the hand, the fingers stop moving when they are in contact with a rigid object, so it is not necessary to know the object's geometry or position precisely at the moment of grasping.

After the fingers close around the object, the forces against the fingertips are monitored to ascertain whether or not the object is actually grasped. One technique to detect grasp stability is first to note that each of the contacts exerts a set of wrenches on the grasped object. If, for example, two fingers stably grasp an object with friction point contacts, then there will be a wrench exerted by each of the contacts that joins the two points [25]. At equilibrium, the intensities of the two contact wrenches are equal and opposite. If, for example, the thumb is moved toward the two opposing fingers while holding a rigid object in equilibrium, the internal grasping forces increase without affecting the stability of the grasp. Therefore, it is possible to test for stability by increasing the force at one of the contact points and detecting the concurrent increase of the force at the opposing contact.

7. SYSTEM STATUS AND EXPERIMENTAL RESULTS

We have implemented the tracking and control aspects of the system, and have experimented with static grasping of objects. We currently are implementing strategy 1 for dynamic grasping. The experimental results described below are for the visual tracking system.

Figure 1 shows the experimental hardware that was used for the initial experiments of the system. It consists of 2 CCD cameras with 25mm lenses mounted approximately 1.5 meters apart, with the moving object's full trajectory in the field of view. In the experiments described below, the lenses were increased to 50mm and the baseline increased, with the cameras placed approximately orthogonal to each other. The cameras were calibrated using the two-plane method of Martins et al [24] in which a planar test pattern is imaged twice in each camera. The calibration was able to determine the three-dimensional position of the test patterns to within 3-4 millimeters in X, Y and Z coordinates. The experimental results were obtained with the cameras mounted approximately 1 meter above the plane of the table pointing down at an angle of about 10 degrees from horizontal.

In the experiments, a model train was tracked by vision as it moved around a circular track and an oval track and the arm was commanded to follow each trajectory. The train was moving at a velocity of approximately 25 cm/sec (the algorithm was tested at faster speeds and works with increasing overshoot in the control algorithm). The results shown below were obtained without the robotic hand mounted on the system. Figure 4 shows the left and right camera images of the moving train, and Figure 5 shows the thresholded motion-energy for each camera found by the algorithm. Figure 6 is the actual arm trajectory obtained by following the train as it moves on an oval track. The trajectory is quite close to the actual trajectory except as the camera distance to the moving object increases. The algorithm is less accurate here for two reasons: first, the inherent stereo digitization error increases with distance from the cameras, and secondly, the profile of the moving object is quite different at this position, yielding two views whose motion centroids may not correspond to the same exact point in space, thus inducing triangulation error. The tracking begins with oscillations until the filter kicks in fairly rapidly as shown. The path is for 5 revolutions and is very repeatable due to the robustness of the vision algorithm. At the end of the path the arm is commanded to smoothly end its motion.

In addition to the trajectories above, we had the arm follow a semi-random trajectory created by a moving toy robot that changes its path direction arbitrarily when it hits an obstacle. In this experiment, the toy robot is placed inside the oval train track and it moves in a path until it hits the tracks, changing direction as it bounces off the tracks. While no ground truth is available for this trajectory, the arm follows the motion quite well, even though the robot moves in a way that is hard to model or predict. The results of these experiments are also available on video tape.

8. FUTURE WORK AND CONCLUSIONS

The vision algorithm described here has worked quite well in providing real-time visual servoing of a robotic arm, a fairly difficult task. We hope to continue to improve this algorithm, specifically by reducing the stereo error problems discussed above. One approach is to perform a better calibration of the camera system, and the other is try to better isolate centroids in each separate camera to be the same physical point. Various heuristics are being investigated on this front.

We are currently working on using the attached robotic hand to intercept and pick up the moving objects via the strategies described. The human pyschological paradigms we have discussed may provide us with further insights into robotic approaches.

References

1. Adelson, E. H. and J. R. Bergen, "Spatio-temporal enrgy models for the perception of motion," *Journal of the Optical Society of America*, vol. 2, no. 2, pp. 284-299, 1985.
2. Allen, Peter, "Real-time motion tracking using spatio-temporal filters," *Proceedings of DARPA Image Understanding Workshop*, Morgan-Kauffman Publishers, Palo Alto, May 1989.
3. Allen, Peter, Bil Yoshimi, and Alex Timcenko, "Real-time visual servoing," *Proceedings DARPA Image Understanding Workshop*, Pittsburgh, PA, September 11-13, 1990.
4. Anandan, P., *Measuring visual motion from image sequences,* Technical Report COINS-TR-87-21, COINS Department, University of Massachusetts, Amherst, 1987.
5. Aspex,, *PIPE User's Manual,* Aspex Incorporated, 530 Spring St., New York, NY.
6. Brown, Christopher, "Gaze controls with interafciton delays," *Proc. DARPA Image Understanding Workshop*, pp. 200-218, Morgan-Kauffman, Palo Alto, May 23-26, 1989.
7. Burt, P. J., J. R. Bergen, R. Hingorani, R. Kolczynski, W. A. Lee, A. Leung, J. Lubin, and H. Shvayster, "Object tracking with a moving camera," *IEEE Workshop on Visual Motion*, pp. 2-12, Irvine, CA, March 20-22, 1988.

8. Burt, P. J., C. Yen, and X. Xu, "Multi-resolution flow-through motion analysis," *Proceedings of the IEEE CVPR Conference*, pp. 246-252, 1983.

9. Buxton, B. F. and H. Buxton, "Computation of optic flow from the motion of edge features in image sequences," *Image and Vision Computing*, no. 2, 1984.

10. Clark, James J. and Nicola J. Ferrier, "Control of visual attention in mobile robots," *IEEE Conference on Robotics and Automation*, pp. 826-831, Scottsdale, AZ, May 15-19, 1989.

11. Corke, Peter, Richard Paul, and K. Wohn, *Video-rate visual servoing for sensory-based robotics,* Technical Report, Grasp Laboratory, Department of Computer and Information Science, University of Pennsylvania, Philadelphia.

12. Goldenberg, Richard, Wan Chi Lau, Alfred She, and Alan Waxman, "Progress on the prototype PIPE," *IEEE Conference on Robotics and Automation*, Raleigh, N. C., March 31 - April 3, 1987.

13. Hayward, Vincent and Richard Paul, "Robot manipulator control under UNIX," *Proc. of the 13th ISIR*, pp. 20:32-20:44, Chicago, April 17-21, 1983.

14. Heeger, David, "A model for extraction of image flow," *First International Conference on Computer Vision*, London, 1987.

15. Hofsten, Claes von, "Catching," in *Perspectives on Perception and Action*, ed. Herbert Heuer and Andries F. Sanders, pp. 33-36, Lawrence Erlbaum Associates, 1987.

16. Hofsten, Claes von, "Early Development of Grasping an Object in Space-Time," in *Vision and Action: The Control of Grasping*, ed. Melvyn A. Goodale, pp. 65-79, Ablex Publishing Company, 1990.

17. Horn, B. K. P., *Robot vision,* M.I.T. Press, 1986.

18. Horn, B. K. P. and B. Schunck, "Determining optical flow," *Artificial Intelligence*, vol. 17, pp. 185-203, 1983.

19. Iberall, Thea and Michael A. Arbib, "Schemas for the control of hand movements: an essay on cortical localization ," in *Vision and Action: The Control of Grasping*, ed. Melvyn A. Goodale, pp. 204-242, Ablex Publishing Company, 1990.

20. Kent, E. W., M. O. Shneier, and R. Lumia, "PIPE: Pipelined image processing engine," *Journal of Parallel and Distributed Computing*, no. 2, pp. 50-78, 1985.

21. Lee, D.N., D.S. Young, P.E. Reddish, S. Lough, and T.M.H Clayton, "Visual timing in hitting an accelerating ball," *Quarterly Journal of Experimental Psychology*, vol. 35A, pp. 333--346, 1983.

22. Lee, Sang Wook and K. Wohn, *Tracking moving objects by a mobile camera,* Technical Report MS-CIS-88-97, University of Pennsylvania, Department of Computer and Information Science, Philadelphia, November 1988.

23. Luo, Ren C., Robert E. Mullen Jr., and Daniel E. Wessell, "An adaptive robotic tracking system using optical flow," *IEEE Conference on Robotics and Automation*, pp. 568-573, Philadelphia, 1988.

24. Martins, H. A., J. R. birk, and R. B. Kelley, "Camera models based on data from two calibration planes," *Computer Graphics and Image Processing*, vol. 17, pp. 173-180, 1981.

25. Mason, Matthew T. and J. Kenneth Salisbury, *Robot hands and the mechanics of manipulation,* M. I. T. Press, Cambridge, 1985.

26. Miller, W. Thomas, "Real-time application of neural networks for sensor-based control of robots with vision," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 19, no. 4, pp. 825-831, July/Aug 1989.

27. Nagel, H. H., "On the estimation of dense displacement vector fields from image sequences," *Workshop on motion: Representation and Perception*, pp. 59-65, Toronto, 1983.

28. Paul, Richard, *Robot Manipulators,* MIT Press, Cambridge, MA, 1981.

29. Rao, B. S. Y. and H. F. Durrant-Whyte, "A fully decentralized algoprithm for multi-sensor kalman filtering," Report OUEL 1787/89, Dept. of Engineering Science, University of Oxford.

30. Safadi, Reem Bassam, "An adaptive algorithm for robotics and computer vision application," Tech. Report MS-CIS-88-05, Department of Computer and Information Science, University of Pennsylvania, January 1988.

31. Schmidt, Richard A., "The Schema Concept," in *Human Motor Behavior: An Introduction*, ed. J.A. Scott Kelso , pp. 219--238, Lawrence Erlbaum Associates, 1982.

32. Scott, G. L., "Four-line method of locally estimating optic flow," *Image and Vision Computing, 5(2)*, 1986.

33. Singh, Ajit, "An estimation-theoretic framework for image-flow computation," *Proc. International Conference on Computer Vision (ICCV-90)*, Kyoto, Japan, 1990.

34. Verghese, Gilbert, Karey Gale Lynch, and Charles R. Dyer, "Real-time motion tracking of three-dimensional objects," *IEEE International Conference on Robotics and Automation*, Cincinnati, May 13-18, 1990.

35. Weiss, Lee E., Arthur Sanderson, and Charles P. Neuman, "Dynamic sensor-based control of robots with visual feedback," *IEEE Journal of Robotics and Automation*, vol. RA-3, no. 5, pp. 404-417, October 1987.

36. Wiklund, Johan and Gosta Granlund, "Tracking of multiple moving objects," in *Time Varying Image Processing and Moving Object Recognition*, ed. V. Cappelini, pp. 241-249, Elsevier Science Publishers, 1987.

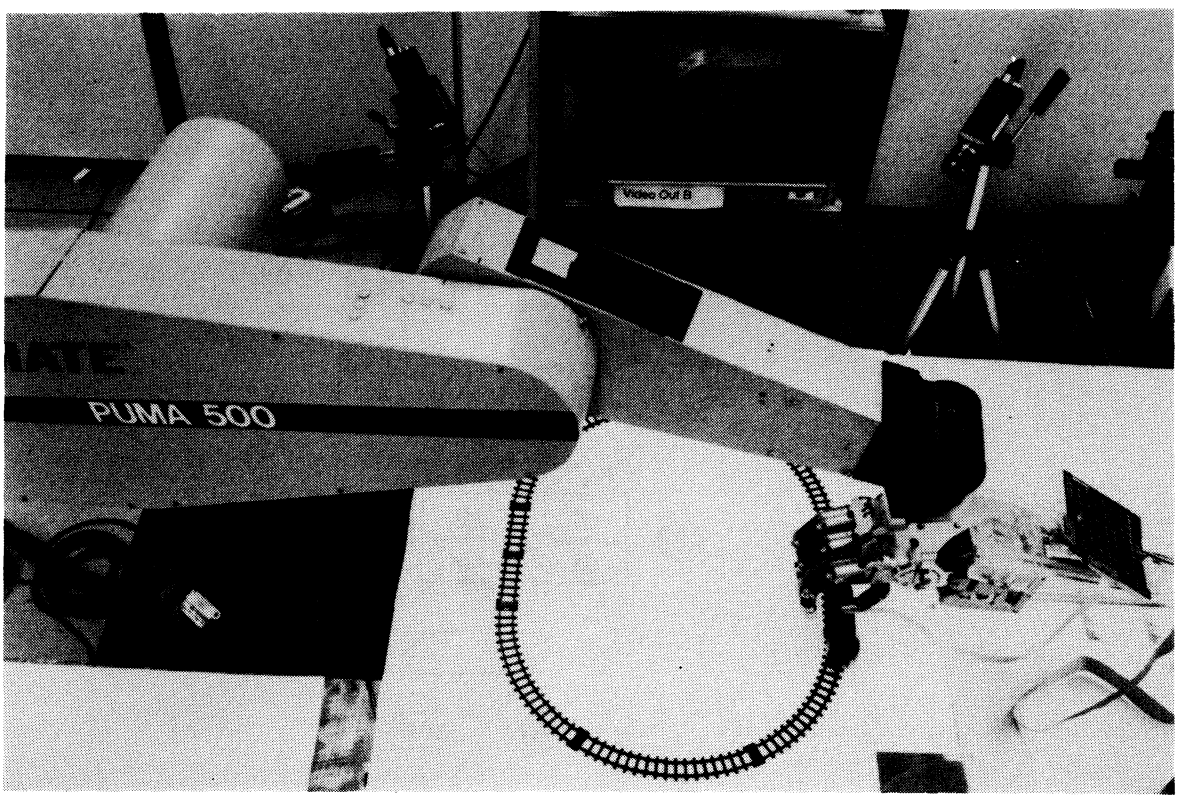


Figure 1: Experimental System.

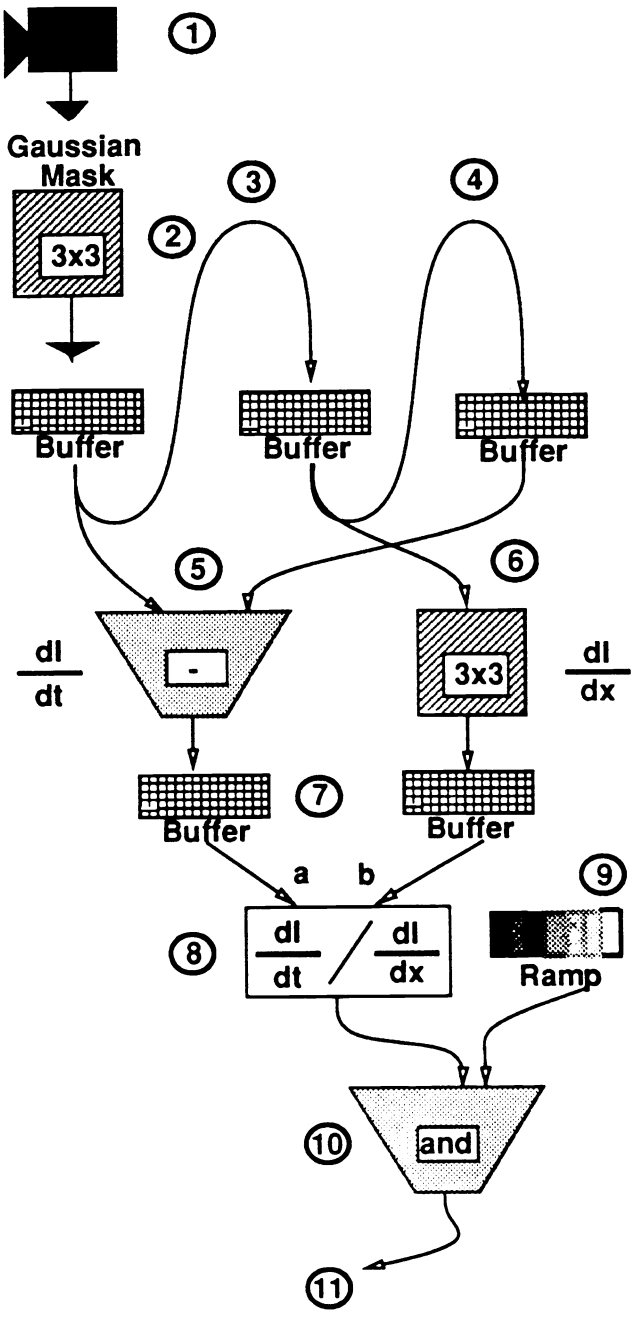
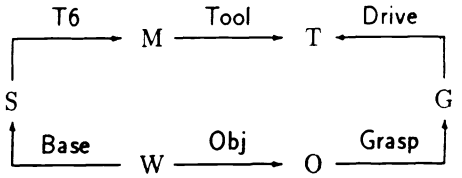


Figure 2: PIPE Optic-Flow algorithm.



W is world-coordinate frame

S is robot shoulder coordinate frame

M is 6th joint coordinate frame

T is tool (gripper) coordinate frame

G is grasping position coordinate frame

O is moving object coordinate frame

Base is constant transform between *W* and *S*

*T*6 is variable transform computed by RCCL in each sampling interval

Tool is variable transform defined by Utah-MIT hand kinematics

Drive is the transform introduced internally by RCCL to obtain straight-line motion in Cartesian coordinates

Grasp is constant transform which defines grasping point relative to the moving object

Obj is variable transform defined by vision subsystem outputs - it defines the position of the moving object in the world coordinate frame

Figure 3: Transform equations.

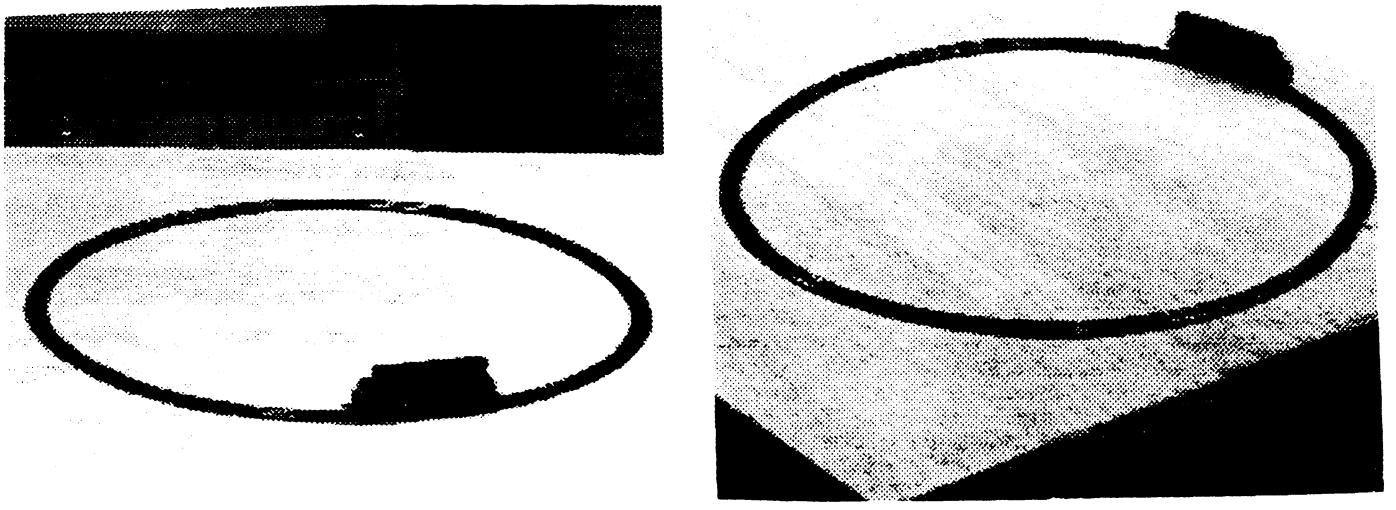


Figure 4: Left and right camera images of the train.

Figure 5: Thresholded motion energy for optic-flow,both cameras

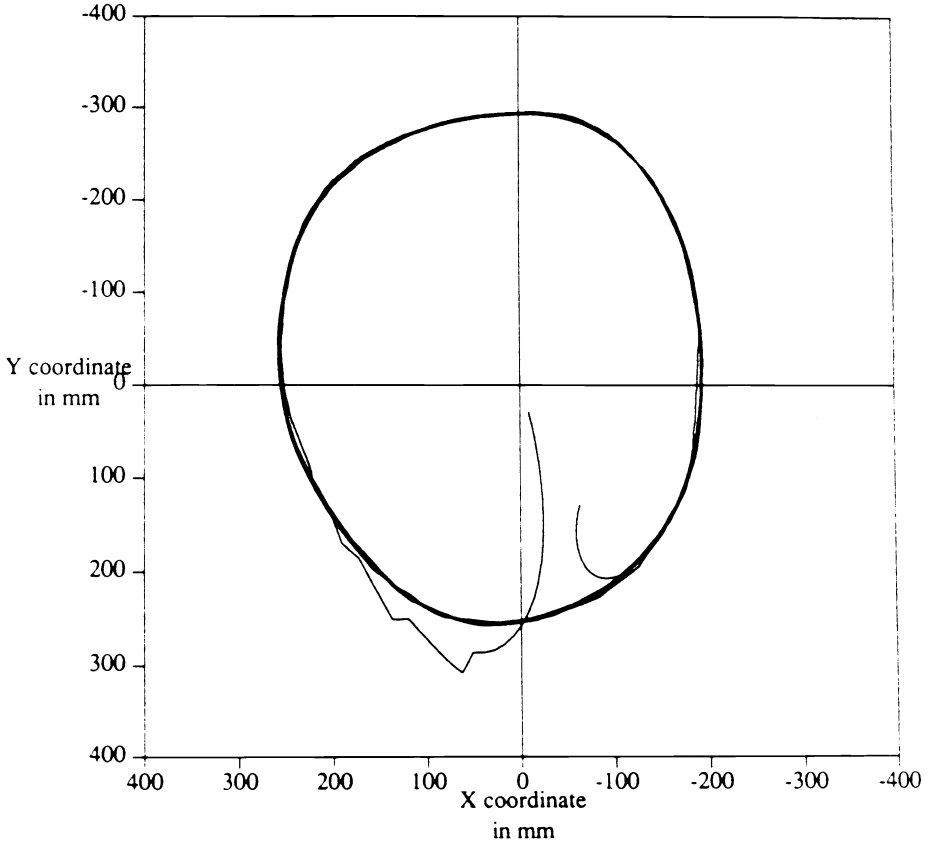


Figure 6: Tracked arm path, oval trajectory.