Presented at the 1994 IEEE Workshop
on Applications of Computer Vision,
December 5-7, 1994, Sarasota, FL

APPLICATION OF THE CONTROLLED ACTIVE VISION FRAMEWORK TO ROBOTIC AND TRANSPORTATION PROBLEMS

Christopher E. Smith Nikolaos P. Papanikolopoulos Scott A.Brandt
Artificial Intelligence, Robotics, and Vision Laboratory
Department of Computer Science
University of Minnesota
4-192 EE/CS Building
200 Union St. SE
Minneapolis, MN 55455

Abstract

Flexible operation of a robotic agent in an uncalibrated environment requires the ability to recover unknown or partially known parameters of the workspace through sensing. Of the sensors available to a robotic agent, visual sensors provide information that is richer and more complete than other sensors. In this paper we present robust techniques for the derivation of depth from feature points on a target's surface and for the accurate and high-speed tracking of moving targets. We use these techniques in a system that operates with little or no a priori knowledge of the object-related parameters present in the environment. The system is designed under the Controlled Active Vision framework [16] and robustly determines parameters such as velocity for tracking moving objects and depth maps of objects with unknown depths and surface structure. Such determination of intrinsic environmental parameters is essential for performing higher level tasks such as inspection, exploration, tracking, grasping, and collision-free motion planning. For both applications, we use the Minnesota Robotic Visual Tracker (a single visual sensor mounted on the end-effector of a robotic manipulator combined with a real-time vision system) to automatically select feature points on surfaces, to derive an estimate of the environmental parameter in question, and to supply a control vector based upon these estimates to guide the manipulator. The paper concludes with applications of these techniques to transportation problems such as vehicle tracking.

1 Introduction

In order to be effective, robotic agents in uncalibrated environments must operate in a flexible and robust manner. The computation of unknown parameters (e.g., the velocity of objects and the depth of object feature points) is essential information for the accurate execution of many robotic manipulation, inspection, and exploration tasks in unstructured settings. The determination of such parameters has traditionally relied upon the accurate knowledge of other related environmental parameters. For instance, traditional approaches to the problem of depth recovery [4][7][11] have assumed that extremely accurate measurements of the camera parameters and the camera system geometry are provided *a priori*, making these methods useful in only a limited number of situations. Similarly, previous approaches to visual tracking assumed known and accurate measures of camera parameters, camera positioning, manipulator positioning, target depth, target orientation, and environmental conditions [3].

This type of detailed information is not always available or, when it is available, not always accurate. Inaccuracies are introduced by positioning, path constraints, changes in the robotic system, and changes in the operational environment. In addition, camera calibration and the determination of camera parameters can be computationally expensive, time consuming, and error prone. In particular, depth derivation and tracking techniques that rely upon stereo vision systems require careful geometry measurements and the solution of the correspondence problem, making the computational overhead prohibitive for real- or near-real-time systems. Furthermore, many structure-from-motion algorithms use simple accidental motion of the camera that does not guarantee the best possible identifiability of the depth parameter. To be effective in uncalibrated environments, the robotic agent must perform under a variety of situations when only simple estimates of parameters (e.g., depth, focal length, pixel size, etc.) are used and with little or no *a priori* knowledge about the target, the camera, or the environment.

One solution to these problems can be found under the Controlled Active Vision framework [16]. Instead of relying heavily on *a priori* information, this framework provides the flexibility necessary to operate under dynamic conditions where many environmental and target-related factors are unknown and possibly changing. The Controlled Active Vision framework is based upon adaptive controllers that utilize the Sum-of-Squared Differences (SSD) optical flow measurements [1] as an input to the control loop. The SSD algorithm is used to measure the displacements of feature points in a sequence of images where the displacements may be induced by manipulator motion, target motion, or both. These measured displacements are then used as an input into the robot controllers, thus closing the control loop. Adaptive control techniques are useful under a variety of situations, including the application areas we have selected.

Instead of an accidental motion of the eye-in-hand system commonly used in depth extraction techniques [11][17], we propose a controlled exploratory motion that provides identifiability of the depth parameter. To reduce the influence of workspace-, camera-, and manipulator-

specific inaccuracies, an adaptive controller is utilized to provide accurate and reliable information regarding the depth of an object's feature points. This information may then be used to guide operations such as tracking, inspection, and manipulation [2][16].

Additionally, we propose a visual tracking system which does not rely upon accurate measures of environmental and target parameters. An adaptive controller is used to track feature points on a target's surface in spite of the unconstrained motion of the target, possible occlusion of feature points, and changing target and environmental conditions. High-speed targets are tracked with only rough operating parameter estimates and no explicit target models. Tracking speeds are ten times faster and have similar accuracy to those reported by Papanikolopoulos [16].

The robustness of the proposed tracking schemes has been tested on several IVHS applications. In general, an important component of a real-time intelligent traffic advisory system is the acquisition, processing, and interpretation of the available sensory information regarding the traffic conditions. At the lowest level, the sensory information is used to derive discrete signals to drive the advisory system and at a higher level this information is used to study the patterns of traffic flow or to adjust the behavior of a global traffic advisory system. Information about the traffic can be obtained through a variety of sensors such as loop detectors and vision sensors. Among them, the most commonly used is the loop detector. However, loop detectors provide local information and present significant errors in their measurements. Recently, many researchers [5][6][8][13][20] have proposed computer vision techniques for traffic monitoring and vehicle control. Since the transportation area seems to be an ideal application area, this paper presents the results from the application of our algorithms to vehicle tracking problems.

We first formulate the equations for visual measurements, including an enhanced SSD surface construction strategy and optimizations, and present the feature point selection scheme. Next, we present an outline of our proposed methods for visual tracking and depth recovery. Furthermore, we discuss results from experiments in both of the selected applications using the Minnesota Robotic Visual Tracker (MRVT). The paper concludes with experimental results from the application of these techniques to transportation problems such as vehicle tracking.

2 Visual Measurements

Our robotic and transportation applications both use the same basic visual measurements which are based upon a simple camera model and the measure of optical flow in a temporal sequence of images. The visual measurements are combined with search-specific optimizations and a dynamic pyramiding technique in order to enhance the visual processing from frame-to-frame and to optimize the performance of the system in our selected applications.

2.1 Camera Model and Optical Flow

We assume a pinhole camera model (focal length f) with a world frame, R_W, centered on the optical axis. A point $\mathbf{P} = (X_W, Y_W, Z_W)^T$ in R_W, projects to a point $\mathbf{p}$ in the image plane with coordinates (x, y). We can define two scale factors s_x and s_y to account for camera sampling and pixel size, and include the center of the image coordinate system (c_x, c_y) given in frame F_A [16]. This results in the following equations for the actual image coordinates (x_A, y_A):

$$x_A = \frac{fX_W}{s_x Z_W} + c_x = x + c_x \tag{2.1}$$

$$y_A = \frac{fY_W}{s_y Z_W} + c_y = y + c_y. \tag{2.2}$$

It is assumed that the camera moves in a static environment with a translational velocity (T_x, T_y, T_z) and a rotational velocity (R_x, R_y, R_z). The velocity of point $\mathbf{P}$ with respect to R_W can be expressed as:

$$\frac{d\mathbf{P}}{dt} = -\mathbf{T} - \mathbf{R} \times \mathbf{P}. \tag{2.3}$$

By taking the time derivatives and using equations (2.1), (2.2), and (2.3), we obtain ($u = \dot{x}$ and $v = \dot{y}$):

$$u = \left[x\frac{T_z}{Z_W} - f\frac{T_x}{Z_W s_x}\right] + \left[x\,y\frac{s_y R_x}{f} - \left(\frac{f}{s_x} + x^2\frac{s_x}{f}\right)R_y + y\frac{s_y}{s_x}R_z\right] \tag{2.4}$$

$$v = \left[y\frac{T_z}{Z_W} - f\frac{T_y}{Z_W s_y}\right] + \left[\left(\frac{f}{s_y} + y^2\frac{s_y}{f}\right)R_x - x\,y\frac{s_x}{f}R_y - x\frac{s_x}{s_y}R_z\right]. \tag{2.5}$$

We use a matching-based technique known as the Sum-of-Squared Differences (SSD) optical flow [1]. For the point $\mathbf{p}(k-1) = (x(k-1), y(k-1))^T$ in the image $(k-1)$ where k denotes the kth image in a sequence of images, we want to find the point $\mathbf{p}(k) = (x(k-1)+u, y(k-1)+v)^T$. This point $\mathbf{p}(k)$ is the new position of the projection of the feature point $\mathbf{P}$ in image k. We assume that the intensity values in the neighborhood N of $\mathbf{p}$ remain relatively constant over the sequence k. We also assume that for a given k, $\mathbf{p}(k)$ can be found in an area Ω about $\mathbf{p}(k-1)$ and that the velocities are normalized by time T to get the displacements. Thus, for the point $\mathbf{p}(k-1)$, the SSD algorithm selects the displacement $\Delta\boldsymbol{x} = (u, v)^T$ that minimizes the SSD measure

$$e(\mathbf{p}(k-1), \Delta\boldsymbol{x}) = \sum_{m,n \in N} \left[I_{k-1}(x(k-1)+m, y(k-1)+n) - I_k(x(k-1)+m+u, y(k-1)+n+v)\right]^2 \tag{2.6}$$

where $u,v \in \Omega$, N is the neighborhood of $\mathbf{p}$, m and n are indices for pixels in N, and I_{k-1} and I_k are the intensity functions in images $(k-1)$ and (k).

The size of the neighborhood N must be carefully selected to ensure proper system performance. Too small an N fails to capture enough contrast while too large an N increases the associated computational overhead and

enhances the background. In either case, an algorithm based upon the SSD technique may fail due to inaccurate displacements. An algorithm based upon the SSD technique may also fail due to too large a latency in the system or displacements resulting from motions of the object which are too large for the method to accurately capture. We introduce the techniques related to search optimizations and dynamic pyramiding to counter these concerns.

2.2 Search Optimizations

The primary source of latency in a vision system that uses the SSD measure is the time needed to identify $(u, v)^{\mathrm{T}}$ in equation (2.6). To find the true minimum, the SSD measure must be calculated over each possible $(u, v)^{\mathrm{T}}$. The time required to produce an SSD surface and to find the minimum can be greatly reduced by employing two schemes that, when combined, divide the search time significantly in the expected case.

The first optimization used is loop short-circuiting. During the search for the minimum on the SSD surface (the search for $u_{\min}$ and $v_{\min}$), the SSD measure must be calculated according to equation (2.6). This requires nested loops for the *m* and *n* indices. During the execution of these loops, the SSD measure is calculated as the running sum of the squared pixel value differences. If the current SSD minimum is checked against the running sum as a condition on these loops, the execution of the loops can be short-circuited as soon as the running sum exceeds the current minimum. This optimization has a worst-case performance equivalent to the original algorithm plus the time required for the additional condition tests. This worst case occurs when the SSD surface minimum lies at the last $(u, v)^{\mathrm{T}}$ position searched. On average, this short-circuit realizes a decrease in execution time by a factor of two.

The second optimization is based upon the heuristic that the best place to begin the search for the minimum is at the point where the minimum was last found on the surface and to expand the search radially from this point. This heuristic works well when the disturbances being measured are relatively regular. In the case of target tracking, this corresponds to targets that have locally smooth velocity, acceleration, and jerk curves. If a target's motion does not exhibit such relatively smooth curves, then the target itself is fundamentally untrackable due to the inherent latency in the video equipment and the vision processing system. During depth recovery, smooth motion curves correspond to smooth manipulator motions. A relatively non-smooth manipulator trajectory would likely be mechanically unachievable by the system.

Under this heuristic, the search pattern in the (k) image is altered to begin at the point on the SSD surface where the minimum was located for the $(k-1)$ image. The search pattern then spirals out from this point, searching over the extent of *u* and *v*. This is in contrast with the typical indexed search pattern where the indices are incremented in a row-major scan fashion. This spiral search strategy may also be combined with a predictive controller to begin the search for the SSD minimum at the position that the predictive aspect of the controller indicates is the possible location of the minimum.

Since the structure that implements the spiral search pattern contains no more overhead than the loop structures of the traditional search, worst-case performance is identical. In general, search time is approximately halved.

When combined, these two optimizations (the loop short-circuiting and the spiral search pattern) find the minimum of the SSD surface approximately ten times faster on average than the unmodified search. Experimentally, the search times for the unmodified algorithm averaged 136 msec over 5000 frames under a variety of relative feature point motions. The modified algorithm with the short-circuiting alone averaged 60-72 msec search times over several thousand frames with arbitrary relative feature point motion. The combined short-circuit/spiral search algorithm produced search times which averaged 13 msec under similar tests. Therefore, these optimizations allow the vision system to track two to three features at RS-170 video rates (33 msec total with vertical blanking) without video under-sampling.

2.3 Dynamic Pyramiding

Dynamic pyramiding is a heuristic technique which attempts to resolve the conflict between accurate positioning of a manipulator and high-speed tracking when the displacements of the feature points are large. Previous applications have typically depended upon one preset level of pyramiding to enhance either the top tracking speed of the manipulator or the positioning accuracy of the end-effector above the target [16].

In contrast, dynamic pyramiding uses multiple levels of pyramiding. The level of the pyramiding is selected based upon the observed displacements of the target's feature points. If the displacements are small relative to the search area, the pyramiding level is reduced; if the measured displacements are large compared to the search area, then the pyramiding level is increased. This results in a system that enhances the tracking speed when required, but always biases in favor of the maximum accuracy achievable. The dynamic switching thus allows the tracker to adjust to accelerations of the target (when displacements increase) and then to increase accuracy when the tracking adapts to the new, higher speed or when the target is at rest.

During the search process, the SSD measurements are centered upon particular pixels in the pyramided search area. Which pixel positions are selected ($\Delta \boldsymbol{x} = (u, v)^{\mathrm{T}}$ in equation (2.6)) is dependent upon which of the four levels of pyramiding is currently active. The lowest level searches every pixel in a square 32×32 pixel patch of the current frame. The fourth and highest level searches every fourth pixel in a 128×128 pixel patch.

3 Feature Point Selection

In addition to the system latency and the effect of large displacements, an algorithm based upon the SSD technique may fail due to repeated patterns in the intensity function of the image or due to large areas of uniform intensity in the image. Both cases can provide multiple matches within a feature point's neighborhood, resulting in inaccurate displacement measures. In order to avoid this problem, our system automatically evaluates and selects feature points.

Feature points are selected using the SSD measure combined with an auto-correlation technique to produce an SSD surface corresponding to an auto-correlation in the neighborhood N [1][16]. Several possible confidence measures can be applied to the surface to measure the suitability of a potential feature point.

The selection of a confidence measure is critical since many such measures lack the robustness required by changes in illumination, intensity, etc. We utilize a two-dimensional displacement parabolic fit that attempts to fit parabola $e(\Delta x_r) = a\Delta x_r^2 + b\Delta x_r + c$ to a cross-section of the surface derived from the SSD measure [16]. The parabola is fit to the surface in several predefined directions. Papanikolopoulos [16] selected a feature point if the minimum directional measure was sufficiently high, as measured by equation (2.6).

For the purpose of depth extraction, we extend this approach in response to the aperture problem [10]. Briefly stated, the aperture problem refers to the motion of points that lie on a feature such as an edge. Any sufficiently small local measure of motion can only retrieve that component of motion orthogonal to the edge. Any component of motion due to movement other than movement perpendicular to the edge is unrecoverable due to multiple matches for any given feature point.

The improved confidence measure for feature points permits the selection of a feature if it has a sufficiently high correlation with the parabola in a particular direction. For instance, a feature point corresponding to an edge will only have a high correlation in the direction perpendicular to the edge.

This directional information is then used during the depth recovery process to constrain the motion of the manipulator. The motion must be constrained such that the displacements of the feature point only occur orthogonal to the directional feature. This ensures that the only component of motion is precisely the component that can be recovered under consideration of the aperture problem. This ideal motion is not always achievable; therefore, motions will occur that are approximately, but not completely, orthogonal to the directional feature. In response to this uncertainty in the actual motion of the manipulator, matches that lie closer to the direction of the ideal manipulator motion will be preferred.

4 Modeling and Controller Design

Modeling of the system in question is critical to the design of a controller to perform the task at hand. In our application areas, the modeling and controller designs are similar but distinct. We present only an outline of the depth recovery modeling and the controller design. Papanikolopoulos *et al.* [15] provide an in-depth treatment of the modeling and controller design for robotic visual tracking applications.

Several robotic servoing problems require accurate information regarding the depth of the object of interest. If the depth is not provided *a priori*, it must be recovered from observations of the environment via sensing. Traditional vision-based depth recovery techniques may suffer from various problems, including computational overhead, expensive calibration, frequent recalibration, obstruction, and solution of the correspondence problem. We present an alternative algorithm for depth recovery using the Controlled Active Vision framework.

Consider a target object at an unknown depth with a feature point, **P**, located on the surface of the object. By moving the visual sensor, a sequence of images and the respective projections of **P** can be effected. By observing the changes of **P**'s projections in successive images, an estimate of the depth of **P** is derived. Over multiple images, this estimate can be refined using adaptive techniques [18].

Previous work in depth recovery using active vision relied upon random camera displacements to provide displacement changes in the $\mathbf{p}(k)$'s [7][12][17]. In our approach, we select features automatically, produce feature trajectories (a different trajectory for every feature), and predict future displacements using the depth estimate [18]. The errors in these predicted displacements, in conjunction with the estimated depth, are included as inputs in the control loop of the system. Thus, the next calculated movement of the system produces a camera movement that will eliminate the largest possible portion of the observed error while adhering to various environmental- and manipulator-specific constraints. This purposeful movement of the visual sensor provides more accurate depth estimates and a faster convergence of the depth measures over a sequence of estimates. More details about our approach can be found in [18].

5 Experimental Design and Results

5.1 The MRVT System

We have implemented the depth recovery and the visual tracking on the Minnesota Robotic Visual Tracker (MRVT) system. The MRVT is a multi-architectural system with two main parts: the Robot/Control Subsystem (RCS) and the Vision Processing Subsystem (VPS).

The RCS is comprised of a PUMA 560 manipulator, its associated Unimate Computer/Controller, and a VME-based Single Board Computer (SBC). The manipulator‘s

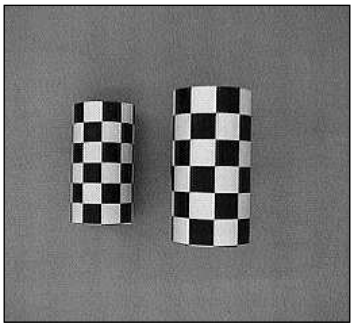

Figure 1: Dual can target

trajectory is controlled via the Unimate controller's Alter line and requires path control updates once every 28 msec. Those updates are provided by an Ironics 68030 VME SBC running Carnegie Mellon University's CHIMERA real-time robotic executive. A Sun SparcStation 330 serves as the CHIMERA host and shares its VME bus with the Ironics SBC via BIT-3 VME-to-VME bus extenders. BIT-3 extenders are also used to allow a shared memory configuration between the RCS and the VPS.

The VPS receives input from a Panasonic GP-KS102 miniature camera that is mounted parallel to the end-effector of the PUMA and provides a video signal to a Datacube system for processing. The Datacube is the main component of the VPS and consists of a Motorola MVME-147 SBC running OS-9, a Datacube MaxVideo20 video processor, a Datacube Max860 vector processor, and a BIT-3 VME-to-VME bus extender.

5.2 Derivation of Depth

The initial experimental runs were conducted using two soda cans wrapped in a checkerboard surface as targets. The leading edges of the cans were placed at 53 cm and 41 cm in depth (see Figure 1). The majority of the errors in the calculated depths were in the order of sub-pixel. For these runs, the initial depth estimate was an underestimate of 25 cm. The reconstructed surfaces of the dual can experiment are shown in Figure 2. The reconstructed surfaces show that the process has captured the curvature of both cans in this experiment.

We duplicated the first experiment without the checkerboard surfaces wrapped around the cans. Instead, we used the actual surfaces of the soda cans under normal lighting

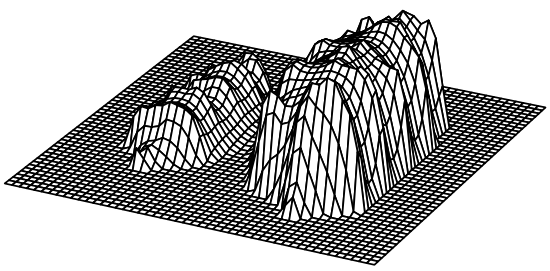

Figure 2: Dual reconstructed surfaces

Figure 3: Dual cans with real texture

conditions. This reduced the number of suitable feature points as well as producing a non-uniform distribution of feature points across the surface of the cans. Additionally, the surfaces of the cans exhibited specularities and reflections which added noise to the displacement measures. This experimental run using the real surfaces of two cans at different depths produced results which are shown in Figure 3 and Figure 4. The results demonstrate that the additional noise and the distribution of suitable features affect the recovery of the curvature on the two surfaces.

5.3 Robotic Visual Tracking

We conducted multiple experimental runs for the tracking of objects that exhibited unknown two-dimensional, translational motion with a coarse estimate of the depth of the objects. The targets for these runs included books, batons, and a computer-generated target displayed on a large video monitor. During the experiments, we tested the system using targets with linear and curved trajectories.

The first experiments were conducted using a book and a baton as targets in order to test the performance of the system both with and without the dynamic pyramiding. These initial experiments served to confirm the feasibility of performing real-time pyramid level switching and to collect data on the performance of the system under the pyramiding and search optimizing schemes. Figure 5 shows the target trajectory and manipulator path from one of these experiments. With the pyramiding and search optimizations, the system was able to track the corner of a book which was moving along a linear trajectory at 80 cm/sec. In contrast, the maximum speed reported by Papanikolopoulos [16] was 7 cm/sec, representing an order of

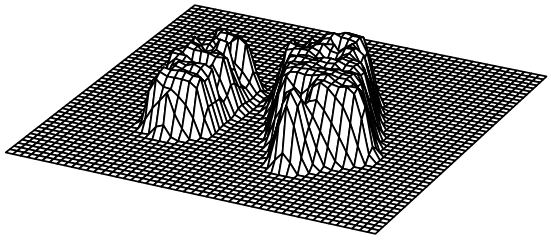

Figure 4: Dual reconstructed surfaces

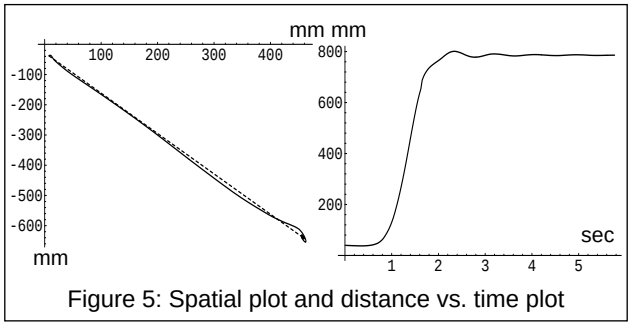

Figure 5: Spatial plot and distance vs. time plot

magnitude increase in tracking speed. The dashed line is the target trajectory and the solid line represents the manipulator trajectory. The start of the experiment is in the upper left corner and the end is where the target and manipulator trajectories come together in the lower right. The oscillation at the beginning is due to the switch to higher pyramiding levels as the system compensates for the speed of the target. The oscillations at the end are produced when the target slows to a stop and the pyramiding drops down to the lowest level. This results in the manipulator centering above the feature accurately due to the higher resolution available at the lowest pyramid level.

Once system performance met our minimal requirements, testing proceeded to a computer-generated target where the trajectory and speed could be accurately controlled. The target (a white box on the video monitor) traced a circle while the PUMA tracked the target. This path produces a smoothly varying velocity curve while producing an infinite acceleration of the target at start-up. We allowed the experiments to continue for many circuits of the target in order to determine the response of the system over time. The results demonstrated an initial oscillation due to the acceleration at start-up followed by smooth tracking once the controller compensated for the extreme initial disturbances. The results tracking are shown in Figure 6. The multiple manipulator paths demonstrate the repeatability achieved using the controller, the algorithms, and the optimizations. The target in this experiment moved with a speed of 35 cm/sec.

5.4 Vehicle Tracking

A primary requirement of an IVHS system is that it must be able to detect and track potential obstacles. Of particular interest are other vehicles moving around and in front of the vehicle upon which the sensor is mounted (the primary vehicle). These vehicles constitute obstacles that must be avoided (collision avoidance) or a target that is to be followed (convoying).

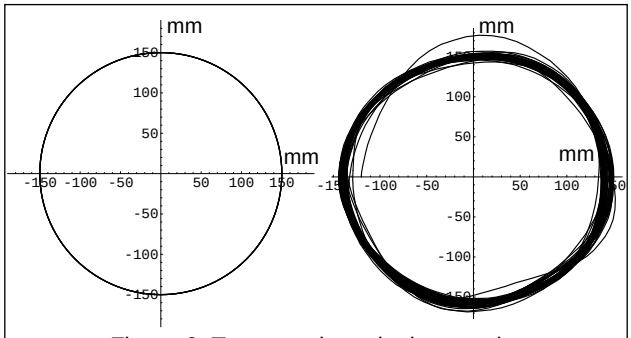

Figure 6: Target and manipulator paths

Collision avoidance of moving vehicles (with a relatively constant velocity) detected near the primary vehicle can be effected with careful path planning. The basic goal is to maintain the desired speed and path (i.e., staying in the same lane of the road or highway) while avoiding other vehicles. Under normal circumstances this means accelerating and decelerating appropriately to avoid cars in front of and behind the primary vehicle. In extreme cases this means taking evasive action or warning the operator of a potential collision situation.

Another application area that involves the same basic problems is vehicle convoying. In this case, all path planning is done by the operator of the vehicle at the head of the convoy and all other vehicles must follow at a specified distance in a column behind the primary vehicle. For these reasons we chose to apply the Controlled Active Vision framework to the problem of tracking vehicles moving in roughly the same direction as the primary vehicle.

The experimental data was collected by placing a camcorder in the passenger seat of a car and driving behind other cars on the freeway. This produced data that closely matched that which could be expected in a typical IVHS application. This data was later played back through a VCR and used as input to the VPS, which tracked the vehicles and produced a series of (x, y) locations specifying the detected locations of the features being tracked. Because the exact world coordinate locations of the features in each frame of the video were unknown, we were unable to provide a complete analysis of the tracking performance of the system. However, a simple assumption about the motion of the vehicle in the frame yielded conclusive results that matched the intuitive results gained from viewing the graphic display of the vehicle tracking algorithm.

The VPS includes a monitor upon which graphic representations of the tracking algorithm are displayed. While the system is tracking, the input data is displayed and a red box is drawn around the tracked location of the feature (see Figure 9). Initial experiments were done by watching the system track the features as they were displayed on the monitor. After initial studies were completed, we proceeded to do a quantitative analysis of the vehicle tracking performance. While we were unable to compare the tracking results against ground-truth data (exact locations of objects/features manually calculated off-line), we were able to provide a reasonable determination of the system performance by assuming that the motion of the vehicle within the frame would be relatively smooth and within easily determinable velocity bounds. In other words, any sufficiently large, quick motions of the tracked location were likely to be the result of a loss of tracking rather than due to motion of the vehicle. This hypothesis was con-

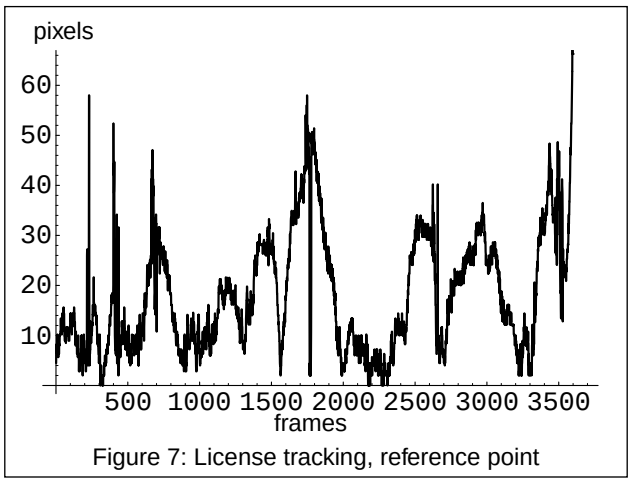

Figure 7: License tracking, reference point

firmed by viewing the tracking results and correlating the spikes in the plots of the tracked locations with the motion of the displayed tracking windows.

One such experimental run tracked a single feature (the license plate) on the back of a single vehicle that was followed for approximately 2 minutes. Figure 7 shows the motion of the feature in the frame with respect to a reference point in the image that corresponds to an initial location of the feature. Figure 8 shows the first derivative of the plot in Figure 7. This plot clearly shows the relatively regular motion of the calculated feature position, corresponding to the relatively smooth motion of the tracked vehicle relative to the vehicle carrying the sensor. The spikes in the plot correspond to the times where the tracking was lost and the tracking window rapidly moved off the feature. The plot has been annotated to indicate the cause of the tracking problems, that in all cases were due to either momentary occlusion of the tracked feature by windshield wipers on the primary vehicle, or extreme changes in the relative brightness and contrast of the tracked feature as a result of the vehicles passing under overpasses on the freeway. In all cases tracking was resumed immediately after the visual disturbance ended.

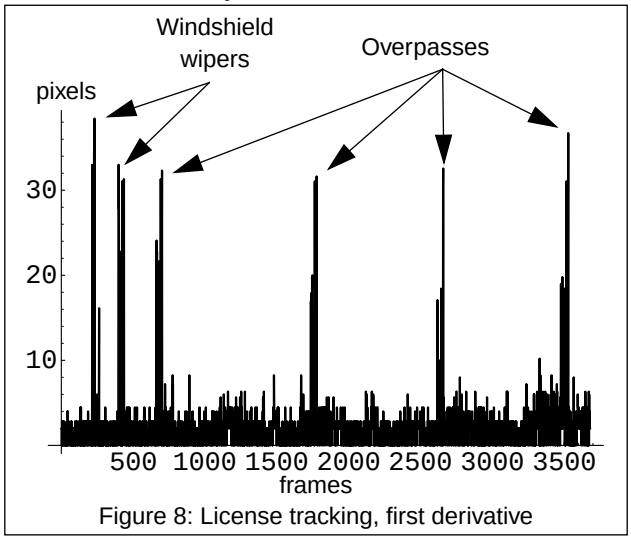

Figure 8: License tracking, first derivative

In real-time this corresponded to less than a second of lost tracking per occurrence. Simple filtering techniques (i.e., Kalman filtering) would remove virtually all perturbation of the tracking window resulting from such events.

Four selected frames of the test are presented in Figure 9 at the end of this paper. The image labeled “(c)” is taken just prior to the loss of tracking due to windshield wiper occlusion. The blur due to the wiper is just appearing in the lower left of the frame.

6 Conclusion

This paper presents robust techniques for the operation of robotic agents in uncalibrated environments (e.g., nuclear sites, highways, etc.). The techniques presented provide ways of recovering unknown environmental parameters using the Controlled Active Vision framework [16]. In particular, this paper presents novel techniques for computing depth maps and for visual tracking through controlled active exploration with a camera. In order to prove the usefulness of the approach, we presented experimental results from the application of these techniques to transportation problems such as vehicle tracking.

For the computation of depth maps, we propose a scheme that is based on the automatic selection of features and design of specific trajectories on the image plane for each individual feature. Unlike similar approaches [9][11] [12][17][19], this approach helps to design trajectories that provide maximum identifiability of the depth parameter. During the execution of the specific trajectory, the depth parameter is computed with the help of a simple estimation scheme that takes into consideration the previous movements of the camera and the computational delays. The approach has been tested and several experimental results have been presented.

For the problem of visual tracking, we propose a technique based upon earlier work in visual servoing [16] that achieves superior speed and accuracy through the introduction of several performance enhancing techniques. In particular, the dynamic pyramiding technique provides a satisfactory compromise to the speed/accuracy trade-off inherent in static pyramiding techniques. The method-specific optimizations presented also enhance overall system performance without affecting worst-case execution times. These optimizations apply to various region-based vision processing applications and in our application, provide the speedup required to increase directly the effectiveness of the real-time vision system. Finally, these optimizations have been tried to tracking problems that appear in transportation. The transportation experiments indicate the robustness and the computational gains that the proposed approach introduces.

Issues for future research include the automatic selection of the feature window size (an issue discussed in [14]) in order to select a window that has some texture variations, the use of active deformable models in conjunction

with the SSD flow, and the implicit incorporation of the system dynamics (e.g., robot dynamics in eye-in-hand applications).

7 Acknowledgments

This work has been supported by the Department of Energy (Sandia National Laboratories) through Contracts #AC-3752D and #AL-3021, the National Science Foundation through Contracts #IRI-9410003 and #CDA-9222922, the Center for Transportation Studies through Contract #USDOT/DTRS 93-G-0017-1, the Army High Performance Computing Research Center, the 3M Corporation, the Graduate School of the University of Minnesota, and the Department of Computer Science of the University of Minnesota.

8 References

[1] P. Anandan, "Measuring visual motion from image sequences," Technical Report COINS-TR-87-21, COINS Department, University of Massachusetts, 1987.

[2] J.T. Feddema and C.S.G. 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.

[3] D.B. Gennery, "Tracking known three-dimensional objects," *Proceedings of the AAAI 2nd National Conference on AI*, 13-17, 1982.

[4] J. Heel, "Dynamic motion vision," *Robotic and Autonomous Systems*, 6(3):297-314, 1990.

[5] A. Houghton, G.S. Hobson, L. Seed, and R.C. Tozer, "Automatic monitoring of vehicles at road junctions," *Traffic Engineering Control*, 28(10):541-453, October, 1987.

[6] R.M. Inigo, "Application of machine vision to traffic monitoring and control," *IEEE Transactions on Vehicular Technology*, 38(3):112-122, August, 1989.

[7] C.P. Jerian and R. Jain, "Structure from motion—a critical analysis of methods," *IEEE Transactions on Systems, Man, and Cybernetics*, 21(3):572-588, 1991.

[8] N. Kehtarnavaz, N.C. Griswold, and J.S. Lee, "Visual control of an autonomous vehicle (BART) — the vehicle-following problem," *IEEE Transactions on Vehicular Technology*, 40(3):654-662, August, 1991.

[9] K.N. Kutulakos and C.R. Dyer, "Recovering shape by purposive viewpoint adjustment," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 16-22, 1992.

[10] D. Marr, *Vision*, W.H. Freeman and Company, San Francisco, CA, 1982.

[11] L. Matthies, T. Kanade, and R. Szeliski, "Kalman filter-based algorithms for estimating depth from image sequences," *International Journal of Computer Vision*, 3(3):209-238, 1989.

[12] L. Matthies, "Passive stereo range imaging for semi-autonomous land navigation," *Journal of Robotic Systems*, 9(6):787-816, 1992.

[13] P.G. Michalopoulos, "Vehicle detection video through image processing: the autoscope system," *IEEE Transactions on Vehicular Technology*, 40(1):21-29, February, 1991.

[14] M. Okutomi and T. Kanade, "A locally adaptive window for signal matching," *International Journal of Computer Vision*, 7(2):143-162, 1992.

[15] N. Papanikolopoulos, P.K. Khosla, and T. Kanade, "Visual tracking of a moving target by a camera mounted on a robot: a combination of control and vision," *IEEE Transactions on Robotics and Automation*, 9(1):14-35, 1993.

[16] N. Papanikolopoulos, "Controlled active vision," Ph.D. Thesis, Department of Electrical and Computer Engineering, Carnegie Mellon University, 1992.

[17] C. Shekhar and R. Chellappa, "Passive ranging using a moving camera," *Journal of Robotic Systems*, 9(6):729-752, 1992.

[18] C.E. Smith and N. Papanikolopoulos, "Computation of shape through controlled active exploration," *Proceedings of the IEEE International Conference on Robotics and Automation*, 2516-2521, 1994.

[19] C. Tomasi and T. Kanade, "Shape and motion from image streams under orthography: a factorization method," *International Journal of Computer Vision*, 9(2):137-154, 1992.

[20] B. Ulmer, "VITA- an autonomous road vehicle (ARV) for collision avoidance in traffic," *Proceedings of Intelligent Vehicles '92*, 36-41, 1992.

(a)

(b)

(c)

(d)

Figure 9: Vehicle Tracking