

# Computation of Shape Through Controlled Active Exploration

Christopher E. Smith and Nikolaos P. Papanikolopoulos  
Artificial Intelligence, Robotics, and Vision Laboratory  
Department of Computer Science  
University of Minnesota  
Minneapolis, MN 55455

## Abstract

*Accurate knowledge of depth continues to be of critical importance in robotic systems. Without accurate depth knowledge, tasks such as inspection, tracking, grasping, and collision-free motion planning prove to be difficult and often unattainable. Traditional visual depth recovery has relied upon techniques that require the solution of the correspondence problem or require known lighting conditions and Lambertian surfaces. In this paper, we present a technique for the derivation of depth from feature points on a target's surface using the Controlled Active Vision framework. We use a single visual sensor mounted on the end-effector of a robotic manipulator to automatically select feature points and to derive depth estimates for those features using adaptive control techniques. Movements of the manipulator produce displacements that are measured using a Sum-of-Squared Difference (SSD) optical flow. The measured displacements are fed into the controller to alter the path of the manipulator and to refine the depth estimate.*

## 1 Introduction

The computation of the relative and absolute depths of objects' feature points is essential information for the accurate execution of several robotic manipulation and inspection tasks. Traditional approaches to the problem of depth recovery have assumed that extremely accurate measurements of the camera parameters and the camera system geometry are provided *a priori*. Such information is not always available or, when it is available, is not always accurate. Inaccuracies are introduced by positioning, path constraints, and environmental changes in the robotic system. Furthermore, camera calibration and the determination of camera parameters can be computationally expensive, time consuming, and error prone. In particular, many depth derivation techniques rely upon stereo vision systems that require careful geometry measurements and the solution of the correspondence problem, making the computational overhead prohibitive for real-time systems. Finally, many structure from motion algorithms use simple accidental motion of the camera that does not guarantee the best possible identifiability of the depth.

One solution to these problems can be found under the

Controlled Active Vision framework. Instead of an accidental motion of the eye-in-hand system [8][13][15], we propose a controlled exploratory motion that provides identifiability of the depth parameter. An adaptive controller can be utilized to reduce the influence of such environmental, camera, and manipulator specific inaccuracies 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 [3][12].

In this paper, we present a technique for the derivation of feature point depth based upon a Sum-of-Squared Differences (SSD) optical flow [1] under an adaptive control framework. The premise behind this technique is that, given an eye-in-hand robotic system, features are automatically selected, feature dependent trajectories are produced, and displacements of feature points are measured using the SSD optical flow. These displacements are compared to the predicted displacements that are derived using the current depth estimate. The errors from these comparisons are then used, in conjunction with the previous displacements, to update the depth estimate and to produce the next control input. The control input is derived such that the amount of the error removed in the next iteration will be maximized given environmental and manipulator specific constraints.

In this paper, we formulate the equations for optical measurements and the feature point selection scheme. Then, we derive the control and measurement equations used for the adaptive controller and elaborate on the selection of the features' trajectories. Finally, we discuss results from simulation and from experiments conducted using the Minnesota Robotic Visual Tracker (MRVT).

## 2 Vision-based measurements and feature selection

### 2.1 Camera model and optical flow

We assume a pinhole camera model with a world frame,  $R_W$ , centered on the optical axis and a focal length  $f$ . 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 also define two scale factors  $s_x$  and  $s_y$  to account for camera sampling and pixel size [12]. Any displacement of the point  $\mathbf{P}$  can be described by a rota-

tion about an axis through the origin and a translation. If this rotation is small, then it can be described as three independent rotations about the three axes  $X_W$ ,  $Y_W$ , and  $Z_W$  [2]. We will assume that the camera moves in a static environment with a translational velocity  $(T_x, T_y, T_z)$  and a rotational velocity  $(R_x, R_y, R_z)$ . This results in the following equations for the optical flow (see [12] for a complete derivation):

$$u = \left[ \frac{x}{Z_W} T_z - \frac{f}{Z_W s_x} T_x \right] + \left[ \frac{xy s_y}{f} R_x - \left( \frac{f}{s_x} + \frac{x^2 s_x}{f} \right) R_y + \frac{y s_y}{s_x} R_z \right] \quad (2.1)$$

$$v = \left[ \frac{y}{Z_W} T_z - \frac{f}{Z_W s_y} T_y \right] + \left[ \left( \frac{f}{s_y} + \frac{y^2 s_y}{f} \right) R_x - \frac{xy s_x}{f} R_y - \frac{xs_x}{s_y} R_z \right]. \quad (2.2)$$

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  $k$ th image in a sequences 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 of  $k$  images. We also assume that for a given  $k$ ,  $\mathbf{p}(k)$  can be found in an area  $\Omega$  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  $\Delta \mathbf{x} = (u, v)^T$  that minimizes the SSD measure

$$\begin{aligned} & (\mathbf{p}(k-1), \Delta \mathbf{x}) = \\ & \sum_{m, n \in N} [I_{k-1}(x(k-1) + m), y(k-1) + n) - I_k(x(k-1) + m + u, y(k-1) + n + v)]^2 \end{aligned} \quad (2.3)$$

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 large displacements while too large an  $N$  enhances the background, causing inaccurate displacements, and increases the associated computational overhead. 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 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 spurious displacement measures. In order to avoid this problem, the system automatically evaluates and selects feature points.

## 2.2 Feature point selection

Feature points selection is based upon a confidence mea-

sure derived using the SSD measure and an auto-correlation technique. The neighborhood  $N$ , centered upon a prospective feature point in image  $I_k$ , is used to collect SSD measures at offsets belonging to the area  $\Omega$ , also in  $I_k$  [1][12]. This produces a surface of SSD measures over the area  $\Omega$ . Several possible confidence measures can be applied to the surface to measure the suitability of the 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 a parabola  $(e(\Delta x_r) = a\Delta x_r^2 + b\Delta x_r + c)$  to a cross-section of the SSD surface in several predefined directions, producing a measure of the goodness of the fit. Papanikolopoulos [12] selected a feature point if the minimum directional measure was sufficiently high. For instance, a corner point produces an SSD surface with very good parabolic fits in all directions. Such a point is selected since the minimal directional measure will be high. A point belonging to an edge would not be selected since the fit in the direction along the edge would be poor due to multiple matches for the prospective feature point.

We extend this approach in response to the aperture problem [7]. 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 the component of motion orthogonal to the edge. Any component of motion due to movement other than that 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 be selected due to the high measure 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. A corner feature would therefore allow manipulator motion in any direction while an edge feature would constrain motion to the direction perpendicular to the edge. This ensures that the only component of resulting motion is precisely the component that can be recovered under consideration of the aperture problem. This ideal motion is not always achievable; therefore, matches that lie closer to the ideal manipulator motion will be preferred, if multiple matches are detected.

## 3 Depth recovery modeling

Several robotic servoing problems require accurate information regarding the depth of the object of interest. When this depth is not provided *a priori*, it must be recovered from observations of the environment via sensing. Traditional depth recovery techniques in a robotic system 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,  $\mathbf{P}$ , located on the surface of the object. The point  $\mathbf{P} = (X_w, Y_w, Z_w)^T$  projects onto a camera's image plane at point  $\mathbf{p}$  with coordinates  $(x, y)$ . By moving the visual sensor, a sequence of images and the respective projections of  $\mathbf{P}$  can be produced. By observing the changes of  $\mathbf{P}$ 's projections,  $\mathbf{p}(k)$  and  $\mathbf{p}(k-1)$ , in two successive images  $(k)$  and  $(k-1)$ , respectively, an estimate of the depth of  $\mathbf{P}$  is derived. Over multiple images, this estimate can be refined using adaptive techniques.

Previous work in depth recovery using active vision relied upon random camera displacements to provide displacement changes in the  $\mathbf{p}(k)$ 's [4][8][9][13][15][16]. In our approach, we select features automatically, produce feature trajectories (a different trajectory for every feature), and predict future displacements using the depth estimate. 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.

### 3.1 Formulation of depth recovery

We begin by modeling the depth recovery problem. We will use the SSD optical flow to measure displacements of feature points in the image plane. For a single feature point,  $\mathbf{p}$ , we assume that the optical flow at time instant  $kT$  is  $(u(kT), v(kT))$ , where  $T$  is the time between two successive frames. It can be shown that at time instant  $kT$ , the optical flow is given by:

$$u(kT) = u_o(kT) + u_c(kT) \quad (3.1)$$

$$v(kT) = v_o(kT) + v_c(kT). \quad (3.2)$$

Since we are recovering the depth of feature points on a static object, the components of optical flow due to the movement of the object ( $u_o(kT)$  and  $v_o(kT)$ ) are zero. Also, equations (3.1) and (3.2) do not account for the computational delays involved with the calculation of the optical flow. By taking these delays into account, and by eliminating the flow components due to object motion, these equations become

$$u(kT) = u_c((k-d+1)T) \quad (3.3)$$

$$v(kT) = v_c((k-d+1)T) \quad (3.4)$$

where  $d$  is the computational delay factor ( $d \in \{1, 2, \dots\}$ ). In order to simplify the notation, the index  $k$  will be used instead of  $kT$ .

By utilizing the relations

$$u(k) = \frac{x(k+1) - x(k)}{T}, \quad v(k) = \frac{y(k+1) - y(k)}{T}$$

the equations can be rewritten as:

$$x(k+1) = x(k) + Tu_c(k-d+1) \quad (3.5)$$

$$y(k+1) = y(k) + Tv_c(k-d+1). \quad (3.6)$$

Furthermore, we can include the inaccuracies of the model (e.g., neglected accelerations, inaccurate robot control) as white noise. The equations (3.5) and (3.6) become

$$x(k+1) = x(k) + Tu_c(k-d+1) + \xi_x(k) \quad (3.7)$$

$$y(k+1) = y(k) + Tv_c(k-d+1) + \xi_y(k) \quad (3.8)$$

where  $\xi_x(k)$  and  $\xi_y(k)$  are zero mean, mutually uncorrelated, stationary random variables with variances  $\sigma_x^2$  and  $\sigma_y^2$  respectively.

The above equations may be written in state-space form as

$$\mathbf{p}(k+1) = \mathbf{A}(k)\mathbf{p}(k) + \mathbf{B}(k-d+1)\mathbf{u}_c(k-d+1) + \mathbf{H}(k)\mathbf{v}(k) \quad (3.9)$$

where  $\mathbf{A}(k) = \mathbf{H}(k) = \mathbf{I}_2$ ,  $\mathbf{p}(k)$  and  $\mathbf{v}(k) \in R^2$ , and  $\mathbf{u}_c(k-d+1) \in R^6$ . The matrix  $\mathbf{B}(k) \in R^6$  is given by:

$$T \begin{bmatrix} \frac{-f}{z_w(k)s_x} & 0 & \frac{x(k)}{z_w(k)} & \frac{x(k)y(k)s_y}{f} & -\left(\frac{f}{s_x} + \frac{x^2(k)s_x}{f}\right) & \frac{y(k)s_y}{s_x} \\ 0 & \frac{-f}{z_w(k)s_y} & \frac{y(k)}{z_w(k)} & \frac{f}{s_y} + \frac{y^2(k)s_y}{f} & \frac{-x(k)y(k)s_x}{f} & \frac{-x(k)s_x}{s_y} \end{bmatrix}. \quad (3.10)$$

The vector  $\mathbf{p}(k) = (x(k), y(k))^T$  is the state vector,  $\mathbf{u}_c(k-d+1) = (T_x(k-d+1), T_y(k-d+1), 0, 0, 0, 0)^T$  is the control input vector (only movement in the x-y plane of the end-effector frame is used), and  $\mathbf{v}(k) = (\xi_x(k), \xi_y(k))^T$  is the white noise vector.

The term,  $\mathbf{B}(k-d+1)\mathbf{u}_c(k-d+1) \in R^6$  can be simplified to

$$\mathbf{B}(k-d+1)\mathbf{u}_c(k-d+1) = T \begin{bmatrix} \frac{-f}{Z_w s_x} T_x(k-d+1) \\ \frac{-f}{Z_w s_y} T_y(k-d+1) \end{bmatrix} \quad (3.11)$$

since  $Z_w$  does not vary over time and only the  $T_x(k-d+1)$  and  $T_y(k-d+1)$  control inputs are used. By substitution into equation (3.9) and simplification, we derive the following:

$$\begin{bmatrix} x(k+1) \\ y(k+1) \end{bmatrix} = \begin{bmatrix} x(k) \\ y(k) \end{bmatrix} + T \begin{bmatrix} \frac{-f}{Z_w s_x} T_x(k-d+1) \\ \frac{-f}{Z_w s_y} T_y(k-d+1) \end{bmatrix} + \begin{bmatrix} \xi_x(k) \\ \xi_y(k) \end{bmatrix}. \quad (3.12)$$

The previous equation is used under the assumption that the optical flow induced by the motion of the camera does not change significantly in the time interval  $T$ . Additionally, the interval  $T$  should be as small as possible so that the relations that provide  $T_x(k-d-1)$  and  $T_y(k-d+1)$  are as accurate as possible. The parameter  $T$  has as its lower bound 16 msec (the sampling rate of standard video equipment). If the upper bound on  $T$  can be reduced such that it is lower than the sampling rate of the manipulator controller (in our case, 28 msec for the PUMA 560), then the system will no longer be constrained by the speed of the vision hardware.

The objective is to design a specific trajectory (a set of desired  $\mathbf{p}^*(k) = (x^*(k), y^*(k))^T$  for successive  $k$ 's) for every individual feature in order to identify the depth param-

ter  $Z_W$ . Thus, we have to design a control law that forces the eye-in-hand system to track the desired trajectory for every feature. Based on the information from the automatic feature selection procedure, we select a trajectory for the feature. For example, the ideal trajectory for a corner feature is depicted in Figure 1. If the feature belongs to a vertical edge, then  $y^*(k)$  is held constant at  $y_{init}^*$ . Then, the control objective function is selected to be

$$F_o(k+d) = E\{(\mathbf{p}_m(k+d) - \mathbf{p}^*(k+d))^T \mathbf{Q} (\mathbf{p}_m(k+d) - \mathbf{p}^*(k+d)) + \mathbf{u}_c^T(k) \mathbf{L}_d \mathbf{u}_c(k)\} \quad (3.13)$$

where  $E\{Y\}$  denotes the expected value of the random variable  $Y$ ,  $\mathbf{p}_m(k)$  is the measured state vector, and  $\mathbf{Q}$ ,  $\mathbf{L}_d$  are control weighting matrices. Based on the equation (3.12) and the minimization of the control objective function  $F_o(k+d)$  (with respect to the vector  $\mathbf{u}_c(k)$ ), we can derive the following control law

$$\mathbf{u}_c(k) = -[\hat{\mathbf{B}}^T(k) \mathbf{Q} \hat{\mathbf{B}}(k) + \mathbf{L}_d]^{-1} \hat{\mathbf{B}}^T(k) \mathbf{Q} \{(\mathbf{p}_m(k) - \mathbf{p}^*(k+d)) + \sum_{i=1}^{d-1} \hat{\mathbf{B}}(k-i) \mathbf{u}_c(k-i)\} \quad (3.14)$$

where  $\hat{\mathbf{B}}(k)$  is the estimated value of the matrix  $\mathbf{B}(k)$ . The matrix  $\hat{\mathbf{B}}(k)$  depends on the estimated value of the depth  $\hat{Z}_W$ . The estimation of the depth parameter is performed by using the procedure described in [12]. In order to use this procedure, we must rewrite equation (3.12) as follows (for one feature point):

$$\Delta \mathbf{p}_m(k) = \zeta_W \mathbf{u}_{cnew}(k-d) + \mathbf{n}(k) \quad (3.15)$$

where

$$\Delta \mathbf{p}_m(k) = \mathbf{p}_m(k) - \mathbf{p}_m(k-1)$$

$$\zeta_W = f/(s_x Z_W)$$

$$\mathbf{n}(k) \sim N(\mathbf{0}, \mathbf{N}(k))$$

$$\mathbf{u}_{cnew}(k) = T \begin{bmatrix} -T_x(k) \\ -s_x T_y(k) \\ s_y \end{bmatrix}$$

The objective of the estimation scheme is to estimate the parameter  $\zeta_W$  for each individual feature. When this compu-

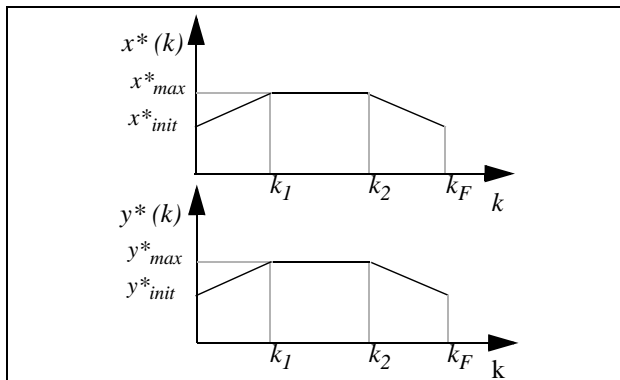


Figure 1: Desired trajectories of the features

tation is completed, it is trivial to compute the parameter  $1/Z_W$  that is needed for the construction of the depth maps. It is assumed that the matrix  $\mathbf{N}(k)$  is constant ( $\mathbf{N}(k) = \mathbf{N}$ ). We use the estimation scheme described in [8][10] (for one feature point)

$${}^p \hat{\zeta}_W(k) = {}^u \hat{\zeta}_W(k-1) \quad (3.16)$$

$${}^p pr(k) = {}^u pr(k-1) + s(k-1) \quad (3.17)$$

$${}^u pr(k) = \left\{ ({}^p pr(k))^{-1} + \mathbf{u}_{cnew}^T(k-d) \mathbf{N}^{-1} \right. \quad (3.18)$$

$$\left. \mathbf{u}_{cnew}(k-d) \right\}^{-1} \quad (3.19)$$

$${}^u \hat{\zeta}_W(k) = {}^p \hat{\zeta}_W(k) + \mathbf{g}^T(k) \{ \Delta \mathbf{p}_m(k) - {}^p \hat{\zeta}_W(k) \mathbf{u}_{cnew}(k-d) \} \quad (3.20)$$

where the superscript  $p$  denotes the predicted value of a variable, the superscript  $u$  denotes the updated value of a variable, and  $s(k)$  is a covariance scalar. The initial conditions are described in [12]. The term  ${}^p pr(0)$  can be viewed as a measure of the confidence that we have in the initial estimate  ${}^p \hat{\zeta}_W(0)$ . The next section describes several simulation and experimental results using this scheme.

## 4 Simulation and experiments

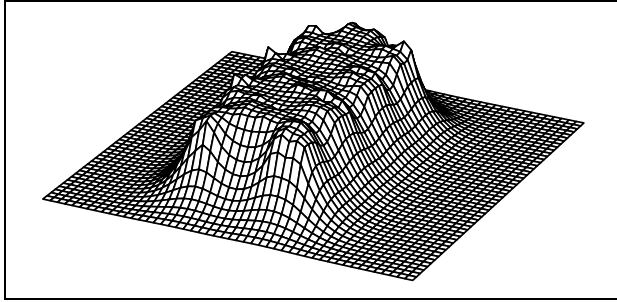
### 4.1 Simulation results

The model was initially verified in simulation experiments. Simulations were performed on a Silicon Graphics Indigo. The camera and manipulator were modeled, as was the target and the environment. The target was displayed in depth and was projected using a pinhole camera model. It appeared in front of a regular textured background that was placed at optical infinity.

The camera was modeled using the perspective transformation of the Indigo's geometry engine to correspond to a 7.5 mm focal length lens and an image plane that was sized to correspond to a 1/2" CCD camera array. The image plane was placed in front of the nodal point of the optical system due to the modeling specifically used by the Indigo.

The manipulator was modeled using second-order partial differential equations that transform the ideal control vector into a realized control vector based upon the current motion vector of the manipulator.

The initial simulation runs were conducted with a soda-can sized cylinder placed 45 cm in depth. The controller began the depth extraction process with an initial depth parameter estimate of 1 meter. The recovery process consisted of a sequence of motions, each producing an image. Once completed, the recovered depth of each feature point was compared to the actual depth to produce a measure of the accuracy of the process. For virtually all feature points, the variation of the recovered versus the computed was subpixel in nature. In fact, the majority of the sub-pixel errors were



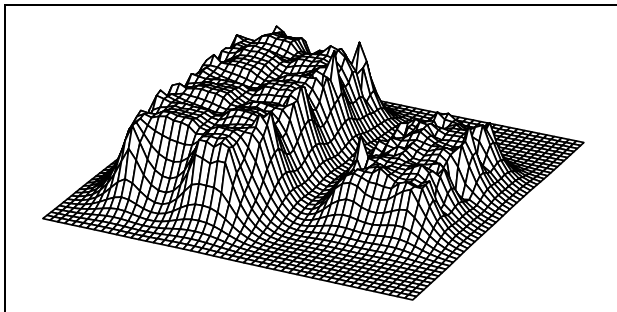
**Figure 2: Single target surface**

less than one half of the error calculated for a given displacement error of one pixel. It should be noted that sub-pixel fitting and multi-grid methods were not employed in this process due to their computational overhead. The reconstructed surface is shown in Figure 2.

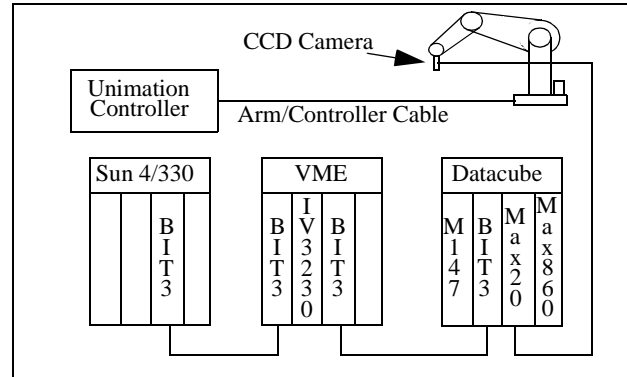
A second set of simulation runs were conducted using two soda-can sized cylinders. The first cylinder was rendered at 45 cm in depth. The second was rendered at 70 cm in depth. Again, a regular textured background was placed at optical infinity. The initial depth parameter estimate was set to 1 meter. The results of the process were again highly accurate, achieving sub-pixel accuracy for virtually all the selected feature points. The result of the surface reconstruction is shown in Figure 3. The reconstruction retains the perspective effect upon the apparent relative size of the targets; thus, the target at greater depth appears smaller.

#### 4.2 Hardware results

Once we determined that the performance of the simulation was satisfactory, the algorithm was implemented on the Minnesota Robotic Visual Tracker (MRVT) (see Figure 4). The system consists of a PUMA 560 manipulator controlled via the Unimation Controller's alter line. The alter line requires path control updates once every 28 msec. Those updates are provided by an Ironics 68030 VME processor that shares its bus with a Sun Sparc 330 and a Datacube vision system via BIT-3 bus extenders. A Panasonic miniature camera is mounted parallel to the end-effector of the PUMA and provides its output to the Datacube for processing. The Datacube system consists of a MVME-147 processor, a MaxVideo 20 video processor, and a Max860 vector processor. The vision sub-system performs the optical flow, calculates the input control vector according to the control law, and sup-



**Figure 3: Dual target surface**



**Figure 4: MRVT system architecture**

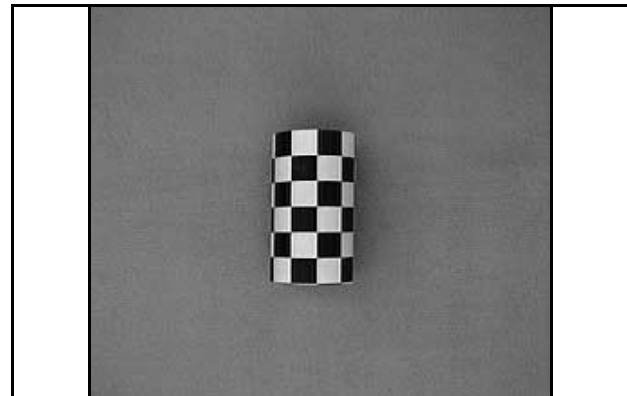
plies the input vector to the Ironics processor for transmission to the Unimate controller.

The initial experimental runs were conducted with a soda-can wrapped in a matte, checkerboard covering and was placed in depth such that the leading edge of the curved surface was 53 cm from the nodal point of the camera (see Figure 5). Similar to the simulation experiments, the initial depth estimate was set to 1 meter of depth. The modelling for the visual process was precisely the same as in the simulation runs; however, the camera scaling factors and the controller gains were changed to meet the specifications of the miniature camera and the PUMA manipulator.

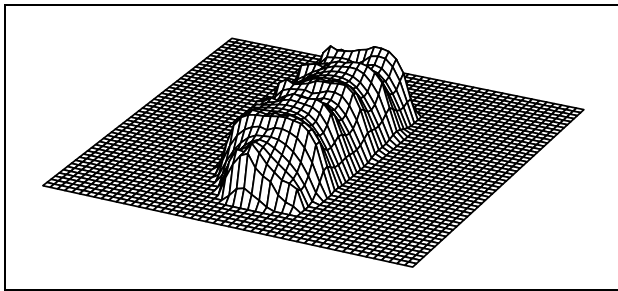
Again, the resulting data for the feature points was not subjected to sub-pixel fitting nor multi-grid methods. It should be noted that, similar to the simulation, the error in the recovered depth of the majority of feature points was sub-pixel in nature. The reconstructed surface of the single soda can is shown in Figure 6.

A second set of experiments was also conducted in the same fashion as the simulations, using two soda cans as targets. The leading edges of the cans were placed at 53 cm and 41 cm in depth (see Figure 7). Again, the majority of the errors in the calculated depths were sub-pixel. For these runs, the initial depth estimate was an underestimate of 25 cm, in contrast to the previous simulations and experiments that used overestimates. The reconstructed surfaces of the dual can experiment are shown in Figure 8.

Further experimental results using real surfaces (rather



**Figure 5: Single can target**



**Figure 6: Single can surface**

than the matte checkerboards) can be found in [14].

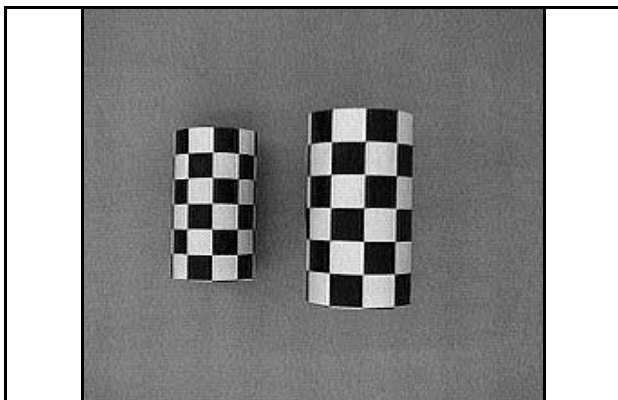
## 5 Conclusion

This paper presents a novel technique for computing depth maps through controlled active exploration with a camera. Unlike similar approaches [6][8][9][13][15], we propose a scheme that is based on automatic selection of features and design of specific trajectories on the image plane for each individual feature. This approach helps us 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 with simulations and experiments, and several experimental results have been presented.

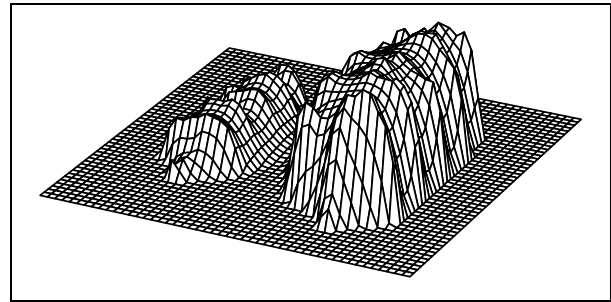
Issues for future research include the automatic selection of the window size (an issue discussed in [11]) in order to select a window that has some texture variations, the computational improvement of the approach, and the implicit incorporation of the robot dynamics.

## 6 Acknowledgments

This work has been supported by the Department of Energy (Sandia National Laboratories) through Contract #AC-3752D, the Center for Advanced Manufacturing, Design, and Control (CAMDAC) of the University of Minnesota, the Center for Transportation Studies through Contract #USDOT/DTRS93-G-0017-01, the 3M Corporation, the



**Figure 7: Two can target**



**Figure 8: Dual can surface**

Graduate School of the University of Minnesota, and the Department of Computer Science at the University of Minnesota.

## 7 References

- [1] P. Anandan, "A computational framework and an algorithm for the measurement of visual motion," *International Journal of Computer Vision*, 2(3):283-310, 1988.
- [2] J.J. Craig, *Introduction to robotics: mechanics and control*, Addison-Wesley, Reading, MA, 1985.
- [3] 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.
- [4] 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.
- [5] J. Heel, "Dynamic motion vision," *Robotic and Autonomous Systems*, 6(3):297-314, 1990.
- [6] 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.
- [7] D. Marr, *Vision*, W.H. Freeman and Company, San Francisco, CA, 1982.
- [8] 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.
- [9] L. Matthies, "Passive stereo range imaging for semi-autonomous land navigation," *Journal of Robotic Systems*, 9(6):787-816, 1992.
- [10] P.S. Maybeck, *Stochastic models, estimation, and control*, Vol. 1, Academic Press, Orlando, FL, 1979.
- [11] M. Okutomi and T. Kanade, "A locally adaptive window for signal matching," *International Journal of Computer Vision*, 7(2):143-162, 1992.
- [12] N. Papanikolopoulos, "Controlled active vision," Ph.D. Thesis, Department of Electrical and Computer Engineering, Carnegie Mellon University, 1992.
- [13] C. Shekhar and R. Chellappa, "Passive ranging using a moving camera," *Journal of Robotic Systems*, 9(6):729-752, 1992.
- [14] C. Smith, S. Brandt, and N. Papanikolopoulos, "Controlled Active Exploration of Uncalibrated Environments," To appear, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1994.
- [15] 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.
- [16] D. Weinshall, "Direct computation of qualitative 3-D shape and motion invariants," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(12):1236-1240, 1991.