J-RA 3/3/ 14999

Error Modeling in Stereo Navigation

L. Matthies
S.A. Shafer

Reprinted from
IEEE JOURNAL OF ROBOTICS AND AUTOMATION
Vol. RA-3, No. 3, June 1987

Error Modeling in Stereo Navigation

LARRY MATTHIES AND STEVEN A. SHAFER, MEMBER, IEEE

Abstract—**In stereo navigation, a mobile robot estimates its position by tracking landmarks with on-board cameras. Previous systems for stereo navigation have suffered from poor accuracy, in part because they relied on scalar models of measurement error in triangulation. Using three-dimensional (3D) Gaussian distributions to model triangulation error is shown to lead to much better performance. How to compute the error model from image correspondences, estimate robot motion between frames, and update the global positions of the robot and the landmarks over time are discussed. Simulations show that, compared to scalar error models, the 3D Gaussian reduces the variance in robot position estimates and better distinguishes rotational from translational motion. A short indoor run with real images supported these conclusions and computed the final robot position to within two percent of distance and one degree of orientation. These results illustrate the importance of error modeling in stereo vision for this and other applications.**

I. Introduction

CONSIDER a robot given the task of going from A to B. At a coarse level its route is planned from a prestored map, while at a fine level the route is determined by sensor information gathered along the way. Incremental motion estimates are integrated to keep track of the robot's position in the map, which in turn is used to predict upcoming landmarks, hazards, or arrival at the destination.

To realize this scenario, a robot needs sensors that can measure its position and detect the presence of three-dimensional (3D) objects nearby. Stereo vision can provide both kinds of information. Stereo matching at one point in time provides a local 3D model for route planning and obstacle avoidance. Selected points in this model become landmarks that are tracked by the stereo system to monitor the robot's progress. Using stereo in this way, to detect nearby objects and to estimate the motion of the robot, is what we refer to as stereo navigation.

We are interested in stereo in this scenario for a number of reasons. First, other motion sensors can be in error, such as shaft encoders when wheels slip or lose contact with the ground. Second, other sensors, such as sonar and radar, can be inappropriate for reasons of concealment, possible confusion with the broadcasts of other robots nearby, or because color and reflectivity information are important. Lastly, we are interested in stereo *per se* and believe that methods developed for this domain can be transferred to other applications.

Manuscript received July 25, 1986; revised December 1, 1986. This work was sponsored in part by the Office of Naval Research under Contract N00014-81-K-0503 and in part by the Defense Advanced Research Projects Agency under Contracts DACA 76-85-C-0003 and F33615-84-K-1520. This work was presented at the ACM/IEEE Fall Joint Computer Conference, Dallas, TX, November 5, 1986.

The authors are with the Computer Science Department, Carnegie-Mellon University, Pittsburgh, PA 15213, USA.

IEEE Log Number 8714999.

Methods for extracting shape and motion information from image sequences can be classified as correspondence-based or flow-based. Correspondence methods [7], [11], [18], [24] track distinct features such as corners and lines through the image sequence and compute 3D structure by triangulation. Flow-based methods [1], [25] treat the image sequence as function $I(x, y, t)$ of row, column, and time, restrict the motion between frames to be small, and compute shape and motion in terms of differential changes in I. This paper deals with error modeling issues in the correspondence paradigm.

One of the first systems for correspondence-based stereo navigation was that built by Moravec [18]. This system moved a robot in a stop–go–stop fashion, digitizing and analyzing images at every stop. Features were matched in stereo images to build a world model consisting of 3D points. After moving and acquiring more images, the points in the world model were matched in the new images to find their coordinates relative to the new robot location. A least squares procedure was applied to the differences between the new and old point locations to infer the actual motion of the robot. The contribution of each landmark point to this motion estimate was multiplied by a scalar weight that varied inversely with the distance to the point.

In earlier work with Moravec [17], we found the motion solving part of this system to be somewhat inaccurate and unstable. This has been a common experience with visual motion solving algorithms in general. In the case of correspondence-based algorithms, this can partly be attributed to inadequate modeling of measurement error in triangulation. In triangulation, 3D coordinates are computed by intersecting rays projected through corresponding points in two images. Errors in locating the image points induce errors in the 3D coordinates, which in turn cause errors in motion estimates based on the 3D information. Modeling the measurement errors can reduce their effect on motion estimates. However, we will demonstrate that using scalar weights to model uncertainty in 3D coordinates leads to poor performance.

More sophisticated methods have been used in a number of places. In photogrammetry [20], two-dimensional (2D) and 3D normal distributions are used to model error in image coordinates and 3D point locations, respectively. Gennery [11] has used 2D normal distributions of image coordinates in camera calibration for computer vision. Hallam [15] used normal error models in conjunction with Kalman filters to track points and estimate robot motion from sonar data. Broida and Challeppa [5] used similar methods to track a known object in monocular image sequences, and Faugeras [9] has discussed the application of these methods to stereo.

This paper shows how these methods can be applied to

0882-4967/87/0600-0239$01.00 © 1987 IEEE

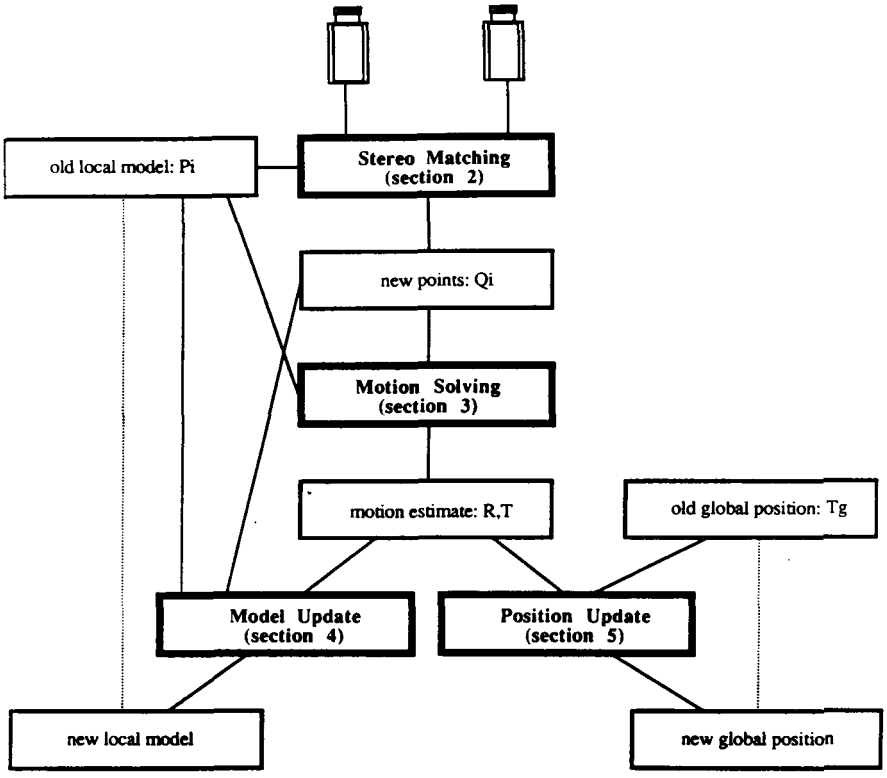

Fig. 1. System block diagram.

stereo navigation and demonstrates by results with real images that they lead to markedly better performance. The system we will describe has evolved from Moravec's [18] and is shown in Fig. 1. The main data structures are a set of 3D points P_i, called the local model and described in robot-centered coordinates, and the robot's current estimate of its position in some fixed global reference frame. The points in the local model are obtained by stereo matching and are used as landmarks. When a new stereo pair is digitized, points from the local model are matched in the images to determine their current locations Q_i relative to the robot. A motion solving algorithm estimates the rotation and translation (R and T) relating the new and old coordinates. The model updating system transforms the old local model into the current coordinate frame and combines it with the new points to create a new local model. Finally, the motion estimate is used to update the robot's global position. The cycle then repeats with the acquisition of a new pair of images.

Section II shows how to model triangulation error in the stereo matcher with 3D normal distributions. In Section III this is incorporated in an algorithm for finding the rotation and translation between successive stereo pairs. The covariance matrix of this transformation is used in Section IV to update the local model with Kalman filters and in Section V to estimate the robot's global position uncertainty. Simulations described in Section VI show that compared to scalar error models this system reduces the variance of position estimates and better distinguishes rotational motion from translation. An experiment with real images, using 54 stereo pairs covering 5.4 m and fully automatic feature tracking, supported these conclusions and computed the final robot position to within two percent of distance and one degree of orientation. Conclusions are summarized in Section VII.

II. Modeling Stereo Triangulation Error

The geometry of stereo triangulation is shown schematically in Fig. 2 for the case of 2D points projecting onto one-dimensional (1D) images. The tick marks on the image planes denote pixel boundaries, and the radiating lines extend these boundaries into space. Suppose point P projects onto the left image at x_l and the right image at x_r. Because of errors in measurement, the stereo system will determine x_l and x_r with some error, which in turn causes error in the estimated location of P. Fig. 2 illustrates this for errors caused by image quantization; because of resolution limits, the estimated location of P can lie anywhere in the shaded region surrounding the true location [22]. Random contributions to measurement error will blur the boundaries of this region, but the qualitative shape will be similar. We want to take this uncertainty into account in any reasoning based on measurements of P.

Three approaches to modeling such uncertainty are discrete tolerance limits, scalar weights, and multidimensional probability distributions. Tolerance regions have been used in object recognition to test candidate model to image matches [14] and to constrain three-dimensional relationships between objects [4], [6]. For example, Baird [4] used tolerance regions in finding the transformation between a two-dimensional set of model points and their measured image positions. Uncertainty was represented with convex polygons surrounding the measured point locations, and the transformed model points were required to lie within these polygons. Acceptable transforma-

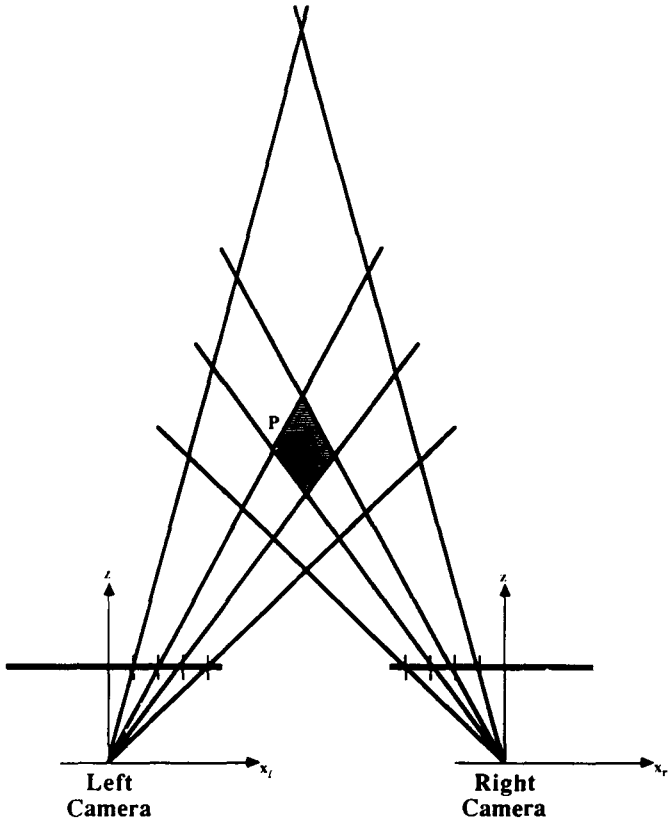

Fig. 2. Stereo geometry showing triangulation uncertainty.

tions were found by linear programming. In our application, statistical minimization and methods are more appropriate because of the stochastic nature of measurement errors and the need to filter time sequences of measurements.

The motivation for using scalar weights is that uncertainty grows with distance, so it can be modeled by weighting points inversely with distance [18]. However, as Fig. 2 shows, the uncertainty induced by triangulation is not a simple scalar function of distance to the point; it is also skewed and oriented. Nearby points have a fairly compact uncertainty, whereas distant points have a more elongated uncertainty that is roughly aligned with the line of sight to the point. Scalar error measures do not capture these distinctions in shape.

These distinctions can be captured by using 3D probability distributions to characterize the uncertainty in point locations. Our approach is to assume 2D, normally distributed (i.e., Gaussian) error in the measured image coordinates and to derive 3D Gaussian distributions describing the error in the inferred 3D coordinates. Similar approaches have been used in photogrammetry [20] and elsewhere in computer vision [11], [5], [12], [15], [9]. The use of Gaussian distributions to model image coordinate error is a common [11], [5], convenient approximation that gives adequate performance, as will be seen in Section VI. For the 3D coordinates, the true distribution *will* be non-Gaussian because triangulation is a nonlinear operation; we approximate this as Gaussian for simplicity and because it gives an adequate approximation when the distance to points is not extreme. We will discuss shortly the cases where this breaks down.

We will now show the details of the triangulation and error model calculation for the general case of 3D points projecting onto 2D images. We assume a camera geometry with parallel image planes, aligned epipolar lines, and image coordinate systems centered at the piercing point of each camera. Let the image coordinates be given by $l = [x_l, y_l]$ and $r = [x_r, y_r]$ in the left and right image, respectively. Consider these as normally distributed random vectors with means μ_l and μ_r and covariance matrices V_l and V_r. From l and r we need to estimate the coordinates $[X, Y, Z]^T$ of the 3D point P. We take the simple approach of using the ideal noise-free triangulation equations $P = [X, Y, Z]^T = f(l, r)$, or

$$X = b(x_l + x_r)/(x_l - x_r)$$
$$Y = b(y_l + y_r)/(x_l - x_r)$$
$$Z = 2b/(x_l - x_r) \tag{1}$$

(assuming a unit focal length and a baseline of $2b$) and inferring the distributions of X, Y, and Z as functions of random vectors l and r. If (1) was linear, P would be normal [8] with mean $\mu_p = f(\mu_l, \mu_r)$ and covariance

$$V_p = J \begin{bmatrix} V_l & 0 \\ 0 & V_r \end{bmatrix} J^T \tag{2}$$

where J is the matrix of first partial derivatives of f or the Jacobian. Since f is nonlinear these expressions do not hold exactly, but we use them as satisfactory approximations.

The true values of the means and covariances of the image coordinates needed to plug into (1) and (2) are unknown. We approximate the means with the coordinates returned by the stereo matcher and the covariances with identity matrices. This is equivalent to treating the image coordinates as uncorrelated with variances of one pixel. Better covariance approximations can be obtained by several methods [2], [11].

What does this error model mean geometrically? Constant probability contours of the distribution of P describe ellipsoids about the nominal mean that approximate the true error distribution. This is illustrated in Fig. 3 where the ellipse represents the contour of the error model and the diamond represents quantization error of Fig. 2. For nearby points the contours will be close to spherical; the farther the points, the more eccentric they become. A covariance matrix with structure $V = wI$, equal to a scalar times the identity matrix, describes only spherical contours. This is the difference between attaching scalar weights to 3D coordinate vectors and using the full 3D distribution; that is, scalar weights are equivalent to spherical covariances whereas the full distribution permits ellipsoidal covariances. In the balance of the paper we will often refer to scalar weights as a spherical error model and the full distribution as an ellipsoidal error model.

Where the Gaussian approximation breaks down is in failing to represent the longer tails of the true error distribution. The true distribution is skewed not unlike the diamond in Fig. 3, whereas normal distributions are symmetric. The skew is not significant when points are close, but becomes more pronounced the more distant the points. A possible consequence is

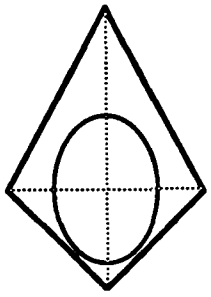

Fig. 3. Quantization error with normal approximation.

biased estimation of point locations, which may lead to biased motion estimates. We will return to these issues in Section VI.

III. Solving for Robot Motion

The previous section showed how to model measurement error in stereo triangulation. In this section we show how to incorporate the error model into an algorithm for estimating the motion between successive stereo pairs. We will begin by showing how motion is computed with scalar weights, then derive an algorithm based on the 3D Gaussian error model, and finally give this algorithm a geometric interpretation.

Referring back to Fig. 1, at this stage in the cycle the robot has two sets of 3D points that have been obtained by stereo matching: a local model of points P_i defined relative to its previous position and the coordinates Q_i of these points relative to its current position. The correspondences between P_i and Q_i are known, but the motion between them is not. Thus we have a set of equations

$$Q_i = RP_i + T$$

in which P_i and Q_i are known point vectors, R is the matrix of the unknown rotation, and T is the unknown translation.

Using scalar weights, one finds R and T by expressing the errors of fit by

$$\epsilon_i = Q_i - RP_i - T$$

and minimizing the weighted sum of squares

$$\sum_{i=1}^{n} w_i \epsilon_i^T \epsilon_i \tag{3}$$

where w_i are the weights. Although the rotation makes this optimization problem nonlinear, two methods are known that give the solution essentially in closed form. The method we have used is due to Schonemann [19]. It treats the nine elements of R as unknowns and applies Lagrange multipliers to force R to be orthogonal. The only iterative part of the algorithm involves taking the singular value decomposition of a 3×3 matrix. Readers are referred to [19] for details. The alternate method, described in [16] and [26, p. 426], parameterizes the rotation as a quaternion and obtains the quaternion elements as the eigenvector corresponding to the largest eigenvalue of a 4×4 matrix.

As will be shown in Section VI, the scalar model of uncertainty embodied in (3) leads to poor performance. Using the 3D Gaussian error model the solution takes a similar, but more complicated form. For simplicity we begin with the case of translational motion. In this case the motion equation is

$$Q_i = P_i + T$$

which we may rewrite as

$$Q_i - P_i = M_i = T$$

to emphasize the role of $M_i = Q_i - P_i$ as measurements of T. From Section II, P_i and Q_i are modeled as normally distributed, uncorrelated random vectors with covariances U_i and V_i, respectively. Therefore, M_i will also be normally distributed with covariance $U_i + V_i$. Now if we consider M_i to be a sequence of noisy measurements of T, each corrupted by noise with zero mean and covariance $U_i + V_i$, application of the maximum likelihood method leads to minimizing the following expression over possible values of T [8]:

$$\sum_{i=1}^{n} \epsilon_i^T W_i \epsilon_i \tag{4}$$

where $\epsilon_i = M_i - T$ and $W_i = (U_i + V_i)^{-1}$. The solution to this is

$$T = \left(\sum_{i=1}^{n} W_i \right)^{-1} \sum_{i=1}^{n} W_i M_i$$

and the covariance matrix of the estimation errors is

$$V_T = \left(\sum_{i=1}^{n} W_i \right)^{-1}.$$

The covariance matrix can be analyzed to assess the quality of the motion estimate. It is also used later in modeling the uncertainty of the robot's global position estimate.

An intuitive interpretation of (4) is shown in Fig. 4. The weight matrices W_i function as norms that measure distance differently for each point. Error vectors making equal contributions to the total error of fit lie on ellipsoidal contours. For example, in Fig. 4, residuals ϵ_a and ϵ_b contribute equally to the total error, but ϵ_c contributes more because $\epsilon_a^T W \epsilon_a = \epsilon_b^T W \epsilon_b < \epsilon_c^T W \epsilon_c$. This effectively gives more weight to errors perpendicular to the line of sight than parallel to it, which, given the nature of stereo, is what we would like to do. The "spherical" error model obtained by using the scalar weights of (3) has the obvious mnemonic meaning that residual vectors making equal contributions to the total error lie on spherical contours. This distinction is what gives the ellipsoid model its power.

Generalizing this method to handle rotation is complicated by the fact that the equations become nonlinear. The function to be optimized takes the form

$$\sum_{i=1}^{n} \epsilon_i^T W_i \epsilon_i \tag{5}$$

with $\epsilon_i = Q_i - RP_i - T$ and $W_i = (RU_iR^T + V_i)^{-1}$.

We have not been able to find direct solutions to this problem or even to approximations in which W_i is not a

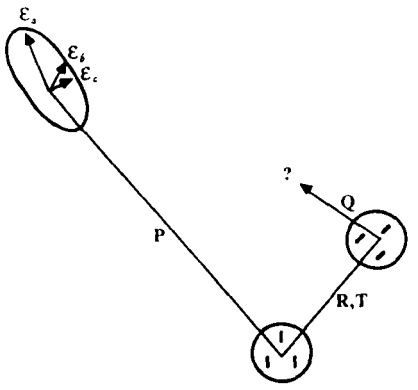

Fig. 4. Interpretation of (4): W_i scales residual vectors, lengthening them parallel to line of sight and shortening them perpendicularly to it.

function of R. Our approach has been to use the direct solution of Schonemann [19] for scalar weights to get an initial estimate of the transformation and to apply the Gauss–Newton method [13, p. 134] to (5) to refine iteratively the estimate. Convergence behavior is good unless all points are very distant; for example, in the experiments with real data described later, the final estimates were obtained after four to eight iterations.

To recap, this section incorporated the error model of Section II in an algorithm for finding the rotation and translation between two 3D points sets. The algorithm replaces the scalar weights of (3) with weight matrices based on the covariances of corresponding points. When the motion is purely translational, the problem is linear and has a direct solution, but when the motion involves rotation we resort to an iterative solution. The error covariance of the motion solution will be used in the following two sections in updating the robot's local model and global position estimate.

IV. Updating the Local Model

So far we have described how to model error in triangulation and how to solve for the motion between two successive stereo pairs. This section deals with how to process a long sequence of stereo pairs. At issue is how to average information from successive images to achieve more accurate landmark localization and consequently more accurate estimates of robot position.

An appropriate tool for this is the Kalman filter [10]. In filtering terminology the quantity to be estimated is called the "state," and when a measurement is taken the filter updates the current estimate of the state. Kalman filters incorporate known statistical properties of the measurements into the update process and produce error covariances for the state estimate. They are widely used in terrestrial and aerospace navigation and guidance applications [10], [26]. In computer vision they have been used in object recognition [3], tracking of known objects with monocular image sequences [5], [12], and for robot navigation and object tracking with sonar data [15].

In our application, the state consists of the locations of the landmark points in the local model. A question arises as to whether the landmarks should be represented in a global stationary frame of reference or in a local moving robot-centered frame. In either case, the update involves transforming coordinates from one frame to the other and applying the filter. If a fixed number of landmarks are being tracked, there is no difference in cost between the two. There will be a difference in the uncertainty of the resulting model; this difference depends on the relative uncertainties of the old model, the new measurements, and the intervening motion. We have not completed an analysis of this situation, but are currently keeping the landmark model in robot-centered coordinates.

The update involves transforming the old local model to the current coordinate frame, inflating its uncertainty to account for the uncertainty of the transformation, and filtering the old model with the new measurements to create the updated model. Let P_{t-1} be the coordinate vector of a single point in the old local model at time $(t-1)$, and let V_{t-1} be its covariance. For purely translational motion, P_{t-1} is transformed to the current frame by

$$\mathcal{P}_{t-1} = P_{t-1} + T \tag{6}$$

where T is the translation from time $(t-1)$ to time t. The translation has an error covariance matrix V_T so the transformed point has covariance

$$\mathcal{V}_{t-1} = V_{t-1} + V_T. \tag{7}$$

Equation (6) introduces some correlation between points that is not accounted for in (7), but we assume this is small enough to ignore. To extend this to rotation, we rewrite (6) as

$$\mathcal{P}_{t-1} = RP_{t-1} + T. \tag{8}$$

This is nonlinear, so to compute $\mathcal{V}_{t-1}$ we proceed by analogy to (2); that is, we premultiply the covariance of R, T, and P_{t-1} by the Jacobian of the transformation and postmultiply by the Jacobian transposed. Since we treat P_{t-1} as uncorrelated with R and T, this leads to

$$\mathcal{V}_{t-1} = J_m V_m J_m^T + R V_{t-1} R^T$$

where J_m contains the derivatives of (8) with respect to the motion parameters and V_m is the covariance of the motion parameters.

Now let Q_t be the measurement of the same point at time t, and let U_t be the covariance of this measurement. Some manipulation of the basic Kalman filter equations leads to the following estimates of the updated point location and covariance:

$$V_t = (\mathcal{V}_{t-1}^{-1} + U_t^{-1})^{-1} \tag{9}$$

$$P_t = \mathcal{P}_{t-1} + V_t U_t^{-1} (Q_t - \mathcal{P}_{t-1}). \tag{10}$$

The intuition behind (10) is as follows. The second term takes the difference $(Q_t - \mathcal{P}_{t-1})$ of the new measurement from the old estimate, weights the difference by $V_t U_t^{-1}$, and applies the result as an update to the old estimate $\mathcal{P}_{t-1}$. Matrix U_t^{-1} will be "larger" the more precise the new measurement, giving it more weight in the update, and smaller the less precise the measurement, giving it less weight. Conversely, V_t will be small if the old estimate is precise and large otherwise. Hence

if the old estimate is already good, the new measurement receives little weight; if it is poor, the new measurement receives more weight.

The procedure we have described assumes that the error in the motion estimate is uncorrelated with the error in the landmark points. When the motion estimate is obtained by using the methods of the previous section this will not be true, although if other sensors are also contributing to the motion estimate, it will be approximately true. This is an issue we are investigating.

V. Updating the Global Robot Position

By using the modules discussed in the previous sections, the robot computes estimates of its motion between successive stereo pairs. Combining these to estimate its global position is a simple matter of concatenating the transformation matrices. It may also be desirable to estimate the uncertainty of the global position, which can be done by propagating the covariance matrices of the incremental motions into a covariance of the global position. For translation this is also very simple. If the global position at time $(t-1)$ is $T_{g_{t-1}}$ and the next incremental translation is T_t, then the next global position is

$$T_{g_t} = T_{g_{t-1}} + T_t. \tag{11}$$

Since this is linear, if the incremental translation estimates have uncorrelated zero-mean Gaussian errors, then T_{g_t} will also have zero-mean, Gaussian error with covariance given by

$$V_{g_t} = V_{g_{t-1}} + U_t$$

where $V_{g_{t-1}}$ and U_t are the covariances of $T_{g_{t-1}}$ and T_t, respectively. The case of motion in the plane, where there are two parameters for translation and one for rotation, has been dealt with by Smith and Cheeseman [21]. In summary, one obtains an equation analogous to (11) in which the three parameters of the global position are expressed as functions of the previous position and the incremental motion. These are nonlinear and error propagation is done by linearization. For general motion in three dimensions, this is not straightforward with the Euler angle representation of rotation we have used to date. In this case other parameterizations of rotation, such as quaternions, may be preferable [9], [26]. We are exploring this further.

VI. Performance

Our evaluation to date has concentrated on comparing the use of the spherical and ellipsoidal error models in the motion solving methods of Section III. Results of tests with simulated and real data are described below.

A. Simulations

Three sets of simulation data will be presented. The first is a base case that compares the standard deviations of position estimates obtained with each error model for a single step of vehicle motion. That is, it considers motion between only two consecutive stereo pairs. It illustrates the difference in the variability of position estimates with each model and reveals the effects on the motion estimates of coupling between the translational and rotational degrees of freedom. The second set also considers only two consecutive stereo pairs and tests limiting performance by tracking progressively more distant points. The last set examines both long-range performance over many images and the effect on performance of different stereo baselines.

The simulations were generated as follows. The "scene" consisted of random points uniformly distributed in a 3D volume in front of the simulated cameras. For the first set of simulations, this volume extended 5 m to either side of the cameras, 5 m above and below the cameras, and from 2 to 10 m in front of the cameras. The cameras themselves were simulated as having 512 × 512 pixels and a field of view of 53°. The stereo baseline was 0.5 m. Image coordinates were obtained by projecting the points onto the images, adding Gaussian noise to the floating point image coordinates, and rounding to the nearest pixel. These coordinates were input to the triangulation and motion solving algorithms. For the ellipsoidal error model, covariance matrices were computed as described in Section II. In the scalar case, weights were derived by taking the Z variance from the covariance matrix. Scalars obtained by several other methods were tried and found to give very similar results. These include the volume and length of the major axis of the standard error ellipsoid and Moravec's half-pixel shift rule [18].

The first set of simulations determined the standard deviation of the estimated motion between two consecutive stereo pairs when the true motion was 1 m. The results are given in Figs. 5 and 6, plotted against the number of points used to compute the motion estimate. For any given number of points tracked, the standard deviations are taken over 5000 random trials with entirely new points generated for each trial. In both figures, the top three curves were obtained with spherical modeling and the bottom three with ellipsoidal. Tilt implies rotation of the camera up or down, pan is the rotation about the vertical axis, and roll the rotation about the camera axis. The most significant thing to note is that the standard deviations obtained with the ellipsoidal model are a factor of 5–10 less than those of the spherical model. The size of the difference will vary with the distance to the points; for example, when they are within 1–2 m of the cameras the factor is 2–4, and when they are within 2–5 m it is 3–6. The case shown in the figures (points from 2–10 m away) approximates the conditions of the indoor run with real data described later. Another point to note is that with the spherical model the estimates of roll and forward translation show less variation than the remaining parameters. This is because lateral translations and panning rotations have coupled effects on the errors of fit, as do vertical translations and tilting rotations. This shows up in the covariance matrix of the computed motion parameters as larger correlations between these pairs of parameters than other pairs. These correlations are present with both error models, but the effects on the variance of the individual parameters are greater in the spherical case. Lastly, note that for a given level of performance fewer points are needed with the ellipsoidal model than the spherical, offsetting the greater expense of the iterative motion solution needed in the

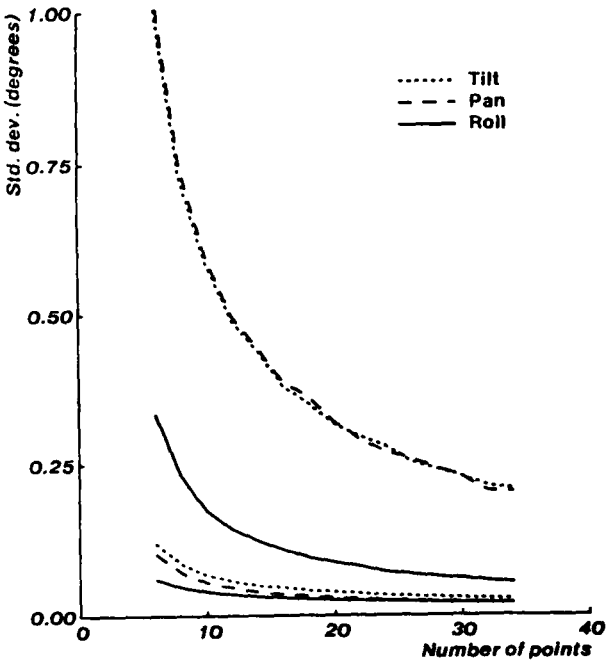

Fig. 5. Standard deviation versus number of points for rotations. Top three curves are for spherical model, bottom three are for ellipsoidal model. Use of ellipsoidal model gave significantly lower variance in estimates.

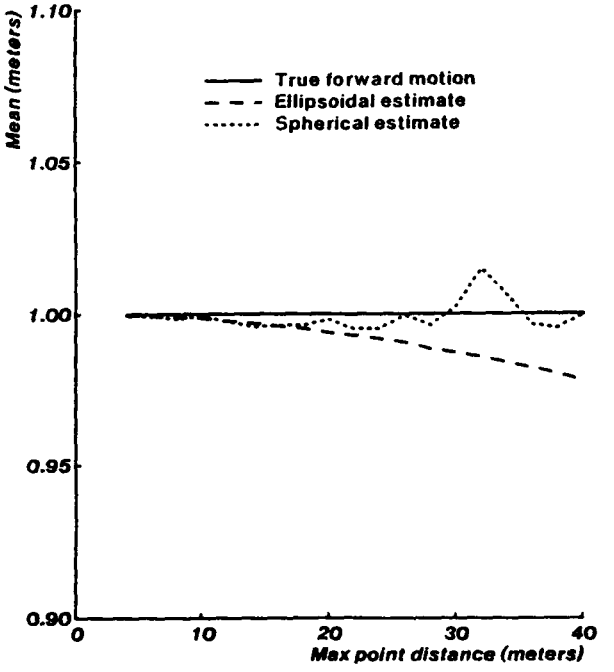

Fig. 7. Mean of estimated forward distance traveled versus maximum distance to points.

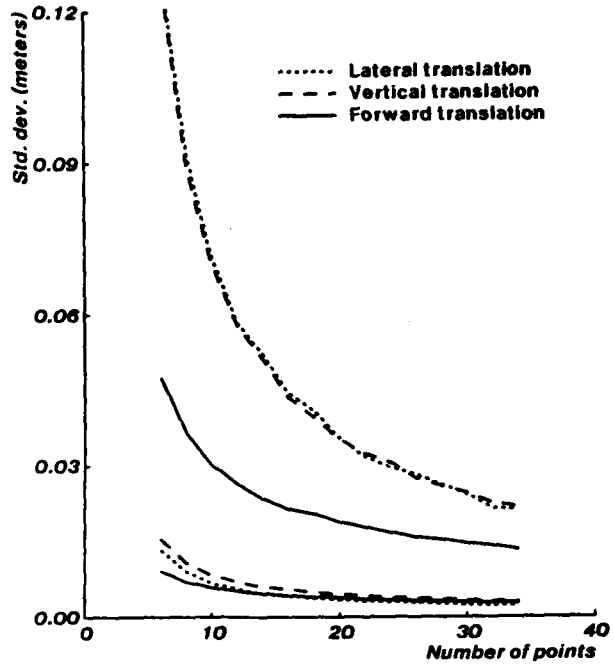

Fig. 6. Standard deviation versus number of points for translations. Top three curves are for spherical model, bottom three, are for ellipsoidal model.

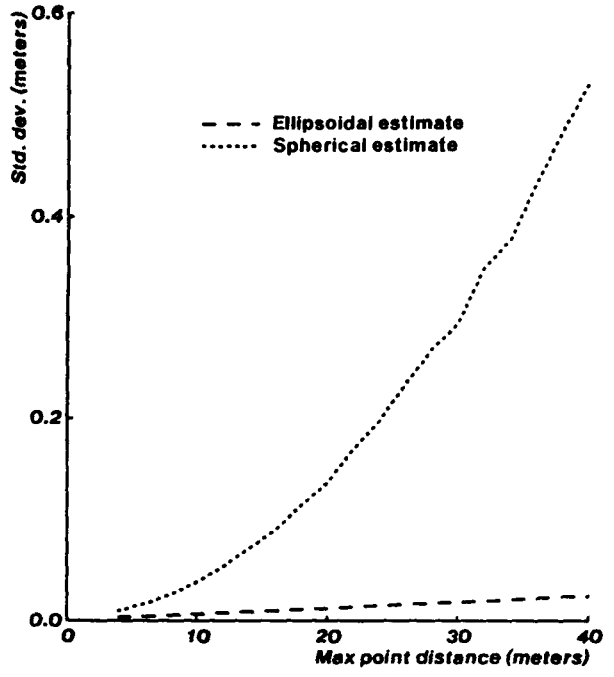

Fig. 8. Standard deviation of estimated forward distance traveled versus maximum distance to points.

ellipsoidal case. The exact relationship will depend on the camera configuration.

The second set of simulations illustrates the dependence of the standard deviation on the distance to the points in the scene. The initial volume for generating points was 2–4 m away; this was expanded by moving the far limit back in stages until the final volume was 2–40 m. As with the previous experiment, for each volume 5000 random trials were performed with different points generated for each trial. Fig. 7 shows the mean of the forward translation estimates as a function of the maximum distance to the points, and Fig. 8 shows the standard deviation. The true forward motion was one meter. The standard deviation tells most of the story. With the ellipsoidal model, the standard deviation remains modest throughout the range of the experiment, reaching a maximum of about three percent of the actual motion. On the other hand, with the spherical model the standard deviation is initially modest but grows rapidly to the point that the estimates are

unusable. The other motion parameters, though not shown, behave similarly. Looking at the means, with the ellipsoidal model there is negligible bias when points are nearby, with a growing tendency to underestimate the distance traveled as the points themselves become more distant. For the spherical model there also appears to be some underestimation when points are nearby, but the rapid growth of the standard deviation makes further interpretation of little value. Thus this experiment illustrates the strong contrast between the algorithms that develops with increasing distance to points.

The last simulation looked at motion over a long sequence of images, both to confirm the above results and to test a hypothesis suggested by the previous simulation: that for equivalent performance, the ellipsoidal model may permit the use of a shorter stereo baseline than the spherical. This is an important consideration, because length of the baseline directly affects the difficulty of stereo matching. Each trial in this experiment involved tracking points 2–10 m from the cameras, with new points added when existing ones passed out of view. Fig. 9 shows the standard deviation of the estimated distance as a function of the true distance. The travel between images was 0.64 m, so the figure represents about 90 images. It shows curves for a 0.5-m baseline with the spherical model and 0.125-, 0.25-, and 0.5-m baselines for the ellipsoidal model. Comparing the curves for 0.5-m baselines, the ellipsoidal model does outperform the spherical. It appears that the curves may eventually run parallel, so that the difference between the methods would be an additive constant rather than multiplicative. Looking at the effects of different baselines, results with the ellipsoidal model are still better than the spherical model with a 0.25-m baseline, though not with 0.125-m. Based on standard deviations of position, it does appear possible to use a shorter baseline. However, another factor involved is bias of the motion estimates. In general, we have found that the narrower the baseline, the more motion is underestimated. The same occurs when we increase the variance of the noise in the image coordinates. This requires further investigation. For the moment we just note that bias can be a problem with short baselines or nontrivial noise levels.

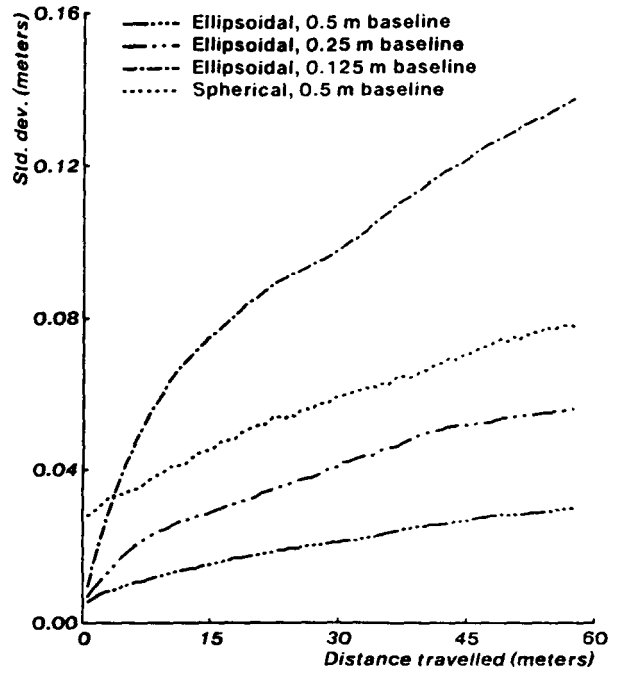

Fig. 9. Standard deviation of estimated forward distance traveled versus true distance.

Fig. 10. One image from lab sequence.

B. Real Images

To verify the simulations on real images, we used both error models to estimate the position of a stereo-equipped robot traveling across the floor of our lab. The scene is pictured in Fig. 10. The robot was driven straight forward in 54 steps of slightly less than 10 cm each. The cameras were on a 20-cm baseline and had a 36° field of view. The FIDO feature-tracking system [23] was used to track points through the image sequence, and the resulting set of matched image coordinates were input to the algorithms described earlier to estimate the robot's position at each step. We will briefly describe the operation of FIDO before discussing the results of the experiment.

FIDO uses the Moravec interest operator and coarse-to-fine correlation algorithm to pick and match point features in stereo pairs. The interest operator is applied to one image of a stereo pair to pick points where intensity varies in all directions; typically these are sharp corners or intersections of lines. The correlator finds these points in the other image of the stereo pair. To find the same points in subsequent stereo pairs, an *a priori* motion estimate is used to predict the location of the point in the new images, a constraint window is defined around the predicted location based on the uncertainty of the motion estimate, and the correlator is applied to find the position of best match within the constraint window. Incorrect matches are culled with a threshold on the correlation coefficient and with a 3D error heuristic called the "3D prune" stage. This heuristic uses the fact that under rigid motion the distance between two 3D points does not change over time. Points which appear to violate this condition are discarded. The advantage of this test is that it does not require knowledge of the motion between stereo pairs. Points that survive this test become input to the motion solving al-

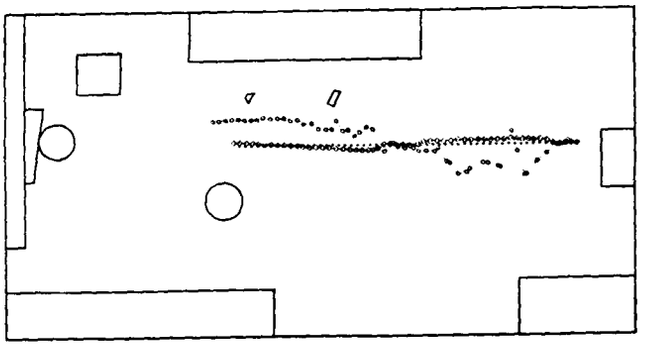

Fig. 11. Position estimates obtained with 3 DOF algorithm and clean data. Dots show actual vehicle positions, diamonds show positions estimated with ellipsoidal model, and circles show positions estimated with spherical model.

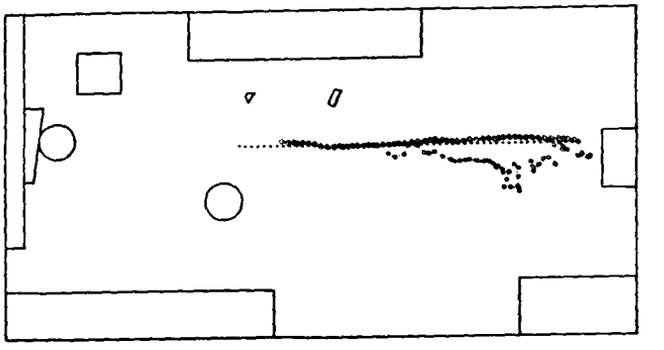

Fig. 12. Results with noisy data. As in Fig. 11, dots show actual vehicle positions, diamonds show ellipsoidal estimates, and circles show spherical estimates.

gorithms. In the experiments to follow, between 30 and 40 points usually remained.

Fig. 11 compares the true motion to the position estimates obtained with the spherical and ellipsoidal error models. For this figure a "planar" motion solver was used that solved only for the parameters of motion in the plane, that is two degrees of translation and one of rotation. The line of heavy dots shows the true position at every step, the path marked with circles shows the positions estimated with the spherical model, and the path marked with diamonds shows the same for the ellipsoidal model. The final position estimated with the ellipsoidal model was correct to within two percent of the distance and 1° of orientation. With the spherical model the corresponding figures were eight percent and 7°.

To gauge the effect of noisier image matches, we adjusted the threshold of the prune stage so that progressively fewer points were discarded. The general effect was to increasingly underestimate the distance traveled. Fig. 12 shows what happened when the prune stage was entirely disabled, leaving only the correlation threshold to detect matching errors. Estimates with the spherical model were initially very bad. We attribute this to matching errors caused by large depth discontinuities around the foreground objects. When these objects fell out of view, the estimates were better behaved. The behavior with the ellipsoidal model was much less erratic.

Finally, we repeated the first experiment (i.e., clean data) with the algorithm that computes all six degrees of freedom (DOF) of motion. The results were in accord with the planar case, with roughly the same levels of error in the final position estimate. It was notable that with the spherical model the error in roll was less than a degree, while in the other rotations it was between 5° and 12°. This is consistent with the observation made from the first simulation about coupled rotation and translation.

VII. Conclusion

Comparing motion estimates obtained with the spherical (scalar) and ellipsoidal (3D Gaussian) error models, there is no question that the ellipsoidal model is preferred. Simulations showed that position estimates with the ellipsoidal model had less variance and live trials confirmed that they were more accurate and less influenced by matching errors. The contrast between algorithms is strongly influenced by the distance to the points being tracked; with nearby points, the difference will be moderate, but it grows very rapidly with increasing distance.

The possibility of bias arose with very large distances to objects and high noise levels. We attribute this to the non-Gaussian nature of the true error distribution in these situations. Under these conditions, better error modeling is an area for further research. The question of whether the ellipsoidal method permits a shorter baseline has only been tested in simulation; based on the variance of the estimates it appears feasible, but the bias issue is unresolved.

Perhaps the most valuable result is demonstrating that accurate position estimates can be achieved in a fully automatic system when an adequate error model is used. The true motion in the examples we showed was pure translation, but we believe that the results will hold for general motion and preliminary simulations bear this out. With matching to subpixel resolution, matching of extended features instead of points, and more sophisticated error detection, it may be possible to obtain much better performance than that quoted here. Another interpretation of our results is that they show the importance of error modeling in stereo and probably other aspects of vision. One area we plan to explore this is in shape from stereo, beginning with the local update paradigm of Section V.

Acknowledgment

We are indebted to Hans Moravec for making us aware of Kalman filters, Peter Highnam for introducing us to Schonemann's algorithm, and Takeo Kanade for pointing out the possibility of using a shorter baseline.

References

[1] G. Adiv, "Determining three-dimensional motion and structure from optical flow generated by several moving objects," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-7, pp. 384–401, July 1985.

[2] P. Anandan and R. Weiss, "Introducing a smoothness constraint in a matching approach for the computation of displacement fields," in *Proc. ARPA IUS Workshop*, SAIC, Dec. 1985, pp. 186–197.

[3] N. Ayache and O. D. Faugeras, "HYPER: A new approach for the recognition and positioning of two-dimensional objects," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-8, pp. 44–54, Jan. 1986.

[4] H. S. Baird, *Model-Based Image Matching Using Location*. Cambridge, MA: MIT Press, 1985.
[5] T. J. Broida and R. Chellappa, "Estimation of motion parameters from noisy images," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-6, pp. 90–99, Jan. 1986.
[6] R. A. Brooks, "Symbolic reasoning among 3-D models and 2-D images," *Artificial Intell.*, vol. 17, pp. 285–348, 1981.
[7] L. Dreschler and H.-H. Nagel, "Volumetric model and 3D trajectory of a moving car derived from monocular TV frame sequences of a street scene," *Comput. Graph. Image Processing*, vol. 20, pp. 199–228, 1982.
[8] T. F. Elbert, *Estimation and Control of Systems*. New York: Van Nostrand Reinhold, 1984.
[9] O. D. Faugeras, N. Ayache, B. Faverjon, and F. Lustman, "Building visual maps by combining noisy stereo measurements," in *Proc. IEEE Int. Conf. Robotics and Automation*, Apr. 1986, pp. 1433–1438.
[10] A. Gelb, Ed., *Applied Optimal Estimation*. Cambridge, MA: MIT Press, 1974.
[11] D. B. Gennery, "Modelling the environment of an exploring vehicle by means of stereo vision," Ph.D. dissertation, Stanford Univ., Stanford, CA, June 1980.
[12] ———, "Tracking known three-dimensional objects," in *Proc. AAAI*, 1982, pp. 13–17.
[13] P. E. Gill, W. Murray, and M. H. Wright, *Practical Optimization*. New York: Academic, 1981.
[14] W. E. L. Grimson and T. Lozano-Perez, "Model-based recognition and localization from sparse range or tactile data," *Int. J. Robotics Res.*, vol. 3, pp. 3–35, Fall 1984.
[15] J. Hallam, "Resolving observer motion by object tracking," in *Proc. Int. Joint Conf. Artificial Intelligence*, 1983.
[16] M. Hebert, "Reconnaissance de formes tridimensionelles," Ph.D. dissertation, L'Universite de Paris-Sud, Centre d'Orsay, Sept. 1983.
[17] L. H. Matthies and C. E. Thorpe, "Experience with visual robot navigation," in *Proc. IEEE Oceans'84 Conf.*, Washington, DC, Aug. 1984.
[18] H. P. Moravec, "Obstacle avoidance and navigation in the real world by a seeing robot rover," Ph.D. dissertation, Stanford Univ., Stanford, CA, Sept. 1980.
[19] P. H. Schonemann and R. M. Carroll, "Fitting one matrix to another under choice of a central dilation and a rigid motion," *Psychometrika*, vol. 35, pp. 245–255, June 1970.
[20] C. C. Slama, Ed., *Manual of Photogrammetry*. Falls Church, VA: American Society of Photogrammetry, 1980.
[21] R. C. Smith and P. Cheeseman, "On the representation and estimation of spatial uncertainty," Tech. Rep. (draft), SRI International, 1985.
[22] F. Solina, "Errors in stereo due to quantization," Tech. Rep. MS-CIS-85-34, Univ. Pennsylvania, Sept. 1985.
[23] C. E. Thorpe, "Vision and navigation for a robot rover," Ph.D. dissertation, Carnegie-Mellon Univ., Dec. 1984.
[24] R. Y. Tsai and T. S. Huang, "Uniqueness and estimation of three-dimensional motion parameters of rigid objects with curved surfaces," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-6, pp. 13–26, Jan. 1984.
[25] A. M. Waxman and J. J. Duncan, "Binocular image flows," in *Proc. Workshop on Motion: Representation and Analysis*, May 1986, pp. 31–38.
[26] J. R. Wertz, Ed., *Spacecraft Attitude Determination and Control*. D. Reidel Publishing, 1978.

Larry Matthies received the M.Math. degree in computer science from the University of Waterloo, Waterloo, ON, Canada, in 1981 and the B.Sc. degree from the University of Regina in 1979. He is a graduate student in the Computer Science Department of Carnegie-Mellon University (CMU), Pittsburgh, PA, where he is doing doctoral work in computer vision.

His primary research interests are in shape and motion estimation from image sequences. In addition to his work on error modeling in stereo navigation, he was instrumental in the construction of two systems for visual obstacle detection in the Mobile Robot Lab at CMU. His current work addresses the problem of shape determination from precise camera motion. He has also been active in computer graphics. He taught courses and published papers in the area while at the University of Waterloo and developed software packages for 3D rendering at the National Research Council of Canada and Tektronix Research Labs.

Steven A. Shafer (M'84) received the B.A. degree in computer science from the University of Florida, Gainesville, in 1977 and the Ph.D. degree in computer science from Carnegie-Mellon University (CMU), Pittsburgh, PA, in 1983.

He has been with the Computer Science Department at CMU since 1983. He is primarily interested in the analysis of color, gloss, and shadows for determining information from computer images. This work is being carried out in the Calibrated Imaging Laboratory at CMU and is oriented toward robotics tasks in which objects are three-dimensional and are typically smooth and solid. He is especially interested in images taken under extended, non-uniform light sources such as typical commercial light fixtures. He is also working on the architecture and blackboard software for the NAVLAB autonomous vehicle project at Carnegie-Mellon University. He is conducting the effort at CMU to integrate into the NAVLAB system modules that are being developed at other universities and research labs. His past work includes segmentation of digital images using recursive region-splitting. He has produced two programs which achieve this, one each at Carnegie-Mellon and at the University of Hamburg in Germany; the former program (Phoenix) is a component of the Darpa Image Understanding Testbed compiled by SRI International. He is the author of one book and numerous papers, appearing in both the computer science and optics literature, and served as a consultant for the *Handbook of Artificial Intelligence*, the *Encyclopedia of Artificial Intelligence*, and the Time-Life book series *Understanding Computers*. He has taught several cources in computer science, including computer vision, and organized and instructed in the CMU Robotics Institute's Tutorial on Computer Vision. In addition to his work in computer vision, he has been active in the enhancement of the UNIX operating system at Carnegie-Mellon. He is the author and maintainer of over 100 system programs and library subroutines, including a relational database manager (DAB) and a software upgrade program (SUP). He is the author of several software reference manuals and instructional documents, and has taught the use of Unix at CMU and elsewhere. His tutorial documents and notes have been distributed to several universities.

Dr. Shafer is a member of professional societies for computer science, optics, and color science.