CVGIP: IMAGE UNDERSTANDING
Vol. 60, No. 3, November, pp. 313–342, 1994

Robust Methods for Estimating Pose and a Sensitivity Analysis

RAKESH KUMAR

David Sarnoff Research Center, Princeton, New Jersey 08543

AND

ALLEN R. HANSON*

Computer Science Department, University of Massachusetts at Amherst, Amherst, Massachusetts 01003

Received January 21, 1993; accepted March 9, 1994

This paper mathematically analyzes and proposes new solutions for the problem of estimating the camera 3D location and orientation (*pose determination*) from a matched set of 3D model and 2D image landmark features. Least-squares techniques for line tokens, which minimize both rotation and translation simultaneously, are developed and shown to be far superior to the earlier techniques which solved for rotation first and then translation. However, least-squares techniques fail catastrophically when outliers (or gross errors) are present in the match data. Outliers arise frequently due to incorrect correspondences or gross errors in the 3D model. Robust techniques for pose determination are developed to handle data contaminated by fewer than 50.0% outliers. Finally, the sensitivity of pose determination to incorrect estimates of camera parameters is analyzed. It is shown that for small field of view systems, offsets in the image center do not significantly affect the location of the camera in a world coordinate system. Errors in the focal length significantly affect only the component of translation along the optical axis in the pose computation. © 1994 Academic Press, Inc.

1. INTRODUCTION

This paper mathematically analyzes and proposes new solutions for the problem of estimating camera location and orientation (*pose determination*) from a set of recognized landmarks appearing in the image. Given correspondences between 3D lines (or points) and 2D lines (or points) found in the image, the goal is to find the rotation and translation matrices which map the world coordinate system to the camera coordinate system. Algorithms are developed for handling both 3D line and 3D point landmarks. In this paper, only the line algorithms are presented; for the point algorithms, see [15]. The paper presents a sequence of least-squares techniques to solve the pose determination problem, each performing better than the previous one. The least-squares techniques minimize nonlinear functions, are iterative in nature, and require an initial estimate. However, experiments show that there is rapid convergence even with significant errors in the initial estimates. For those cases in which an initial estimate of rotation and translation is not available, techniques based on sampling the rotation space to provide initial estimates are developed. A mathematical analysis of an uncertainty measure is developed, which relates the variance in the output parameters to the noise present in the input parameters. For the experiments and analysis reported in this paper, it is assumed that there is no noise in the 3D model data and the only input noise is in the image data. In Kumar's thesis [15], pose determination techniques for handling errors in the 3D model are presented.

Least-squares techniques are known to be sensitive to gross errors or *outliers* in the data; techniques based on robust statistics to handle outliers are therefore developed. We investigate two different statistical techniques for the "robust" estimation of pose with data having outliers. The first set of techniques, called M-estimation techniques, minimize nonconvex functions of the individual residual errors. The effect of large errors is bounded by saturating the minimization function. Experimentally, the M-estimate methods seem to be susceptible to initial estimates and are not able to handle a large number of outliers (a maximum of approximately 20%). However, there are efficient computational methods for minimizing the associated error functions.

The second set of algorithms developed are capable of performing correctly in situations where the number of outliers is less than 50% of the number of data points. Also, they are not as sensitive as the M-estimate techniques to initial estimates. These estimate the LMS (least

* This research was supported by the following grants: Advanced Research Projects Agency (via TACOM) Grant DAAE07-91-C-RO35 and National Science Foundation Grant CDA-8922572.

1049-9660/94 $6.00

median of squares) of the residual error across all landmarks. The LMS pose computed is used to detect and remove outliers, and then a least-squares fit on the remaining data is used to obtain the rotation and translation matrix estimates. These algorithms are computationally much slower than those based on M-estimation techniques. Using random subset selection methods, the average time complexity of the LMS-based algorithms is substantially reduced with a minimal loss in robustness.

The camera model assumed is perspective projection. Intrinsic camera parameters, such as focal length, field of view, center of the image, and size of image, are assumed to have been estimated by a camera calibration procedure [5, 16, 26]. The sensitivity of the pose parameters to errors in the intrinsic camera parameters such as focal length and image center is analyzed. In particular, for "small" field of view imaging systems, incorrect knowledge of the camera center (or "principal point") fortunately does not significantly affect the determination of the location of the robot camera in the world coordinate system. However, the orientation of the robot is affected. The amount of error in the orientation is linearly related to the error in the estimate of the center. It is also shown that incorrect estimates of the focal length only significantly affects the z-component (i.e., the component parallel to the optical axis) of the translation in camera coordinates.

The remainder of the paper is organized as follows: Section 2 discusses the geometry of perspective projection and the rotation and translation constraints. Section 3 presents the least-squares nonlinear technique and solution methods for situations when there is no good initial estimate. Section 4 presents robust pose techniques which can deal with outliers. Extensive experimental results using the developed pose algorithms are reported at the ends of Sections 3 and 4. The sensitivity analysis of pose estimation is presented in Section 5.

2. ROTATION AND TRANSLATION CONSTRAINTS

In this section, the basic constraints for pose determination are developed. Given correspondences between 3D lines (or points) and 2D lines (or points) found in the image, the goal in pose determination is to find the rotation and translation matrices which map the world coordinate system to the camera coordinate system. The constraints developed in this paper relate the rotation and translation parameters to the 3D model line (or point) coordinates and the corresponding 2D image line (or point) coordinates. These constraints are used in Section 3 to develop the objective functions, which are minimized to find the optimum pose parameters given noisy input data.

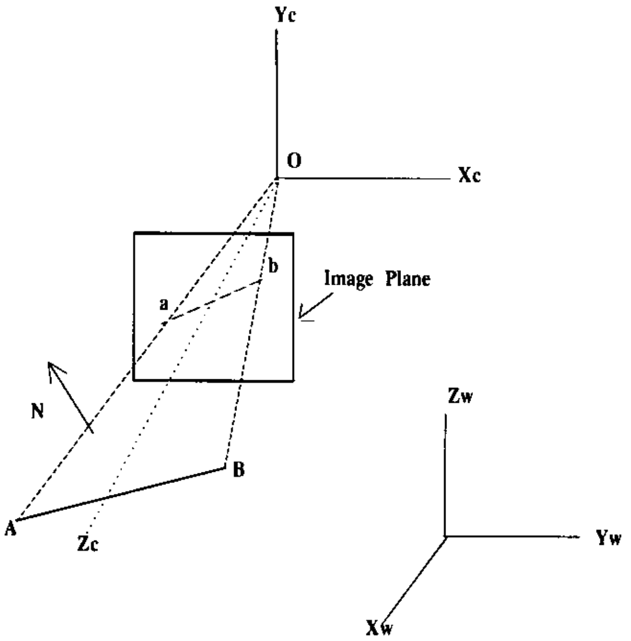

FIG. 1. Perspective projection of image lines.

2.1. Pose Constraint for Points

Points in 3D space are represented by 3D vectors $\mathbf{p}$. Lines in 3D are represented by two endpoints $\mathbf{p}_1$ and $\mathbf{p}_2$. The unit vector corresponding to the direction $\hat{d}$ of the 3D line is determined by subtracting the two endpoint vectors. The rigid body transformation from the world coordinate system to the camera coordinate system can be represented as a rotation (R) followed by a translation ($\mathbf{T}$). The point $\mathbf{p}$ in world coordinates gets mapped to the point $\mathbf{p}_c$ in camera coordinates. The mapping is given by

$$\mathbf{p}_c = R(\mathbf{p}) + \mathbf{T}. \tag{1}$$

Figure 1 shows the camera and world coordinate systems with X_w, Y_w, and Z_w representing the axes of the world coordinate system. O is the optical center of the lens and also the origin of the camera coordinate system $OX_cY_cZ_c$. OZ_c is the optical axis of the camera. In Eq. (1) the translation vector $\mathbf{T}$ represents the location of the origin of the world coordinate system in camera coordinates. Equation (1) can be rewritten to map points in the camera coordinate system to the world coordinate system:

$$\mathbf{p} = R^{\mathrm{T}}(\mathbf{p}_c) - R^{\mathrm{T}}(\mathbf{T}) = R^{\mathrm{T}}(\mathbf{p}_c) + \mathbf{T}_w. \tag{2}$$

In this equation, R^{T} is the transpose of the rotation operator[1] in Eq. (1). $\mathbf{T}_w$ represents the location of the origin of the camera coordinate system in world coordinates.

[1] Since, the rotation operator is expressed as an orthonormal matrix, its transpose is the same as its inverse.

The 3D line AB in Fig. 1 projects to the image line ab. Image points **I** and **J** are represented by 2D vectors. A 3D point $\mathbf{p}_c$ projects to an image pixel **I** by the equations

$$I_x = s_x \frac{p_{cx}}{p_{cz}}, \quad I_y = s_y \frac{p_{cy}}{p_{cz}}, \tag{3}$$

where s_x and s_y are the scale factors along the X and Y directions, respectively.

Using Eq. (1) and (3) the constraint equations for the pose parameters given point correspondences are

$$I_x = s_x \frac{(R\mathbf{p} + \mathbf{T})_x}{(R\mathbf{p} + \mathbf{T})_z}, \quad I_y = s_y \frac{(R\mathbf{p} + \mathbf{T})_y}{(R\mathbf{p} + \mathbf{T})_z}. \tag{4}$$

2.2. Pose Constraint for Lines

In the case of line correspondences, it is assumed that model and image line endpoint correspondences cannot be established. Constraint equations for the pose parameters can be developed in two ways. In the first case (''infinite image line'') a model line segment is aligned with an infinitely extended image line, whereas in the second case (''infinite model line'') an image line segment is aligned with an infinitely extended model line.

2.2.1. *The Infinite Image Line Case*

In the infinite image line case an image line is represented by its (ρ, θ) parameters. Any image point (I_x, I_y) on the ith line must satisfy the equation

$$I_x \cos\theta + I_y \sin\theta = \rho. \tag{5}$$

Substituting I_x and I_y from Eq. (3) into Eq. (5), the equation of the projection plane formed by the image line and the optical center is obtained:

$$\frac{(s_x \cos\theta) p_{cx} + (s_y \sin\theta) p_{cy} - \rho p_{cz}}{p_{cz}} = 0. \tag{6}$$

In Fig. 1, the projection plane formed by the image line ab is given by the plane Oab and the 3D line AB must lie in this plane. The normal **N** to the projection plane is given by

$$\mathbf{N} = (s_x \cos\theta,\ s_y \sin\theta,\ -\rho)^{\mathrm{T}}. \tag{7}$$

Using Eqs. (1) and (7), Eq. (6) can be rewritten as

$$\frac{\mathbf{N} \cdot (R(\mathbf{p}) + \mathbf{T})}{(R(\mathbf{p}) + \mathbf{T})_z} = 0 \tag{8}$$

This equation is the basic constraint for the infinite image line case. The projected model endpoint must lie on the infinitely extended image line. The perpendicular distance of a 3D point **p** on the projection plane is given by $(\hat{\mathbf{N}} \cdot (R(\mathbf{p}) + \mathbf{T}))$. Note, $\hat{N}$ is the unit vector of **N**. Setting the expression for perpendicular distance equal to zero, a simpler 3D constraint is formulated:

$$\hat{N} \cdot (R(\mathbf{p}) + \mathbf{T}) = 0. \tag{9}$$

The 3D constraint represents the fact that any point on the 3D line in camera coordinates ideally must lie in the projection plane. The vector formed from the origin (optical center) to this point must be perpendicular to the normal of the projection plane.

These constraint equations (8) and (9) relate both rotation and translation pose parameters to the 3D model and 2D image coordinates. A separate constraint involving just rotation is obtained, if we subtract Eq. (9) for two points $\mathbf{p}_1$ and $\mathbf{p}_2$ lying on a line [18]:

$$\begin{aligned} \hat{N} \cdot R(\mathbf{p}_1 - \mathbf{p}_2) = \hat{N} \cdot R(\mathbf{d}) &= 0 \\ \hat{N} \cdot R(\hat{d}) &= 0. \end{aligned} \tag{10}$$

The above equation is obtained by subtracting the left-hand side of Eq. (9) for two 3D points lying on a line. Note the direction vector $\mathbf{d} = (\mathbf{p}_1 - \mathbf{p}_2)$. The constraint reflects the fact that the 3D line must lie in the projection plane formed by its corresponding image line. Rigid body transformation can be represented as a rotation followed by a translation. Since translation does not change the direction of the line, the direction of the 3D line after rotation in Eq. (10) must be perpendicular to the normal of the projection plane of the image line.

From the constraints represented in Eq. (9) and (10), two algorithms have been developed to solve for rotation and translation. In the first algorithm (''R_then_T'') the constraint in Eq. (10) is used to solve for rotation. Then the rotation result returned from this step is used in conjunction with Eq. (9) to solve for translation.

In the second algorithm (''R_and_T''), only Eq. (9) is used and both rotation and translation are solved for simultaneously. For each line, the two endpoints must satisfy Eq. (9). The tables in Section 3.7 will show that R_and_T performs much better than R_then_T. Finally, a third algorithm (''R_and_T_img'') is developed which uses the image-based constraint equation (8) instead of the 3D-based constraint equation (9) to solve for rotation and translation simultaneously.

2.2.2. *The Infinite Model Line Case*

In the infinite model line case, an image line segment is aligned with the infinite extension of a projected model line. The (ρ, θ) parameters in the line equation (5) are now used to represent the infinite extension of the projected model line. The image line endpoints ($\mathbf{I}_1$ and $\mathbf{I}_2$) must lie on this projected model line and satisfy eq. (5). We now develop the algebra for relating the projected model line to the 3D model endpoints. Since we wish to develop the constraint and corresponding error function to be image plane errors, the following algebra is a little tedious.

Expressions can be derived relating $\cos(\theta)$, $\sin(\theta)$, and ρ in Eq. (5) to the model line endpoints (p_1, p_2), and the pose (R, $\mathbf{T}$). J_1 and J_2 are the image projections of the 3D model line endpoints (p_1 and p_2). The expressions for ($\cos(\theta)$, $\sin(\theta)$, and ρ) for the image line between ($\mathbf{J}_1$ and J_2) are

$$\cos\theta = \frac{(J_{2y} - J_{1y})}{\sqrt{((J_{1x} - J_{2x})^2 + (J_{2y} - J_{1y})^2)}} \tag{11}$$

$$\sin\theta = \frac{(J_{1x} - J_{2x})}{\sqrt{((J_{1x} - J_{2x})^2 + (J_{2y} - J_{1y})^2)}} \tag{12}$$

$$\rho = \frac{(J_{1x}J_{2y} - J_{2x}J_{1y})}{\sqrt{((J_{1x} - J_{2x})^2 + (J_{2y} - J_{1y})^2)}}. \tag{13}$$

The projections of the 3D model line endpoints into the image plane points ($\mathbf{J}_1$ and $\mathbf{J}_2$) are given by Eq. (4). Substituting these expressions for points ($\mathbf{J}_1$, $\mathbf{J}_2$) into the equations for ($\cos(\theta)$, $\sin(\theta)$, and ρ), the following is obtained,

$$\cos\theta = \frac{1}{s_x M}(R(\mathbf{p}_2 - \mathbf{T}_w) \times R(\mathbf{p}_1 - \mathbf{T}_w))_x \tag{14}$$

$$\sin\theta = \frac{1}{s_y M}(R(\mathbf{p}_2 - \mathbf{T}_w) \times R(\mathbf{p}_1 - \mathbf{T}_w))_y \tag{15}$$

$$\rho = -\frac{1}{M}(R(\mathbf{p}_2 - \mathbf{T}_w) \times R(\mathbf{p}_1 - \mathbf{T}_w))_z, \tag{16}$$

where the vector $\mathbf{A}$ and scalar $\mathbf{M}$ are defined as

$$\mathbf{A} = R(\mathbf{p}_2 - \mathbf{T}_w) \times R(\mathbf{p}_1 - \mathbf{T}_w) \tag{17}$$

$$M = \sqrt{\left(\left(\frac{A_x}{s_x}\right)^2 + \left(\frac{A_y}{s_y}\right)^2\right)}. \tag{18}$$

The line equation (5) can be rewritten as the dot-product of two vectors:

$$(I_x, I_y, 1) \cdot (\cos\theta, \sin\theta, -\rho) = 0.0. \tag{19}$$

We now define vector $\mathbf{B}$ corresponding to the projection ray from the focal point to an image point $\mathbf{I}$ as

$$\mathbf{B} = \left(\frac{I_x}{s_x}, \frac{I_y}{s_y}, 1\right)^T. \tag{20}$$

Using the above definition and substituting Eqs. (14)–(16) into the line equation (19) the infinite model line constraint equation becomes

$$\frac{1}{M}(R^T\mathbf{B} \cdot (\mathbf{p}_2 - \mathbf{T}_w) \times (\mathbf{p}_1 - \mathbf{T}_w) = 0. \tag{21}$$

For each image line endpoint, the constraint equation (21) is developed. It represents the constraint that the image line endpoints $\mathbf{I}_1$ and $\mathbf{I}_2$ must lie on the projection of the 3D model line. From this constraint, another least-squares algorithm (R_and_T_mod) is developed in Section 3 for computing the pose parameters given line correspondences. In Section 3.7, it is shown that algorithm (R_and_T_mod) performs more robustly than algorithm (R_and_T_img) when there is significant line fragmentation in the image data.

2.3. Minimum Number of Lines/Points

Both rotation and translation in the 3D world can be represented by three parameters each. Each line or point data provides two constraint equations (4). Thus, a minimum of three lines or points are needed. However, in many cases, with three lines or points, there is no unique solution. If the three lines are parallel in 3D space or lie on the same projection plane, then an infinite number of solutions can be found. If the three lines meet at a common point in 3D space, then there are two solutions for rotation (the Necker cube phenomena) and an infinite number of solutions for translation. In general, there are eight possible solutions for the three-point or line case, four of which correspond to the 3D points lying in front of the camera and four corresponding to the 3D points lying behind the camera [4, 7, 9].

3. LEAST-SQUARES TECHNIQUES

3.1. Previous Work and Our Approach

The problem of pose determination has been referred to by various other names including "exterior orientation," "determination of camera location and orientation," "pose refinement," "perspective inversion," and "model alignment." There have been many papers on pose determination, but most assume only point data are available; only a few have presented techniques for line data (see [15] for an extensive review). No closed form solution has yet been found for the general pose problem with an arbitrary set of lines/points in an arbitrary configuration. Most techniques minimize nonlinear error functions, are iterative in nature, and require an initial estimate.

Iterative techniques for point data based on linearization of a nonlinear error function are presented in the photogrammetry literature [28]. Wu *et al.* [30] use extended Kalman filtering to solve for the pose and motion of an object given point correspondences from a matching algorithm. They track the pose of the object over a sequence of frames and define their state vector to consist of both the position and motion of the object. Their dynamic model assumes that the velocity of the object remains constant. Mitchie *et al.* [23] use constraints based on conservation of distance under rigid body transformation to solve for the length of the "legs" of an arbitrary set of points. Once the points have been located in the 3D camera coordinate system, the pose can be computed by employing a simpler absolute orientation (3D–3D pose) algorithm. This technique is employed by them to solve the related problem of relative orientation between two camera coordinate systems (structure from motion) given correspondences between points in the two image frames. However, their method involves minimizing a nonlinear error function with as many variables as the number of points. In contrast, the techniques presented here solve directly for the six pose parameters and the dimension of the error function does not increase with the number of lines/points.

Some researchers [2, 27] have examined the use of image lines as an alternative to point data; the idea is that lines provide a more stable image feature to match. It is assumed that line endpoints cannot be reliably extracted and hence endpoint correspondences cannot be established. This has led here to the formulation of two sorts of constraint equations for pose determination: the infinite image line case and the infinite model line case. In the infinite image line case, a model line segment is aligned with an infinitely extended image line, whereas in the infinite model line case extracted image line segments are aligned with infinitely extended model lines. Beveridge *et al.* [2] report that, in the case of a 2D–2D pose problem, if extracted image lines have significant line breaks, then the infinite model line algorithms perform much better than the infinite image line algorithms. We develop a series of algorithms R_then_T, R_and_T, and R_and_T_img based on the infinite image line constraint, each performing better than the previous algorithm. Finally, we show that our algorithm R_and_T_mod based on the infinite model line constraint performs best of all.

Liu *et al.* present a solution to the "camera location determination" problem which works for both point and line data [18]. Their constraint is based on the infinite image line case and on the observation that three-dimensional lines in the camera coordinate system must lie on the projection plane formed by the corresponding image line and the optical center. Using this fact, constraints for rotation can be separated from those of translation. Equations (9) and (10) express these constraints. They first solve for the rotation and then use the rotation result to solve for the translation. Two methods are suggested to solve for the rotation constraint. In the first method, rotation is represented as an orthonormal matrix and a smallest eigenvalue-based solution technique is presented. However, the six orthonormality constraints for an orthonormal matrix are not enforced. It is not clear how they would find the nearest orthonormal matrix to the matrix their algorithm returns, and whether that would be a solution to the original problem. The second method represents rotation by Euler angles and is a nonlinear iterative solution obtained by linearizing the problem about the current estimate of the output parameters. The translation constraint is solved for by a linear least-squares method. Liu *et al.* extend their technique to point data by drawing virtual lines between pairs of points [18]. They use the same rotational constraint; however, the translation constraint is different from that for lines.

The algorithm R_then_T developed here is similar to the line-based algorithm developed by Liu *et al.* [18]. However, a different nonlinear technique based on the Gauss–Newton method is used for the minimization process. We believe that the application of this technique gives us much better convergence properties that Liu's solution using Euler angles.

One of the results of this paper is that decomposition of the solution into the two stages, of first solving for rotation and then for translation, does not use the set of constraints effectively. This same observation was made by other researchers working on the structure from motion problem [22]. The rotation and translation constraints (refer to Eqs. (9), (10)), when used separately, are very weak constraints. When solving for them separately, small errors in the rotation stage are amplified into large errors in the translation stage. This is particularly true with the large distances of the landmarks from the camera in our application. Much better noise immunity is obtained if both rotation and translation are solved for simultaneously. That is the approach adopted here. Another algorithm (R_and_T) was developed to solve for the rotation and translation simultaneously. Section 3.7 presents results that show algorithm R_and_T performs much better than R_then_T.

Lowe [19] presents iterative techniques for both point and line data. He reparameterizes the rigid body transformation parameters to compute rotation and a translation vector D (p. 120 of his book). The original translation vector T (as defined in Eq. (1)) can only be meaningfully recovered from this vector D when all the landmarks have the same depth. Therefore the solution presented in his book is not applicable to the problem of camera location determination as formulated here, where the typical scenes that we consider have landmarks spread over a large range in depth. For line data, the solution presented in his book is based on the infinite image line case (similar to our R_and_T_img algorithm). However, in his recently published papers [20, 21], the rigid body transformation parameters adopted are similar to ours and the error functions minimized are based on the infinite model line case (similar to the R_and_T_mod algorithm). His papers, however, do not discuss the relative merits of the infinite model line-based error functions versus the infinite image line-based error functions. He also does not extend his techniques to be robust with respect to outliers. Instead he integrates the pose estimation step with the matching step and uses a best-first search technique to find the best match and pose [21].

Worrall *et al.* [29] present a least-squares technique for line data which minimizes an error measure derived from the infinite image line constraint equation (9). They compare results with one of Lowe's solutions and report similar performance. However, they represent rotations as Euler angles and use a different nonlinear technique from that presented here. They also do not handle outliers or provide a mathematical analysis of the errors.

3.2. Camera Calibration

It is important to draw the distinction here between techniques for "camera calibration" [5, 16, 26], also called "interior orientation," and the techniques for camera location determination. Camera calibration techniques solve for intrinsic camera parameters (such as focal length, image center, lens distortion) along with the pose parameters (rotation and translation). The techniques for camera calibration require very precise image measurements and are less tolerant to noise. Pose determination techniques are less susceptible to image noise but one needs to know the intrinsic camera parameters. In Section 5, the effect of errors in the intrinsic camera parameters on the pose determination problem is studied.

Ganapathy [8] presents a linear closed form solution for point data. In addition to solving for the rotation and translation parameters, he also solves for the center of the image and scaling along the x and y directions in the image. Our implementation of his technique shows that it is extremely susceptible to noise, probably due to the use of a linear least-squares minimization where it is assumed that all the parameters are independent when they are not. Faugeras and Toscani [5] developed a technique to solve a similar system of equations with appropriate constraints and report better results. Lenz and Tsai [16, 26] have also developed state of the art camera calibration techniques. They assume that the lens distortion is mostly radial and solve for the lens distortion parameters and the pose parameters using a radial alignment constraint equation. The image center and the relative scale along the image x and y axes are solved for separately using a variety of special methods [16]. For their technique to work image points must be located to within 0.1 pixel accuracy.

3.3. Objective Functions for Line Data

Ideally, if there was no noise, the minimal set of constraint equations (4) or (8) and (21) for three sets of point or line correspondences, respectively, could be used to solve for the unknown pose parameters. The correct pose parameters would then cause perfect alignment between the projected model landmarks and the image measurements. However, in practice, measurements are noisy and perfect alignment cannot be realized. Therefore an objective function is minimized to find the pose parameters. The objective function is typically some function of the alignment error between the transformed model landmarks and image measurements. All of the objective functions constructed here are nonlinear in nature. Therefore a crucial factor in the choice of the objective function is the ability to construct suitable minimization algorithms for it. Suitability of minimization techniques is gauged by the following parameters: (1) speed of minimization, (2) convergence from a suitably large distribution of initial estimates, (3) numerical stability of the algorithm with respect to noise, finite length calculations, etc.

In this section, a set of different objective functions for line and point data is presented. The objective functions are a sum of squares of a function of the alignment error between each 3D model feature and its corresponding image measurement. To optimally estimate the pose (rotation and translation) parameters, the error for each landmark feature must be appropriately weighted based on the assumptions of noise in the measurement process. In the least-squares algorithms presented in this section, it is assumed that the image measurements are corrupted with zero-mean independent gaussian noise. It is also assumed in this section that the input noise in the 3D model is not significant. Thus the error terms corresponding to each image line or point measurement are weighted by the inverse of the squared standard deviation of the measurement noise.

The first set of objective functions developed is based on the line constraint equations (9) and (10):

$$E_R = \sum_{i=1}^{n} w_i(\hat{N}_i \cdot R(\hat{d}_i))^2 \tag{22}$$

$$E_1 = \sum_{i=1}^{2n} w_i(\hat{N}_i \cdot (R(\mathbf{p}_i) + \mathbf{T}))^2. \tag{23}$$

If there was no noise, the normal of the projection planes formed using the image lines would be perpendicular to the direction of the rotated model lines and thus the cosine of the angle between them would be zero. E_R is the sum of squares of these cosines. E_1 is the weighted sum-of-the-squares of the perpendicular distances from the endpoints of the 3D lines to their corresponding projection planes. For each line two 3D endpoints are used and

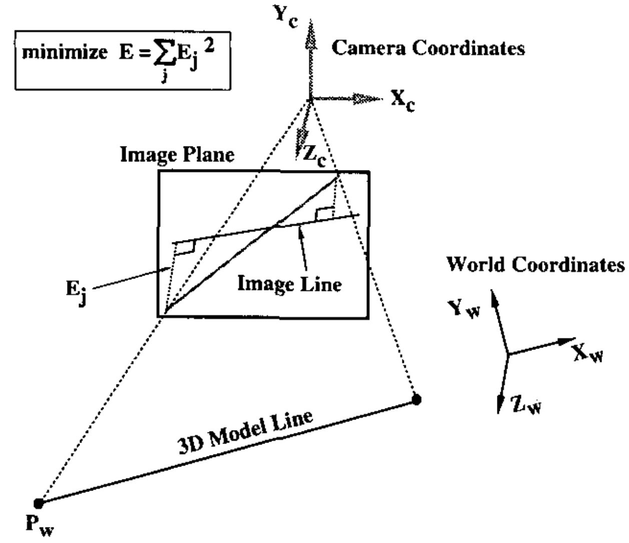

FIG. 2. Error function based on the infinite image line constraint.

therefore each line contributes twice to the objective function E_1. In these equations, w_i is the weight applied to the error for each line for optimal estimation.

In the R_then_T algorithm for line data, the rotation objective function E_R is minimized to solve for rotation R. The estimated rotation is used to minimize the objective function E_1 to determine the translation **T**. In contrast, the R_and_T algorithm minimizes only E_1 to determine rotation R and translation **T** simultaneously. Objective function E_1 is a 3D alignment error between model line endpoints and the infinitely extended image line. However, the only significant noise is assumed to be in the image measurements. Thus to optimally weight the error terms in E_1, the weight terms w_i must be computed by reverse projecting the standard deviation value of the image noise onto the 3D projective plane. A similar "optimal" objective function can be obtained more simply by minimizing the sum of squares of the residual error function defined in the image plane:

$$E_2 = \sum_{i=1}^{2n} w_i \left(\frac{\mathbf{N}_i \cdot (R(\mathbf{p}_i) + \mathbf{T})}{(R(\mathbf{p}_i) + T)_z} \right)^2. \qquad (24)$$

This error function is based on the constraint equation (8). The resulting algorithm which minimizes E_2 is called R_and_T_img. The objective function E_2 is the sum of squares of the perpendicular distance between projected model endpoints of the infinitely extended image line. Note in E_2 the normal vector **N** is not a unit vector. It is defined by Eq. (7). Figure 2 shows this alignment error for one line pair. The weights in Eq. (24) are the inverse values of the standard deviations of the image noise in the direction perpendicular to the line. E_2 is also a rational function and therefore it is more difficult to optimize. To minimize E_2, at each iteration in our nonlinear technique, the denominator term $(R(\mathbf{p}_i) + T)_z^2$ for each point is held constant. In the next iteration, the denominator is updated with the new R and **T**. Therefore, we are able to employ the same algorithm as used for E_1. This seems to work for all the cases the algorithm has been run on. The same technique is applied for other error functions which have a denominator term.

In contrast to E_1 and E_2, which are based on the infinite image line constraints given in eqs. (9) and (8), respectively, an objective function (E_3) is constructed based on the infinite model line constraint equation (21),

$$E_3 = \sum_{i=1}^{n} \sum_{j=1}^{2} \frac{w_i}{M_i^2} (R^{\mathrm{T}}\mathbf{B}_{ji} \cdot (\mathbf{p}_{2i} - \mathbf{T}_w) \times (\mathbf{p}_{1i} - \mathbf{T}_w))^2 \quad (25)$$

where $\mathbf{T}_w$ *and* R^{T} *are the translation and rotation in the* world coordinate system (2). M is defined in Eq. (18) and **B** is defined in Eq. (20). E_3 is the sum-of-the-squares of the perpendicular distances of the endpoints of the image lines to the projected model line. Figure 3 shows the alignment error for one line pair. It is minimized by a method similar to the techniques used for E_1 and E_2; the algorithm which minimizes E_3 is called R_and_T_mod.

In Section 3.7, it is shown that algorithm R_and_T_mod performs more robustly than algorithm R_and _ T_img when there is significant line fragmentation in the image data. This is because of the difficulty in optimally weighting the error terms in the objective function E_2 minimized by algorithm R_and_T_img. This problem can

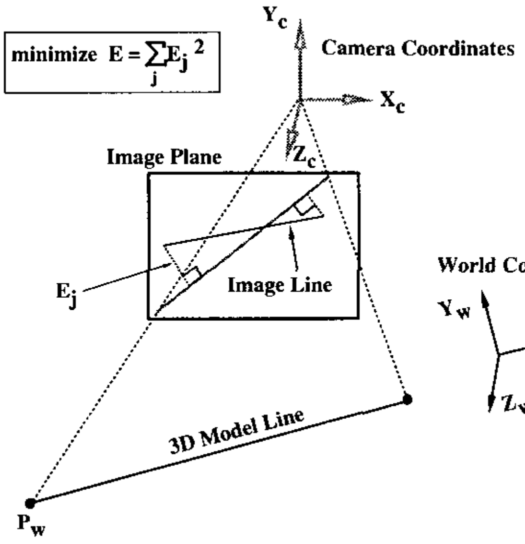

FIG. 3. Error function based on the infinite model line constraint.

be more easily understood by first considering the effect of noise on the measurement of image lines.

The noise in the measurement of image lines can be decomposed into two components: noise in measuring the location of the image line endpoints perpendicular to the line and noise in measuring the location of the image line endpoints along the length of the line. We model the noise in locating the image line endpoints perpendicular to the length of the line as a gaussian random variable. However, it seems intuitively not plausible that fragmentation of image lines is reasonably modeled as a gaussian process. In fact, the noise in locating line endpoints along the length of the line seems significantly dependent upon factors such as the particular line extraction algorithm used, the image contrast across the line length, and intensity structures neighboring the line in the image. Thus it is difficult to develop a general model for the noise in locating line endpoints along the length of the line.

In algorithm R_and_T_img the error in aligning model line segments with infinitely extended image lines is optimized (objective function E_2). In order to derive the optimal weights for the error terms, the alignment error for each line must be approximated by a function of the measurement noise. If the model line segments and the corresponding measured image line segments are of similar length, then the perpendicular noise component in locating image line endpoints reasonably approximates the alignment error. However, if the model line segments are much larger or smaller than the measured image line segments the alignment error can only be "well" approximated by a function of both components of the noise in locating image line endpoints. This, as we noted earlier, is very difficult to do. In contrast, in algorithm R_and_T_mod, the objective function E_3 to be optimized is the error in aligning image line segments with infinitely extended model lines. The alignment error considered in this case depends only on the perpendicular noise component in locating the image line endpoints. Thus, to optimally weight the error terms in E_3 it is not necessary to know (or erroneously model) the measurement noise along the length of the image lines. In the experiments reported in Section 3.7, for both algorithms R_and_T_img and R_and_T_mod, the weights used in the respective objective functions are based only on the perpendicular noise components.

E_1, E_2, E_3, and E_R are all minimized by modifying the same basic nonlinear technique. Therefore, only the technique for minimizing E_1 is presented in the next section. For algorithm R_then_T, since rotation has already been estimated, E_1 is minimized by a straightforward linear least-squares algorithm.

3.4. Nonlinear Technique for R_and_T

To minimize E_1, we adapt an iterative technique formulated by Horn [12] to solve the problem of relative orientation. An initial estimate is required for both R and $\mathbf{T}$. The technique linearizes the error terms about the current estimate for R and $\mathbf{T}$. At each iteration, the linearized error function is minimized to determine adjustment vectors for the rotation and translation terms. The iterative adjustments are made to the rotation and translation terms until the objective function E_1 converges to a minimum. Note that the algorithm, like all such descent algorithms, does not guarantee a global minimum.

Assume we have a current estimate R for rotation. The coordinates $\mathbf{p}_i'$ of a rotated 3D point is given by $\mathbf{p}_i' = R(\mathbf{p}_i)$. An incremental rotation vector $\delta\omega$ is added to the rotation estimate $\mathbf{R}$; the direction of this incremental vector is parallel to the axis of rotation, while its magnitude is the angle of rotation.

This incremental rotation takes $\mathbf{p}_i'$ to $\mathbf{p}_i''$:

$$\mathbf{p}_i'' = \mathbf{p}_i' + \delta\omega \times \mathbf{p}_i' \tag{26}$$

This follows from Rodrigue's formula [12] for the rotation of a vector r to r' by angle θ about the three dimensional axis vector ω:

$$r' = r(\cos\theta) + \sin\theta(w \times r) + (1 - \cos\theta)(\omega \cdot r)\omega, \tag{27}$$

where $\theta = \|\delta\omega\|$ and $\omega = \delta\omega/\|\delta\omega\|$

Let $\boldsymbol{\Delta}\mathbf{T}$ represent a small translation added to the current translation estimate $\mathbf{T}$. Thus, the linearized energy function about the current estimate R and $\mathbf{T}$ becomes

$$E = \sum_{i=1}^{2n} w_i(\mathbf{N}_i \cdot (\mathbf{p}_i' + \boldsymbol{\delta\omega} \times \mathbf{p}_i' + T + \boldsymbol{\Delta}\mathbf{T}))^2. \tag{28}$$

Let $\mathbf{b}_i = \mathbf{p}_i' \times \mathbf{N}_i$. Using the chain rule of triple scalar product for vectors, differentiating the objective function with respect to $\boldsymbol{\Delta}\mathbf{T}$ and $\boldsymbol{\delta\omega}$ respectively, and setting the results equal to 0, the following two equations are obtained after some manipulation:

$$\sum_{i=1}^{2n} w_i(\mathbf{N}_i \cdot \boldsymbol{\Delta}\mathbf{T} + \boldsymbol{\delta\omega} \cdot \mathbf{b}_i)\mathbf{N}_i = -\sum_{i=1}^{2n} w_i(\mathbf{N}_i \cdot (\mathbf{p}_i' + \mathbf{T}))\mathbf{N}_i \tag{29}$$

$$\sum_{i=1}^{2n} w_i(\mathbf{N}_i \cdot \boldsymbol{\Delta}\mathbf{T} + \boldsymbol{\delta\omega} \cdot \mathbf{b}_i)\mathbf{b}_i = -\sum_{i=1}^{2n} w_i(\mathbf{N}_i \cdot (\mathbf{p}_i' + \mathbf{T}))\mathbf{b}_i. \tag{30}$$

Together, these two vector equations constitute six linear scalar equations in the six unknown components of $\boldsymbol{\Delta}\mathbf{T}$ and $\boldsymbol{\delta\omega}$. They can be rewritten in the more compact matrix form:

$$A\begin{bmatrix}\boldsymbol{\Delta}\mathbf{T}\\ \boldsymbol{\delta\omega}\end{bmatrix} = \mathbf{f}. \tag{31}$$

In the above equation, A is a 6×6 matrix and $\mathbf{f}$ is a 6×1 vector. A and $\mathbf{f}$ are defined

$$A = \begin{bmatrix} C & F \\ F^T & D \end{bmatrix} \tag{32}$$

$$\mathbf{f} = \begin{bmatrix} \bar{c} \\ \bar{d} \end{bmatrix} \tag{33}$$

where

$$C = \sum_{i=1}^{2n} w_i \mathbf{N}_i \mathbf{N}_i^T, \quad D = \sum_{i=1}^{2n} w_i \mathbf{b}_i \mathbf{b}_i^T, \quad F = \sum_{i=1}^{2n} w_i \mathbf{N}_i \mathbf{b}_i^T,$$

while

$$\bar{c} = \sum_{i=1}^{2n} w_i (\mathbf{N}_i \cdot (\mathbf{p}_i' + \mathbf{T}))\mathbf{N}_i \text{ and } \bar{d} = \sum_{i=1}^{2n} w_i (\mathbf{N}_i \cdot (\mathbf{p}_i' + \mathbf{T}))\mathbf{b}_i.$$

Solving the above set of six linear equations gives a way of finding small changes in rotation and translation that reduce the overall objective function. The algorithm can therefore be expressed in the following four steps.

Step 1. Given an initial estimate for rotation R and translation $\mathbf{T}$ and a list of correspondences between 3D modeled lines and their image measurements.

Step 2. Compute the coefficients of the matrices in Eq. (31). Solve the linear system for $\boldsymbol{\Delta}\mathbf{T}$ and $\boldsymbol{\delta\omega}$.

Step 3. Compose $\boldsymbol{\delta\omega}$ with the current estimate R of rotation to get the new estimate. Add $\boldsymbol{\Delta}\mathbf{T}$ to $\mathbf{T}$ to get the next estimate for translation.

Step 4. Stop if the algorithm has converged or has exceeded a maximum number of iterations; otherwise, go back to Step 2.

The current rotation estimate R is represented as a quaternion. At each iteration, the rotation increment $\boldsymbol{\delta\omega}$ is transformed into a quaternion and composed with the current estimate to form the new rotation estimate [12]. The iteration procedure is terminated when either a maximum number of iterations is exceeded or when the difference in the result between two successive iterations is less than a prespecified minimum. The computational complexity of the algorithm is $O(kn)$ where there are n points or lines and k iterations are needed to converge to the optimal solution.

The covariance matrix Λ_P of the estimated pose parameters is related to the coefficient matrix A (defined in Eq. (32)):

$$\Lambda_P = A^{-1} \tag{34}$$

The above formula for the covariance matrix is only valid if the residual error terms of the objective function are optimally weighted. The covariance matrix is computed using the final pose parameters estimated at the last iteration. The matrix A is known as the information matrix. It becomes singular when there are an infinite number of solutions, e.g., a data set of less than three lines, all lines are parallel, and all lines meet at a point. For these cases, incremental adjustments to the current pose estimate cannot be computed.

3.4.1. *Performance of the Nonlinear Algorithm*

The nonlinear algorithm described above requires the user to specify an initial estimate for translation and rotation. How close the initial estimate must be to the final values, in order to ensure convergence, depends on the particular data set. The algorithm seems to converge for initial estimates which differ considerably from the correct solution. Generally, the rotation estimate is more important than the translation estimate. For some data sets, convergence seems to be almost independent of the starting point; for others the initial rotation estimate must be within 40° for all the three Euler angles representing the rotation.

Another important question asked about nonlinear iterative algorithms is speed of convergence. In our experiments, for "good" data sets of about 10 or more lines, the algorithm typically converges in three or four iterations for initial estimates that may be more than 40° off in rotation and 100 ft off in translation. For instance, Fig. 4 shows an initial set of input image lines used for an experiment to demonstrate the convergence properties of the above minimization technique. Figure 4 is the first frame of a set of outdoor images on which the pose algorithms were tested. Figure 5 is the projection of the model using the initial estimate of the pose parameters. The initial estimate is off by more than 100 ft in translation along the walkway direction, 20 ft under the walkway in the vertical direction, and about 15° off from the axis for rotation. Figure 6 shows the projection of the model using the estimate of the pose parameters obtained after the first

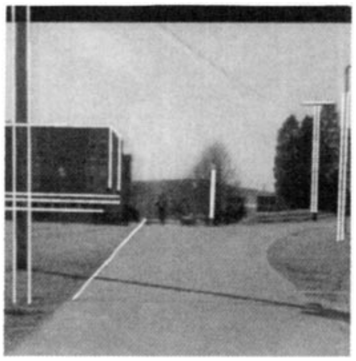

FIG. 4. Image lines for outdoor frame 1 used for convergence experiment.

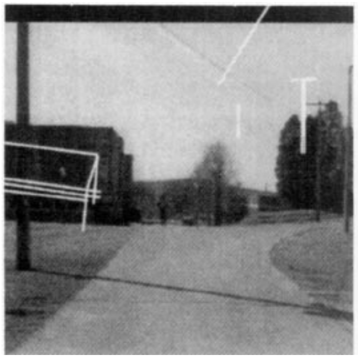

FIG. 5. Projection of model using initial estimate for convergence experiment.

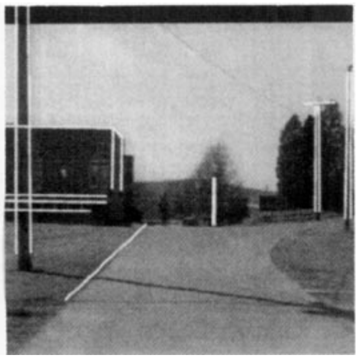

FIG. 7. Projection of model using estimate after second iteration for convergence experiment.

iteration. The pose is now within 10 ft of the correct answer. Figure 7 shows the projection of the model using the estimate of the pose parameters after the second iteration; the pose is now almost correct. The algorithm converges in the third iteration and the projection using the final estimate is shown in Fig. 8. Note that in Fig. 8 additional model lines have been projected to show the accuracy of the final projection.

Finally, for certain near singular data sets, it is possible that the technique can diverge rather than converge. Iterative minimization techniques can be looked upon as moving in a multidimensional space, searching for the bottom of the nearest valley. Ideally, at each iteration or move, the function value would decrease until the valley bottom is reached. The Gauss–Newton minimization method adopted here is a second-order technique. Second-order methods have the property of extremely fast convergence under normal conditions. Unlike first-order (or gradient based) methods, they are not guaranteed to descend in every iteration.

For the near singular cases where the technique might diverge, a simple solution exists to guarantee convergence. This solution is motivated by the Levenberg–Marquardt method for minimizing nonlinear functions and its application to this minimization technique for pose was first noted by Lowe [21]. The Levenberg–Marquardt method combines first- and second-order methods. At any current iteration, the increment is first calculated by second-order methods. If the new estimate causes the objective function to increase, then the increment is reestimated by adding a component of the gradient to the old estimate of the increment. Note that moving along the gradient guarantees descent and hence convergence. But gradient-descent algorithms are slow to converge. The Levenberg–Marquardt method attempts to combine the best of both methods by moving along directions close to the gradient only if moving in the direction computed by the second-order method causes divergence.

In the nonlinear technique described above for the pose problem, the gradient direction at any iteration is given by the (6×1) vector $\mathbf{f}$ defined in Eq. (31). If the A matrix in Eq. (31) is diagonal then the incremental move calculated is in the direction of the gradient. Thus, if the new estimate at the end of an interation increases the

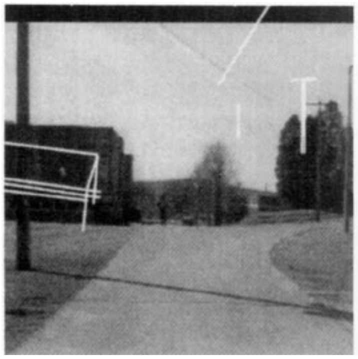

FIG. 6. Projection of model using estimate after first iteration for convergence experiment.

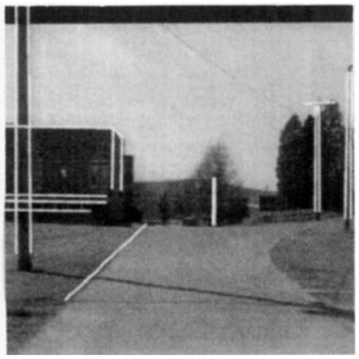

FIG. 8. Projection of model using estimate after third and final iteration for convergence experiment.

total error, then the diagonal terms of A are multiplied by a constant factor (10 in our experiments). This biases the new increment to be toward the gradient direction. This procedure is repeated until the error function decreases after the addition of the increments to the current estimate. Experiments have proven that the method is effective in forcing convergence. However, in most of the data sets we have experimented with, divergent behavior of the second-order method was not observed.

3.5. Initial Estimates of Rotation and Translation

For some applications, initial estimates for rotation and translation may not be available. In that case, the rotation space can be sampled and each of the samples used as an initial estimate for the rotation estimation part of R_then_T. The rotation and translation estimates made by the algorithm R_then_T are then used as initial estimates for R_and_T. We have successfully tried this procedure with 12 uniform samples of the rotation space based on the rotation group of a tetrahedron.

The algorithm for pose determination, when no initial estimate is available, is the following:

Step 1. Pick a rotation estimate from a uniform sampling of the rotation space.

Step 2. Run algorithm R_then_T to get estimates for both R and $\mathbf{T}$.

Step 3. Use the estimates from Step 2 as initial estimates to R_and_T.

Step 4. Repeat Steps 1–3 until all rotation samples have been considered.

Step 5. Return the estimate which gives the smallest alignment error in Step 3 as the final estimate.

It is possible to speed up the computation by running algorithm R_then_T in Step 2 for all initial rotation samples. The best estimate is then used as the initial estimate for a single run of R_and_T. In practice, this has been found to be good enough most of the time. Finally, in Step 3, the algorithms R_and_T_img or R_and_T_mod can replace R_and_T.

3.6. Prior Estimates

In many applications a prior estimate for the pose parameters *is* available; in many cases a prior covariance (or uncertainty) matrix of the pose parameters is also available. For instance, when tracking independently moving objects their location and orientation can be predicted by incorporating a motion model such as constant velocity or constant acceleration. Similarly in the mobile robot navigation domain, the location of the robot can be predicted by dead-reckoning from its previous location. The strength of belief in such predictions of the robot pose are captured by the prior covariance matrix. In this section, the iterative methods developed earlier to estimate pose are extended to handle the information represented by the prior estimate and its covariance matrix. The basic tool used is akin to extended Kalman filtering [15].

Let the initial estimate be denoted by the (6 × 1) vector $\mathbf{Q}$ and the associated prior covariance matrix by the (6 × 6) matrix Λ_Q. In the iterative system developed earlier, a linear system of equations (31) is solved at each iteration to determine the pose increments $\Delta\mathbf{T}$ for translation and rotation $\delta\boldsymbol{\omega}$. To incorporate the additional information available in the covariance matrix, this linear system of equations is modified as

$$(A + \Lambda_Q^{-1}) \begin{bmatrix} \Delta\mathbf{T} \\ \delta\boldsymbol{\omega} \end{bmatrix} = \mathbf{f} - \Lambda_Q^{-1}(\boldsymbol{\theta} - \mathbf{Q}). \tag{35}$$

TABLE 1
Average Absolute Error of Translation and Rotation in Camera Coordinates for Algorithm R_and_T

Noise			Rotation error			Translation error		
No. lines	θ (deg)	ρ (pixels)	$\delta\omega_x$ (deg)	$\delta\omega_y$ (deg)	$\delta\omega_z$ (deg)	ΔT_x (ft)	ΔT_y (ft)	ΔT_z (ft)
	Correct		0.00	0.00	0.00	0.00	0.00	0.00
5	1.0	1.0	0.24	0.15	0.04	0.21	2.03	1.16
5	5.0	5.0	1.20	0.79	0.19	1.08	10.14	6.20
5	1.0	5.0	0.24	0.16	0.04	0.21	2.04	1.18
5	5.0	1.0	1.19	0.78	0.19	1.08	10.14	6.20
14	1.0	1.0	0.07	0.06	0.08	0.03	0.77	0.02
14	5.0	5.0	0.34	0.30	0.39	0.17	3.80	0.12
30	1.0	1.0	0.03	0.05	0.06	0.06	0.48	0.06
30	5.0	5.0	0.16	0.24	0.31	0.32	2.39	0.32

Note. The average for each experiment is taken over 100 samples of uniform noise.

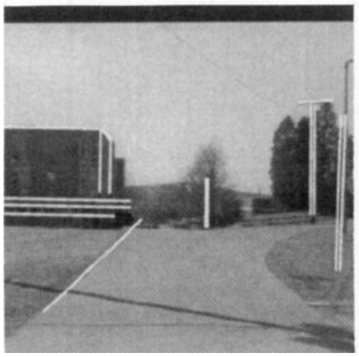

FIG. 9. Outdoor frame 4 with input image lines.

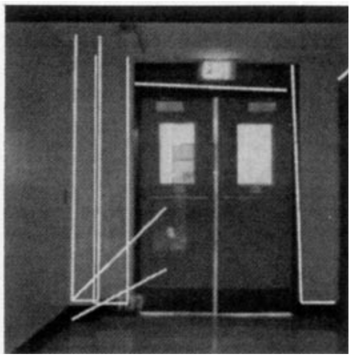

FIG. 12. Projection of model for indoor frame 3 using final estimate of algorithm R_and_T_img.

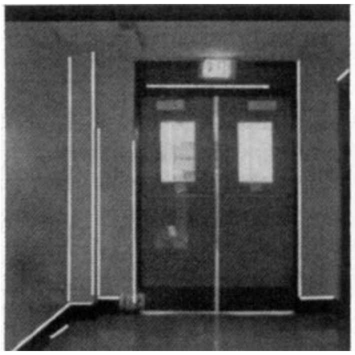

FIG. 10. Indoor frame 3 with input image lines.

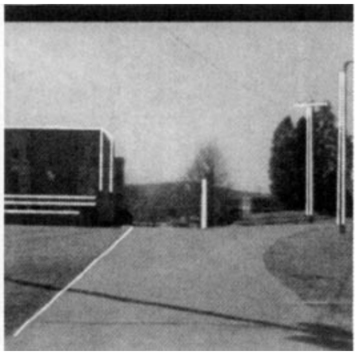

FIG. 13. Projection of model (outdoor frame 4) using final estimate of algorithm R_and_T_mod.

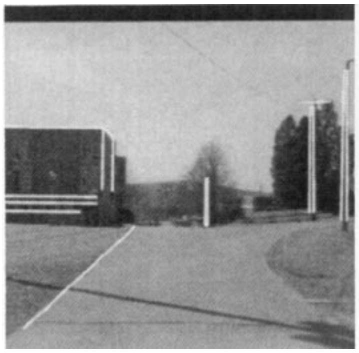

FIG. 11. Projection of model (outdoor frame 4) using final estimate of algorithm R_and_T_img.

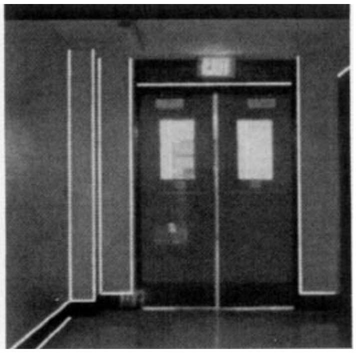

FIG. 14. Projection of model for indoor frame 3 using final estimate of algorithm R_and_T_mod.

In the above equation, $\boldsymbol{\theta}$ is the current pose estimate. Note that A is a (6×6) matrix while $\mathbf{f}$ and $\boldsymbol{\theta}$ are (6×1) vectors. Therefore, when prior estimates and covariance matrices are available Eq. (35) instead of Eq. (31) is solved at each iteration. The output covariance matrix of the pose parameters is given by $(A + \Lambda_Q^{-1})^{-1}$ evaluated at the final pose estimate.

3.7. Least-Squares Results Using Line Data

The development of the algorithms presented in this paper are part of a larger effort to enable the UMass robot "Harvey" to navigate the sidewalks and interior hallways of a part of the UMass campus [6]. Consequently, results using line data are presented for both indoor hallway images and outdoor sidewalk images. Figures 9 and 10 are examples of outdoor and indoor images respectively.

The indoor model was built by measuring distances with a tape measure and is accurate to approximately 0.1 ft [6]. The outdoor 3D model was built over two passes. In the first pass, blueprints of the campus, drawn to a scale of 40 ft to an inch, were used. Errors of up to 10 ft were found in the resulting 3D model; errors of this magnitude are unacceptable for our navigation goals. An error of 1 ft in the location of a 3D landmark, 50 ft away from the camera, can cause its projection to be displaced by 24 pixels in the image. In the second pass, landmarks were surveyed using theodolites. We believe most of our 3D model is now accurate to within 0.3 ft. Some landmarks, such as poles and posts, are difficult to position accurately, because of their cylindrical shape and their lack of any distinguishing points.

The images were acquired using a Sony B/W model AVC-D1 camera mounted on the robot vehicle. Linked to a Gould frame grabber, 512 by 484 pixel images are obtained, with a field of view of 24.0° by 23.0°. Calibration was not done for the image center; it was assumed to be at the frame center.

3.7.1. *Synthetic Data Results for R_and_T and R_then_T*

The synthetic data experiments were conducted for both R_and_T and R_then_T using the outdoor 3D model. The landmarks used for the outdoor scene experiments were the 3D lines forming the visible corner of the building, window lines, lampposts, telephone poles, and one sidewalk line (see Fig. 4). The camera was placed approximately 300 ft distant from the building. The two algorithms were run with three different sets of data lines, each set being perturbed by at least two different amounts of noise. Zero mean uniform noise was added to the ρ and θ of each image line (refer to Eq. (5)). In Tables 1 and 2 the noise for each simulation is specified in the ρ and θ columns. One pixel noise in ρ means that to the correct ρ of each line, we added a $\Delta\rho$, which was a random number in the range $[-1, +1]$. Similarly, 1° of noise in θ means that to the correct θ of each line, we added a $\Delta\theta$, which was a random number in the range $[-1, +1]$. Simulations were performed for a maximum of 1° or 5° noise in θ, and a maximum of 1 or 5 pixel noise in ρ.

The results presented in the tables are the average absolute error of the computed rotation and translation over 100 data samples, for each set of lines and each noise specification. The results for the R_and_T and R_then_T algorithms are shown in Table 1 and Table 2, respectively. Rotation and translation errors in these tables for synthetic data are specified with respect to the camera coordinate system (Fig. 1). The rotation errors are specified in terms of the error in degrees of the axis–angle 3D rotation vector. ΔT_x corresponds to error in translation in the direction along the rows in the image plane (in camera coordinates, see Fig. 4). ΔT_y corresponds to error in translation in the vertical direction in camera coordinates. ΔT_z corresponds to error in translation along the direction of the optical axis in camera coordinates.

The first set of 5 lines consisted of the four corner edges of the building visible in Fig. 4 and 1 window line in that same building. The second set of 14 lines consisted of the four corner edges of the building, as above, six lampposts and telephone pole lines, 3 more lines on the building, and 1 sidewalk line. The third set of 30 lines consisted of the above set of 14 lines plus a set of 16 virtual lines that were drawn between six real vertices in the scene.

A comparison of the results shows that the performance of the R_and_T algorithm (Table 1) is much better than the R_then_T algorithm (Table 2). With zero noise

TABLE 2
Average Absolute Error of Translation and Rotation in Camera Coordinates for Algorithm R_then_T

Noise			Rotation error			Translation error		
No. lines	θ (deg)	ρ (pixels)	$\delta\omega_x$ (deg)	$\delta\omega_y$ (deg)	$\delta\omega_z$ (deg)	ΔT_x (ft)	ΔT_y (ft)	ΔT_z (ft)
	Correct		0.00	0.00	0.00	0.00	0.00	0.00
5	1.0	1.0	1.08	5.06	0.62	11.44	13.96	51.16
5	5.0	5.0	3.19	14.65	1.62	32.69	39.85	149.40
14	1.0	1.0	0.29	0.29	0.18	0.35	2.37	0.23
14	5.0	5.0	1.50	1.56	0.91	1.92	12.44	1.27
30	1.0	1.0	0.09	0.10	0.13	0.40	1.01	0.36
30	5.0	5.0	0.45	0.50	0.66	2.09	5.05	1.82

Note. The average for each experiment is taken over 100 samples of uniform noise.

specified, both algorithms gave the correct result. For each set of lines and each specification of noise, R_and_T performs much better than R_then_T. The results for R_then_T are particularly bad for the five-line case. This can be explained by the observation that, in the five-line case, three of the lines form a trihedral junction. As noted before, trihedral junctions can give rise to an infinite number of translations. Thus, the translation result is determined from this infinite set solely by the two remaining lines, both of which are vertical and not too far from each other. With noise, therefore, we would expect large errors in translation. This problem is compounded even further when the rotation stage is separated from the translation stage as is the case in algorithm R_then_T.

In the results for both algorithms, the error decreases appreciably as the number of lines increases. In the R_and_T case (Table 1) results are shown for experiments with the five-line data set for two extra cases of noise. Examination of the results for these two cases in Table 1 shows an appreciably larger error when the noise in θ is 5°. However, when the noise in ρ is 5 pixels and the noise in θ is 1°, the errors are much smaller; in general noise in θ for lines is much more harmful than noise in ρ. Finally, in all experiments, the error in ΔT_y was often found to be larger than the errors in ΔT_x and ΔT_z. This is due to the fact that in the data sets for these experiments, the majority of the 3D lines are vertical.

3.7.2. *Synthetic Data Results for R_and_T_mod and R_and_T_img*

In this subsection, synthetic data results for algorithms R_and_T_mod and R_and_T_img are discussed. The results for the experiments are presented in Table 3. The 3D line model used for these experiments is the same as that built for the indoor hallway images (see Fig. 10). The camera was assumed to be 40 ft from the door in Fig. 10. The field of view and other intrinsic camera specifications are the same as those used for the synthetic data experiments discussed in the previous section. Given an input pose, the 3D model of 10 lines is projected to create a set of 2D data lines. The endpoints of the 2D lines are then corrupted by 2D gaussian noise. The gaussian noise has two components; the first is perpendicular to the 2D line and the second is along the length of the line. Noise for each endpoint was assumed to be independent. For all the experiments reported in Table 3 the standard deviation of the component of the noise perpendicular to the line was 1 pixel. The standard deviation of the component of the noise along the length of the line is specified as percentage of the length of the line. This is reported in column 2 in Table 3. For each noise specification, 1000 noisy sample sets are created and the two algorithms run on each of the noisy 2D data sets. First-order and second-order statistics are collected for each set of 1000 runs.

TABLE 3
Standard Deviation of Translation and Rotation Error in World Coordinates for Algorithms R_and_T_img and R_and_T_mod with High Prior Covariance Estimates for Translation

Noise			Rotation error			Translation error		
No. lines	Length (%)	Perp. (pixels)	$\delta\omega_x$ (deg)	$\delta\omega_y$ (deg)	$\delta\omega_z$ (deg)	ΔT_x (ft)	ΔT_y (ft)	ΔT_z (ft)
			Algorithm R_and_T_img					
	Prior estimate		57.30	57.30	57.30	94.86	94.86	94.86
10	1.0	1.0	2.77	0.46	2.48	0.25	0.82	0.70
10	5.0	1.0	2.79	0.47	2.50	0.25	0.83	0.70
10	10.0	1.0	2.83	0.48	2.53	0.26	0.84	0.71
10	20.0	1.0	3.03	0.54	2.71	0.27	0.89	0.77
10	30.0	1.0	6.68	1.35	5.27	0.96	1.72	1.49
10	40.0	1.0	10.33	2.32	8.45	2.00	2.44	2.12
			Algorithm R_and_T_mod					
	Prior estimate		57.30	57.30	57.30	94.86	94.86	94.86
10	1.0	1.0	2.69	0.46	2.40	0.25	0.79	0.68
10	5.0	1.0	2.69	0.46	2.40	0.25	0.79	0.68
10	10.0	1.0	2.70	0.46	2.40	0.25	0.80	0.68
10	20.0	1.0	2.79	0.48	2.48	0.27	0.82	0.70
10	30.0	1.0	2.92	0.48	2.59	0.28	0.86	0.73
10	40.0	1.0	3.06	0.45	2.69	0.27	0.90	0.76

Note. The statistics for each experiment is taken over 1000 samples of gaussian noise.

From the first-order statistics (i.e., estimation of the mean) it is observed that the final estimates are unbiased. The second-order statistics are the experimentally derived covariance matrices of the output pose parameters. The square root of the diagonal values (or standard deviations) of each of the pose parameters for each noise specification and each algorithm are reported in Table 3.

From Table 3, it can be seen that the two algorithms perform comparably as long the component of noise along the length of the line is small. But when the standard deviation of the noise along the length of the line becomes 20% or more of the length, then the results for algorithm R_and_T_mod sharply improve over algorithm R_and_T_img. This confirms what was predicted earlier in the discussion: algorithm R_and_T_mod performs more robustly than algorithm R_and_T_img when there is significant line fragmentation in the image data.

3.7.3. *Real Data Results for R_and_T_mod and R_and_T_img*

In this subsection, results for algorithms R_and_T_mod and R_and_T_img over two real data outdoor and indoor sequences are presented. The first sequence consists of six outdoor frames. The first and fourth frames are shown in Figs. 4 and 9, respectively. For the outdoor sequence, the camera was moved in an approximate forward motion 25 ft along the walkway. Each subsequent frame was taken after a movement of 5 ft down the walkway. The 2D image lines were taken from the output of a 2D line matching system [2]. For each frame, column 2 in Table 4 gives the number of lines the 2D line matcher was able to correctly match. The 2D matcher does not return the original image lines; rather it returns the location of the matched 2D lines as predicted by the final 2D–2D pose. One consequence of this is that the input lines used in our experiment for the outdoor sequence are not broken and fragmented like the original extracted image lines may be. Figure 9 shows the input 2D lines as returned by the line matcher for frame 4 of the outdoor image sequence. These were used as input to our algorithm along with the 3D model.

The error in the estimated location by algorithms R_and_T_img and R_and_T_mod is given in Table 4. In the table, the translation direction x is the horizontal direction oriented with the long side of the hallway (for indoor images) or the walkway (for outdoor images), the y direction is the horizontal direction perpendicular to the long sides of the hallway or walkway, and the z direction is aligned with the direction of gravity. Ground truth for rotation was not available; thus the algorithms were compared only with respect to location estimation. A qualitative estimation of the orientation accuracy can be obtained by seeing how well the projected model aligns with the original image data (e.g., as in Figs. 13 and 14). In most cases, for the outdoor frames the robot is located to within 1 or 2 ft. The measurement errors in Table 4 are approximate to 0.5 ft. The precise location of the camera is not known. It is better to judge the performance of the algorithm by looking at the projections of the 3D landmarks on the image after the pose has been estimated. Figures 11 and 13 are the projection of the 3D model lines

TABLE 4
Estimated Errors of Translation in World Coordinates for Algorithm R_and_T_img and R_and_T_mod

		Translation error					
		R_and_T_img			R_and_T_mod		
Frame no.	No. lines	ΔT_x (ft)	ΔT_y (ft)	ΔT_z (ft)	ΔT_x (ft)	ΔT_y (ft)	ΔT_z (ft)
		Outdoor frames					
1	17	0.47	0.02	0.54	0.42	0.05	0.54
2	15	0.41	0.56	0.59	0.37	0.86	0.85
3	12	2.03	0.48	0.41	2.34	0.58	0.50
4	7	0.64	1.02	0.88	0.77	1.16	0.90
5	13	1.30	0.87	0.61	1.32	0.87	0.61
6	13	1.72	1.23	0.79	1.41	1.43	0.88
		Indoor frames					
1	22	0.24	0.04	0.04	0.24	0.05	0.01
2	22	0.10	0.17	0.02	0.12	0.08	0.01
3	12	1.58	4.87	4.25	0.11	0.11	0.03
4	10	2.95	1.06	0.79	0.10	0.03	0.01

Note. Real data results for outdoor and indoor frames without outliers.

using the poses estimated by the algorithm R_and_T_img and R_and_T_mod for outdoor frame 4, respectively. As can be seen, in all cases there is fairly good alignment between the 3D model and the original image. The two algorithms perform comparably for this sequence. This is probably due to the fact that the output of the 2D matcher returns whole lines and there is no fragmentation of the 2D image lines.

The results of the two algorithms R_and_T_img on four indoor hallway frames are also given in Table 4. The camera location for these frames ranged from 32 to 23 ft from the door. The results in Table 4 are with no outliers in the input data. The line correspondences for the four frames were obtained from a motion line tracking system [27]. In this case, the input lines used are the extracted image lines. As a result, some of the input lines are fragmented. If more than one image line was matched to a model line, then the longest matched line was selected for the input set given to the pose algorithms. The input 2D lines and the final estimated projection of the 3D model lines by the algorithms R_and_T_img and R_and_T_mod for indoor frame 3 are shown in Figs. 10, 12, and 14, respectively.

Algorithms R_and_T_mod is able to locate the camera within 0.3 ft for all four frames. In contrast, algorithm R_and_T_img performs poorly for the last two frames. This is due to the severe and noisy fragmentation of the image lines in the last two frames. The final estimate of algorithm R_and_T_img for indoor frame 3 is wrong by more than 4 ft (see Table 4). In Fig. 12, it can be seen that the projected model lines (especially, the baseboard lines on the left-hand side) for indoor frame 3 are not at all aligned with the input image. The line extraction algorithm has recovered only a small and noisy fragment of the lower left baseboard line (see Fig. 10). This input line is an outlier for algorithm R_and_T_img and causes the final projection to be skewed. Note that if the lower left baseboard line is removed from this data set, the algorithm R_and_T_img locates the robot within an inch of the correct location.

From the above experiment for indoor frame 3, two facts are demonstrated. The first, of course, is that algorithm R_and_T_mod is more robust than algorithm R_and_T_img. The second fact is that even a single outlier can cause a least-squares algorithm to fail catastrophically. Thus, we must develop algorithms which are robust with respect to outliers; this is the subject of the next section.

4. ROBUST METHODS

This section develops and analyzes pose determination techniques which are robust with respect to outliers or gross errors in the data. In the pose problem, outliers occur either due to incorrect image-to-world correspondences or if parts of the 3D model are incorrect. Traditionally, least square techniques have been used for regression analysis or model fitting. In Sect 3, least-squares techniques were presented to solve the pose problem. Least squares is optimum and reliable when the underlying noise in the data is gaussian. However, when outliers are present in the data, the gaussian assumption is violated and the least-squares result is skewed in order to make the data approximate a gaussian. Because of the skewing of the result, trying to detect outliers by comparing the residual errors of each line with a threshold will not work. Throwing away one line at a time and doing least squares on the remaining subset also does not work when more than one outlier is present.

Statisticians have suggested many different robust techniques [13, 17, 24, 25, 31] to handle outliers and these techniques are currently gaining popularity in computer vision [1, 7, 10, 14]. A measure used to analyze these robust algorithms is the breakdown point: the smallest fraction of outliers present in the input data which may cause the output estimate to be arbitrarily wrong. Algorithms based on minimizing $L1$, $L2$, or Lp error measures have breakdown points of $1/n$ where n is the number of data items. Another measure of robust statistical procedures is their "relative efficiency" [14, 25]. It is defined in Kim *et al.* [14] as the "ratio between the lowest achievable variance for the estimated parameters (the Cramer–Rao bound) and the actual variance provided by the given method," so that the best possible value is 1. Kim *et al.* also note that "the least mean square estimator in the presence of gaussian noise has an asymptotic (large sample) efficiency of 1 while the median's efficiency is only 0.637" [14, 24]. As we shall see, there is a trade-off between algorithms with high breakdown points versus those with high efficiency. Finally, most research in robust statistics appears to have been done for linear problems. When applying these techniques to nonlinear problems, another important consideration is the initial estimate. Nonlinear problems are often solved iteratively using an initial estimate. How close must this initial estimate be for the robust technique to work?

4.1. Previous Work: Outliers

There are two major sets of statistical techniques for handling outliers. The first attempts to detect outliers before forming a robust estimate. The goal is to find "leverage points," i.e., data points which are on the outskirts of the data cluster. These points, if wrong, can have the largest influence on the final estimate. Standard outlier detection techniques are based on the diagonal entries of the "Hat" matrix, Mahalanobis distance, etc.; these generally work for certain kind of outliers and often cannot handle more than one outlier.

The second set of robust statistical techniques detects outliers and computes robust estimates simultaneously. Most of these techniques attempt to define objective functions whose global minimum would not be significantly affected by outliers. The techniques analyzed in the next subsection are of this kind. Standard among these are M-estimates (maximum likelihood type estimates), L-estimates (linear combinations of order statistics), and R-estimates (estimates based on rank transformations). Functions minimized by M-estimate techniques attempt to bound the maximum possible error value (e.g., the biweight "redescending" function suggested by Tukey; see Fig. 15) or bound its rate of change (e.g., Huber's minimax function). Most of these M-estimate techniques have been shown to have breakdown points of $1/(p + 1)$ or lower [17, 25], where p is the number of unknowns ($p = 6$ for the pose refinement problem). They have high efficiencies, however, typically more than 0.9. Huber [13] suggests iterative algorithms to minimize these error functions and these algorithms are relatively fast to compute. However, it is important to have good initial estimates because of the number of local minima.

Haralick *et al.* [10] adapt these M-estimate techniques for the pose problem using point data. They use the redescending function suggested by Mosteller and Tukey [24] and the "minimax" function suggested by Huber [13], respectively. These techniques are also adapted here for the pose problem using line correspondences. Better results were obtained using Mosteller and Tukey's function as compared to Huber's. An algorithm "Tuk_wts" using Tukey's error function and the infinite image line constraint for pose is presented in Section 4.2.

Another robust technique suggested by Rousseeuw and Leroy [25] is based on the minimization of the median of the squares of the residual errors (LMS: least median of squares). This method has a breakdown point of 0.5 and consequently is able to handle data sets which contain less than 50% outliers. However, since the median is not a differentiable function, it has to be minimized by a combinatorial method and is comparatively very slow compared to the M-estimate techniques. To minimize the median square error, a brute-force technique is employed that computes the median square error using all p size data subsets, where p is the number of unknown parameters. Rousseeuw and Leroy [25] believe that only techniques of this brute force nature will be able to achieve a high breakdown point. They suggest that by sacrificing 100% probability of correctness, a large gain in computational speed can be obtained by considering only a small random set of the minimal subsets (this point is discussed further in Section 4.4). Using the best median pose, data points whose residual error is greater than a certain threshold are weeded out as potential outliers. The threshold may be either fixed a priori or determined based on the computed scale (standard deviation) of the nonoutlier gaussian noise. Finally a reduced weighted least squares (RLS) or a one step M-estimate is done, using the median estimate as the initial guess. This greatly improves the relative efficiency of the median-based estimate.

In Section 4.3, two algorithms based on the least-median-squares technique are developed. The first ("Med_R_and_T_img") minimizes the median of the square of the alignment error given by the infinite image line constraint discussed in Section 2 (see Eq. (8)). The second ("Med_R_and_T_mod") minimizes the median of the square of the alignment error given by the infinite model line constraint (see Eq. (21)). In both cases the random sampling techniques suggested by Rousseeuw and Leroy [25] to achieve higher computational speed are implemented and finally a weighted reduced least squares is done to improve the efficiency.

4.2. M-Estimation Techniques: The Tuk_wts Algorithm

M-estimation techniques, developed by Huber [13] and other statisticians, minimize the sum of a function $\rho(e_i/s)$, where e_i is the error function for the ith data vector and s is a scaling factor:

$$Minimize \sum_{i}^{n} \rho(e_i/s). \tag{36}$$

The function ρ is designed to be a continuous, symmetric function with minimum value at $u = 0$ [13, 31]. Also $\rho(u)$ must be monotonically increasing from $u = 0$ to ∞ and from $u = 0$ to $-\infty$.

The ρ function proposed by Mosteller and Tukey [24] is

$$\rho(u) = \begin{cases} u^2/2 - u^4/2a^2 + u^6/6a^4 & \text{if } |u| \leq a \\ a^2/6 & \text{otherwise.} \end{cases} \tag{37}$$

Figure 15 shows Mosteller and Tukey's function and its derivative plotted with a set to 2.0. In this function, $\rho(u)$ approximately varies as the square of u for small values and then tapers off to a constant maximum value of $a^2/6$. Thus the effect for any outlier data element cannot be arbitrarily large. A rationale for this function is that for small error values, u corresponds to gaussian noise and thus for optimum relative efficiency their square value must be minimized. However, as the error grows, the data element is probably an outlier and therefore its influence must be bounded. Blake and Zisserman [3] use a variation of Mosteller and Tukey's function in solving the problem of fitting piece-wise smooth surfaces to range data. They provide another rationale for the above func-

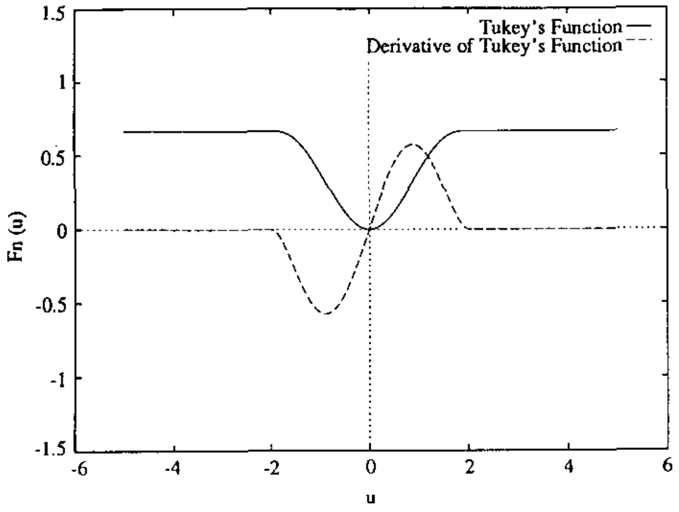

FIG. 15. Tukey's biweight redescending function and its derivative.

tion: the constant maximum value is assumed to be the cost of assigning a data element to be an outlier. Thus nonoutlier data elements are minimized as the sum of squares of their error values and there is a cost of assigning a data element to be an outlier. The final estimation is a trade-off between minimizing the number of outliers and the fitting error for the nonoutlier data points.

Inherent in these M-estimation techniques is a simultaneous computation of a scale s. The scale corresponds to the standard deviation of the residual errors. If a good estimate of the standard deviation of the errors of nonoutlier data can be made, then data points whose error lies beyond a certain number of standard deviations from the center (that is, in the "tails" of the distribution) can be classified as outliers. The estimate of scale therefore itself must be robust and not affected by outliers. M-estimation techniques vary both on their choice of the ρ function and the method used to compute scale. In the algorithm presented below, scale s is computed by the equation

$$s = \frac{\text{median}_i\, |e_i|}{0.6745} \tag{38}$$

where 0.6745 is one-half of the interquantile range of the gaussian normal distribution $N(0, 1)$. This equation relates the scale (standard deviation) of a gaussian distribution to the median of the absolute values of a sampled set of data points. It is based on the fact that the median of the absolute values of random numbers sampled from the gaussian normal distribution $N(0, 1)$ is estimated to be approximately 0.6745.

Huber [13] suggests two methods for minimizing the error functions in Eq. (37). The two methods are the modified residual method and the modified weights method [13, 10]. In our experiments better performance was obtained using the modified weights method; consequently only this version will be presented here.

The modified weights method is an iterative reweighting least-squares (IRLS) algorithm. Let θ_j represent the set of unknown parameters (rotation and translation in our case) to be estimated. Differentiating the error expression in Eq. (36) with respect to each parameter θ_j we get the set of equations

$$\sum_i \psi\left(\frac{e_i}{s}\right)\frac{\delta e_i}{\delta\theta_j} = 0.0, \tag{39}$$

where ψ is the derivative of the ρ function with respect to u. The ψ function used in the Tuk_wts algorithm is obtained by differentiating Mosteller and Tukey's ρ function as given in Eq. (37):

$$\psi(u) = \begin{cases} u\left(1 - \dfrac{u^2}{a^2}\right)^2 & \text{if } |u| \le a \\ 0.0 & \text{otherwise.} \end{cases} \tag{40}$$

The ψ function is also sketched in Fig. 15. Note that for small values of u around $u = 0.0$ it is almost linear and then it slowly tapers off to zero on both ends.

Equation (39) can be written in the standard weighted form as

$$w_i = \frac{\psi(e_i/s)}{e_i/s} \tag{41}$$

$$\sum_i w_i e_i \frac{\delta e_i}{\delta\theta_j} = 0.0. \tag{42}$$

If the weights w_i in the above equation (42) are held constant and the residual error e_i is a linear function of the unknown set of parameters, then a linear system of equations in the unknown parameters θ_j is obtained. The modified weights algorithm solves the equations (42) iteratively. At each iteration the weights w_i are held constant and equal to the values computed using the previous iteration's estimates of the unknown parameters θ_j and the linear system of equations is solved to obtain new estimates. The iterative procedure is repeated until the parameter values converge to a final value. Huber [13] proves that the above iterative procedure leads to a minimum (possibly a local minimum) of the objective function as given in Eq. (36) for linear e_i residual error functions. Note, although e_i may be linear, $\rho(e_i)$ is not a quadratic function.

In this iterative procedure, if the error of a data element is greater than a for a current estimate, then by the weighting function defined in Eq. (41) its weight will be zero and therefore will not contribute to the new estimation. Data elements with small errors will have almost unit weight while data elements with slightly larger errors (but less than a) will contribute partially (between 0 and 1) to the final fit. This partial contribution of these data elements greatly improves the relative efficiency of the final estimation. This can be intuitively understood by recalling that the noise model assumes that most of the data is contaminated by gaussian noise and only some of the data elements are outliers. The important issue is the criteria for deciding when a data element should be classified as an outlier (i.e., the number of standard deviations to be used as an outlier threshold). The relative efficiency is improved by using all data elements which are not outliers for the final fit. The above procedure does not automatically cut off all data elements beyond a certain threshold; instead it gives increasingly small weight to data elements with larger errors until a final weight of zero is given for data elements whose error is larger than a standard deviations.

We now turn to the problem of applying this technique to the minimization of the pose error functions, which are nonlinear. In Section 3.4, an iterative method was developed to minimize the sum of squares of the error function. This method entailed linearizing the error function about the current estimate and then solving for the small increments in translation (ΔT) and rotation ($\delta\omega$). Equations (29) and (30) shown in Section 3.4 are the linear system of equations solved at each iteration of the R_and_T3 algorithm described there. An equivalent set of equations can be developed for algorithms R_and_T_img and R_and_T_mod. The error for each data element is weighted by w_i. To incorporate the robust techniques described above into the minimization procedure for R_and_T (and by extension for R_and_T_img and R_and_T_mod) described in Section 3.4, the only change required is to replace the weights in Eqs. (29) and (30) by the weighting function given in Eq. (41). This procedure leads to rapid convergence and a robust estimate, as will be seen in the experiments described in Section 4.4. For most of our experiments, convergence is reached in about 10 iterations.

The Tuk_wts algorithms developed here solves the system of equations (29) and (30) corresponding to algorithm R_and_T_img at each iteration and composes the results with the previous estimates of rotation and translation. The weights are calculated using Eq. (41). The steps used in the algorithm are identical to the steps used in algorithm R_and_T_img. At each iteration there is also an estimation of scale using Eq. (38). Finally, although not done here, the R_and_T_mod algorithm can also be similarly modified to a robust M-estimate version.

4.3. LMS Technique

In this section another set of robust techniques for the pose determination problem are developed. These techniques were developed by Rousseeuw and Leroy [25] for the linear regression problem. They are based on minimizing the median of the square of the error function over all data elements:

$$\text{Minimize } \underline{\text{Median}}_i\, e_i^2 \tag{43}$$

It was noted earlier that in least-squares systems the large error values of the outlier data elements causes a skew of the final fit. If the data elements are ranked in ascending order according to error values, the median corresponds to the error of the middle data element. Minimizing the median, therefore, ignores the errors of the larger ranked half of the data elements. Thus, this method can automatically perform robustly in situations where less than 50% of the data elements are outliers; it has a breakdown point of $\frac{1}{2}$.

This procedure can be modified to incorporate a priori knowledge of the level of contamination of the data set with outliers. For example, if in a particular application, it is guaranteed that no more than 30% of the points are outliers, then the 70% ranked error could be minimized rather than the median. In the absence of any such guarantees, the median is the best choice to minimize. This corresponds to finding that pose which fits (with minimum error) at least half of the data elements. In other words, the final estimated pose is the best consensus fit over all subsets of size equal to half the number of the data elements.

To develop robust pose algorithms, any of the pose alignment error functions developed in Section 3 can be substituted for e_i in Eq. (43). In this section, two such robust algorithms Med_R_and_T_img and Med_R_and_T_mod are developed, based on the alignment errors used in the least-squares algorithms R_and_T_img and R_and_T_mod, respectively. In Med_R_and_T_img the following objective function E_{m1} is minimized

$$E_{m1} = \underline{\text{Median}}_i \left(\left(\frac{\mathbf{N}_i \cdot (R(\mathbf{p}_{i1}) + \mathbf{T})}{(R(\mathbf{p}_{i1}) + T)_z} \right)^2 + \left(\frac{\mathbf{N}_i \cdot (R(\mathbf{p}_{i2}) + \mathbf{T})}{(R(\mathbf{p}_{i2}) + T)_z} \right)^2 \right). \tag{44}$$

E_{m1} is a modification of the objective function E_2 given in Eq. (24) in Section 3.3. The right-hand side corresponds to the median of the total alignment error for each line. The total alignment error is based on the infinite image line constraint and is the sum of squares of the perpendicular distances from the projected model endpoints to the infinitely extended image line.

The Med_R_and_T_mod algorithm is based on the infinite model line constraint and the following objective function E_{m2} is minimized:

$$\mathbf{C}_i = (\mathbf{p}_{2i} - \mathbf{T}_w) \times (\mathbf{p}_{1i} - \mathbf{T}_w) \tag{45}$$

$$\mathbf{B}_j = \left(\frac{I_{xj}}{s_x}, \frac{I_{yj}}{s_y}, 1\right)^T \tag{46}$$

$$E_{\mathrm{m2}} = \underline{\mathrm{Median}}_i \frac{1.0}{M_i^2} ((R^{\mathrm{T}}(\mathbf{B}_{i1}) \cdot \mathbf{C}_i)^2 + (R^{\mathrm{T}}(\mathbf{B}_{i2}) \cdot \mathbf{C}_i)^2). \tag{47}$$

E_{m2} is a modification of the objective function E_3 given in Eq. (25) of Section 3.3; M_i is defined in Section 2.2. Again, the right-hand side corresponds to the median of the total alignment error for each line. In this case, however, the total alignment error is based on the infinite model line constraint and is the sum of squares of the perpendicular distances from the image line endpoints to the infinitely extended projected model line.

Since the median is not a differentiable function, E_{m1} and E_{m2} must be minimized by combinatorial methods. In the method adopted here, candidate poses are generated by using the least-squares algorithms developed in Section 3 on subsets of the data elements. The pose which gives the minimum median error across all data elements is chosen as the optimal median pose. The goal is to find at least one subset which has no outliers in it; this should give the minimum median error. The "pose determination problem" needs a minimum of three input lines. Thus, subsets of line data elements used to generate the candidate poses must be of size m ($m \geq 3$). Typically, poses are generated from all subsets of size m of the data elements. Experimentally, we have found the choice of m is important. The larger the size of the subsets, the greater the probability of them having an outlier. However, choosing $m = 3$, the minimum as suggested by Rousseeuw and Leroy [25], often leads to local minima. Good results are typically obtained with $m = 6$ or higher. This is because the poses estimated by using just three line data sets have large variances and some of the nonoutlier lines can get labeled as outliers.

To speed up the computation, instead of using all subsets, only a random set of all size m subsets is used. If ε is the fraction of contaminated data and we choose k different random subsets of size m, then the probability P that all k different subsets will contain at least one or more outliers is

$$P = (1 - (1 - \varepsilon)^m)^k. \tag{48}$$

The probability that at least one random subset has no outliers is given by $(1 - P)$. This is the probability that the correct answer will be found by the median algorithm. For example, if we want the correct answer with 99% probability and expect no more than 30% outliers when using subsets of size 3 ($m = 3$), then only 37 out of the 1140 subsets (for a set of 20 lines) need to be randomly chosen. In practice, however, we find that a much larger set needs to be chosen. Finally, some subsets will lead to degenerate solutions because all lines are parallel, etc. This can be detected before any further processing of the subset is done. A simple method to detect degenerate subsets in the case of 3 lines is to threshold on the determinant of the matrix whose rows are the unit direction vectors of the 3D lines.

Using the best median pose, data points whose residual error is greater than a certain threshold are weeded out as potential outliers. The threshold may be either fixed a priori or determined based on the computed scale (standard deviation) of the nonoutlier gaussian noise. In many of our experiments, it is assumed that the scale of the line fitting process to edge data is 2 pixels. Equation (38) developed in the previous section may also be used to compute the scale. Finally, the weighted least-squares algorithm (R_and_T_img) or (R_and_T_mod) are run on the remaining lines. This last step greatly improves the "relative efficiency" of the robust algorithm.

The algorithms Med_R_and_T_img and Med_R_and_T_mod are summarized as follows:

Step 1. Select k random subsets of size m from the input data.

Step 2. For each subset, determine the pose by using R_and_T_img or R_and_T_mod. Estimate the residual error for all n lines given this pose and find the median square error.

Step 3. Select the pose which gives the minimum median error E_{m1} or E_{m2} and compute the scale "s" (if not known a priori) using Eq. (38).

Step 4. Filter out lines as outliers whose squared residual error for that pose is greater than $(a \times s)^2$; a is an algorithm parameter and is set equal to 2.0 for all experiments discussed in Section 4.4.

Step 5. Minimize the error function given in Eq. (24) or (25) on the remaining lines using the least-squares algorithm R_and_T_img or R_and_T_mod and return the estimated translation and rotation as the final output.

4.4. Robust Methods: Results

In this section we present and analyze the results of running the three algorithms Tuk_Wts, Med_R_and_T_img, and Med_R_and_T_mod on real image data. The results are presented for both the indoor hallway images (Fig. 16) and the outdoor sidewalk images (Fig. 20). The indoor and outdoor images are similar to the ones used in Section 3.7 for the least-squares experiments. In some cases, the data used for experiments in Section 3.7 were artificially altered to create outliers. In other cases, the data are directly an output of the 2D matching system developed by Beveridge *et al.* [2]. The camera parame-

TABLE 5
Estimated Errors of Translation in World Coordinates for Algorithms R_and_T_img, MED_R_and_T_img, and Tuk_Wts; High Covariance Case

		R_and_T_img			Med_R_and_T_img				Tuk_Wts		
		Translation error			Translation error				Translation error		
Frame no.	No. lines	ΔT_x (ft)	ΔT_y (ft)	ΔT_z (ft)	ΔT_x (ft)	ΔT_y (ft)	ΔT_z (ft)	Outliers fnd.	ΔT_x (ft)	ΔT_y (ft)	ΔT_z (ft)
					Indoor frames						
1	24	0.35	0.26	0.00	0.08	0.01	0.01	7	0.11	0.01	0.02
2	26	0.87	0.48	0.08	0.19	0.02	0.02	11	0.47	0.32	0.06
3	24	0.63	0.09	0.02	0.01	0.42	0.35	10	0.13	0.01	0.01
4	15	0.57	0.03	0.23	0.55	0.22	0.10	7	0.64	0.25	0.29
5	16	0.43	0.23	0.03	0.27	0.01	0.07	5	0.27	0.01	0.10
6	16	0.74	0.37	0.47	0.65	0.18	0.23	4	0.59	0.26	0.30
7	46	0.38	0.25	0.41	0.11	0.21	0.35	17	0.06	0.02	0.07
8	19	5.67	0.27	0.56	0.05	0.38	0.32	9	0.04	0.00	0.01
					Outdoor frames						
1	17	10.42	2.76	2.38	0.95	0.36	1.22	7	1.16	0.36	0.14
3	15	40.91	13.56	10.19	2.39	0.21	0.12	5	36.72	13.41	13.46

Note. Prior standard deviation (σ) of the initial robot pose estimate is 94.86 ft and 1 rad for each axis of location and orientation, respectively.

ters and the indoor and outdoor models used for the experiments are exactly the same as those described in Section 3.7. Tables 5 and 6 show the results of the three algorithms on eight indoor hallway images and three outdoor walkway images.

The Med_R_and_T_img and Med_R_and_T_mod algorithms were run on both the indoor and outdoor data sets with the same set of parameters. In each case, 500 random sample sets of size 6 were generated. The scale of the gaussian noise was assumed to be 2 pixels for all lines. Using the best median pose, any line whose alignment error was more than 4 pixels was declared to be an outlier and was given zero weight for the final least-squares fit. Note, instead of choosing the scale of gaus-

TABLE 6
Estimated Errors of Translation in World Coordinates for Algorithms R_and_T_mod, MED_R_and_T_mod; High Covariance Case

		R_and_T_mod			Med_R_and_T_mod			
		Translation error			Translation error			
Frame no.	No. lines	ΔT_x (ft)	ΔT_y (ft)	ΔT_z (ft)	ΔT_x (ft)	ΔT_y (ft)	ΔT_z (ft)	Outliers Fnd.
				Indoor frames				
1	24	0.26	0.20	0.03	0.02	0.47	0.44	6
2	26	0.75	0.47	0.05	0.20	0.03	0.02	9
3	24	0.14	0.12	0.02	0.04	0.04	0.01	3
4	15	0.50	0.04	0.10	0.02	0.08	0.00	4
5	16	0.34	0.34	0.12	0.15	0.01	0.07	3
6	16	0.77	0.59	0.11	0.85	0.49	0.33	4
7	46	0.31	0.20	0.61	0.14	0.01	0.03	17
8	19	5.47	0.30	0.78	0.05	0.38	0.32	9
				Outdoor frames				
1	17	10.41	2.82	2.28	1.00	0.38	1.27	7
3	15	41.23	13.90	7.91	2.53	0.19	0.10	5

Note. Prior standard deviation (σ) of the initial robot pose estimate is 94.86 ft and 1 rad for each axis of location and orientation, respectively.

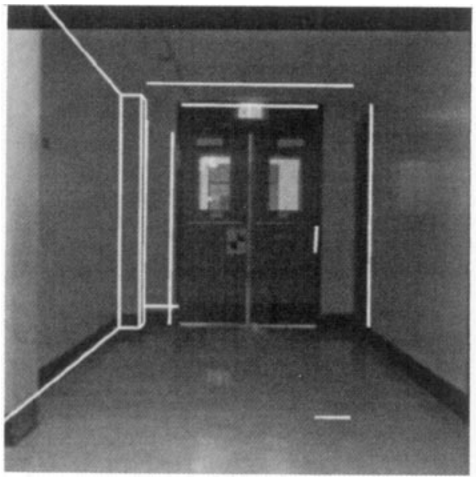

FIG. 16. Indoor frame 8 with input image lines.

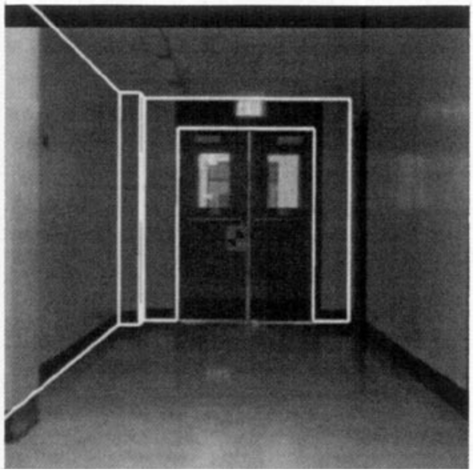

FIG. 18. Projection of model for indoor frame 8 using final estimate of algorithm Med_R_and_T_mod with high prior covariance matrix.

sian noise a priori, we could have computed it from the residual errors of the inliers computed by the LMS algorithm. For the experiments reported in this paper, similar results were obtained in both situations. In each case, the prior covariance matrix was a diagonal matrix. In Tables 5 and 6, the prior covariance matrix specified for the least-squares and median-based algorithms had high diagonal values corresponding to standard deviations of 94.86 ft for translation terms and 1 rad or 57.3° for the rotation terms. It was ensured for all experiments that the initial estimates used were within a standard deviation of the correct estimate. Finally, the Tuk_wts algorithm was always run with the threshold a in Eq. (40) set to 2.0.

The results of the least-squares algorithms R_and_T_img and R_and_T_mod on the same data sets are also presented in the tables. The performance of the least-squares algorithms gives a measure of the severity of the outliers in each of the data sets. Generally, the more gross an outlier is with respect to the noncorrupt data the easier it is to detect and remove. This is actually more true for the median algorithms. Severe outliers can affect the Tuk_Wts algorithm because of its low breakdown point. However, the robust algorithms must perform comparably or better than the least-squares algorithms both when the least squares algorithms completely fail and when they do reasonably well. The results presented in the Tables 5 and 6 document both these cases. The first case is the first seven frames of the indoor hallway images. From any of the four tables, it can be observed that the least-squares algorithms performed reasonably well

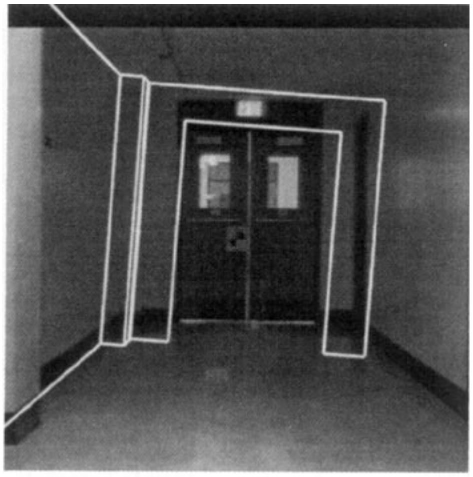

FIG. 17. Projection of model for indoor frame 8 using final estimate of algorithm R_and_T_mod with high prior covariance matrix.

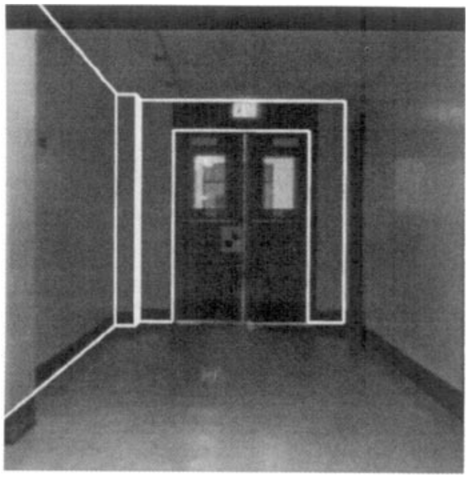

FIG. 19. Projection of model for indoor frame 8 using final estimate of algorithm Tuk_wts. Note that the projection of model using final estimate of Med_R_and_T_mod with low prior covariance is almost identical.

for the first seven indoor frames. The outliers in these cases have not very significantly affected the performance of the least-squares algorithms and the robust algorithms perform only slightly better. The average error in locating the robot for the first seven indoor frames by the least-squares algorithms R_and_T_img and R_and_T_mod when run with high initial covariance matrices is 0.676 and 0.585 ft, respectively. The high covariance setting corresponds to standard deviations of 94.86 ft for location along x, y, and z directions and 1 rad for all rotation angles. The average values are calculated from data in Tables 5 and 6. Note that R_and_T_mod performs slightly better than R_and_T_img. The robust algorithms perform better than both the least-squares algorithms. From the data shown in the Tables 5 and 6, the average error in locating the robot for the seven indoor frames by algorithms Med_R_and_T_img, Tuk_wts, and Med_R_and_T_mod (run with high initial covariance matrices) is 0.405, 0.387, and 0.363 ft, respectively. Note again that Med_R_and_T_mod performs slightly better then Med_R_and_T_img.

In contrast to the first seven indoor frames, the least-squares algorithms totally fail in locating the robot for indoor frame 8 and outdoor frames 1 and 3. The results for indoor frame 8 and outdoor frame 3 are discussed in more detail in the next two subsections.

4.4.1. *Indoor Frame 8*

Figure 16 shows the input set of lines for indoor frame 8. This data set of 19 lines has eight outliers, i.e., approximately 40% of the data is contaminated. Only the 11 lines on or close to the left wall are correct. Both the least-squares algorithms R_and_T_img and R_and_T_mod fail (for all variance settings) on this data set. Their final pose estimates are more than 5 ft off from the correct location. Figure 17 shows the projection of the model using the pose estimated by algorithm R_and_T_mod (run with the low initial variance setting) and as can be seen from the figure, it is quite skewed.

From Tables 5 and 6, the error in locating the robot for this frame by the robust algorithms Med_R_and_T_img and Med_R_and_T_mod (run with high initial covariance) is about 0.38 ft in the y axis and 0.32 ft in the z axis (or vertical direction). Figure 18 shows the projection of the model using the pose estimated by Med_R_and_T_mod when run with high initial covariance. In that figure, it can be noted that the line in depth in the top left corner of the image is slightly skewed. All other lines are fairly well aligned. This line has been detected as an outlier. Note that in Table 6, 9 lines have been labeled as outliers. Of the 11 lines in the data set for indoor frame 8, only 2 are lines in depth. The rest of the lines are coplanar and are clustered near the door. The best pose returned by the median is a consensus of only half the number of lines. As a result, one of the lines in depth remains out of the best consensus set and becomes an outlier.

In contrast, when algorithm Med_R_and_T_mod is run using subsets of size 6 or 8 with a low prior covariance setting, the height of the robot is pinned and the final estimate of location returned by the algorithm is within 0.05 ft and only the eight outlier lines have been detected as outliers. Note that the low covariance setting corresponds to standard deviations of 3 ft for location along x and y directions, 0.1 ft for z direction, and 1 rad for all rotation angles. Figure 19 shows the projection of the model using the pose estimated by Med_R_and_T_mod when run with low initial covariance. In this case, all lines, including the top left line in depth, are well aligned.

Finally, the Tuk_wts algorithm performed very well on this data set, locating the robot to within 0.04 ft. The projection of the model for indoor frame 8 using the pose estimated by the Tuk_wts algorithm is almost identical to the projection shown in Fig. 19.

4.4.2. *Outdoor Frame 3*

The outdoor frame 3 data set (shown in Fig. 20) is another case where the least-squares algorithm fails completely. The outlier here is due to the telephone pole being mismatched to the street light. Thus, 3 of the 15 input lines are clear outliers. The location of the robot returned by the least-squares algorithm is off by about 40 ft for the high initial covariance case (see Tables 5 and 6). Figure 21 shows the projection of the 3D model using the pose estimated by algorithm R_and_T_mod for the high covariance case.

The estimates of the robot pose returned by the two median algorithms are much better than the least-squares algorithms, but are still off by about 2.5 ft along the walkway direction for both the high and low initial covariance cases. The best pose returned by the median also classifies the two lines of the street lamp on the right side of the image as outliers (see Fig. 22). This is because in this image there are two sets of data lines that have significant "leverage" on the resultant pose: the walkway line (as mentioned earlier, lines in depth are important for these scenes) and the street lamp (which is the only nonoutlier data on the right side of the image). Since both these sets of lines are slightly incorrect, the median chooses a pose which best explains the walkway line together with the rest of the data (which is mostly 300 ft away).

In contrast to its performance for indoor frame 8, the Tuk_wts algorithm failed completely on outdoor frame 3 (Table 5). The algorithm is extremely sensitive to initial estimates for outdoor frame 3 (see Fig. 23). For algorithm Tuk_wts to converge to the right answer, the initial estimate must be within 1 ft of the correct translation. However, if one of the outlier telephone lines is removed from the data set, the algorithm will converge to the correct

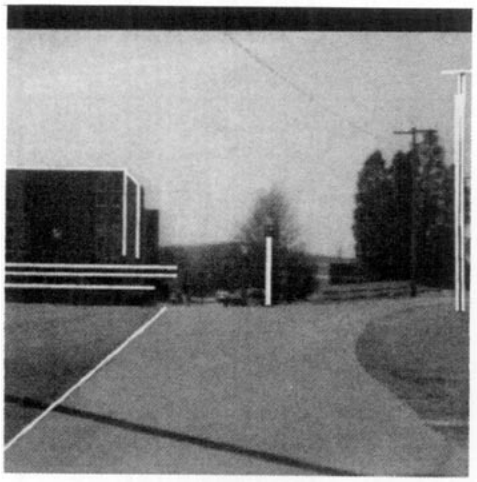

FIG. 20. Outdoor frame 3 with input image lines.

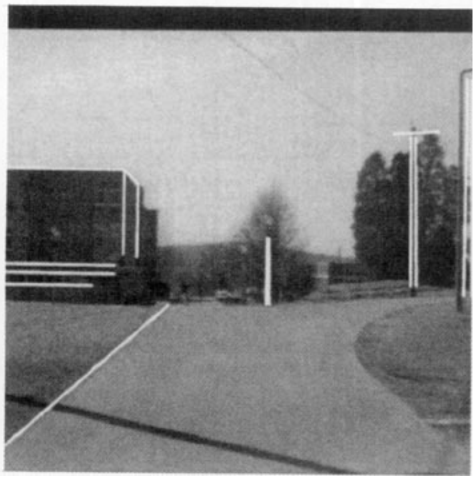

FIG. 22. Projection of model for outdoor frame 3 using final estimate of algorithm Med_R_and_T_mod with high prior covariance matrix.

answer from initial estimates up to 10 ft or so away. Finally, if two of the telephone outlier lines are removed, convergence to the correct answer is obtained from initial estimates up to 20 ft away (see Fig. 22). This problem of an accurate initial estimate is due to the presence of multiple local minima in the objective function minimized by the Tuk_Wts algorithm. A possible solution would be to use a global minimization technique for minimizing the M-estimate error functions with some prior estimate of scale [3]. The algorithm Med_R_and_T, on the other hand, is not affected by these initial estimates and produces the same answer as shown in the tables for all of them.

4.5. Discussion

The output of the pose refinement process depends on the quality of the input provided to it. The best data sets have many good lines in all directions and lines running both near and far from the camera. In contrast, indoor frame 7 and outdoor frame 3 are impoverished data sets where most of the input lines are at approximately the same distance from the camera and one or two lines are much closer to the camera. As a result these close lines have a significantly larger effect in determining the final pose estimation and have "high leverage" in the final estimation. Note that the high leverage data lines can be detected by examining the diagonal values in the hat matrix when there are no outliers.

A consensus-based algorithm tries to find the best pose which explains a significant proportion of the set of lines. Given that the observations for the nonoutlier data are

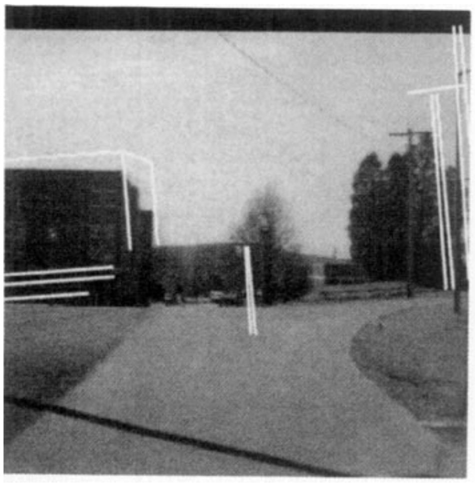

FIG. 21. Projection of model for outdoor frame 3 using final estimate of algorithm R_and_T_mod with high prior covariance matrix.

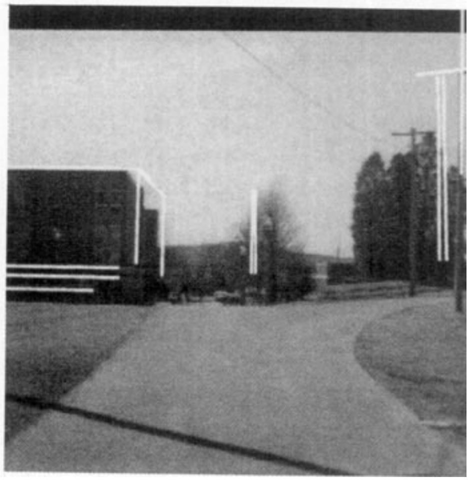

FIG. 23. Projection of model for outdoor frame 3 using final estimate of algorithm Tuk_wts.

noisy, it is conceivable that the pose returned by the consensus algorithm explains a significant set of observations with "low leverage" quite well and makes a non-outlier observation with high leverage an outlier. This is what happened in the indoor frame 8 and outdoor frame 3 case, where some of the lines with high leverage become outliers. A consequence (and an inherent danger) of the incorrect removal of the high leverage lines as outliers from an impoverished data set is that the output covariance matrix of the computed pose parameters will be much higher than what is optimally obtainable for that data set. The consensus-based algorithms will work best when there are no observations with high leverage and the data set is well rounded with a sufficient number of input data lines in all directions both close and far to the camera.

To conclude, we quote from Li [17]: "No one robust regression technique has proved superior to all others in all situations, partly because of the challenge of handling many forms of influential observations." We have observed this especially with the performance of the median and other similar consensus-based algorithms using many different error functions. For instance, in another consensus algorithm we experimented with, instead of minimizing the median, the optimization criterion was to find that pose which fits the maximum number of lines under a certain error value. The results of this algorithm were of a similar nature to the median-based algorithm. There appears to be no one algorithm which works best for all data sets. The robust statistical algorithms presented in this paper, given data with outliers, can locate the robot in the indoor scenes in a range of 0.4 ft (on the average) for the high prior covariance cases and within 0.3 ft (on the average) for the low prior covariance cases. This was sufficient for most of our robot navigation tasks.

5. SENSITIVITY ANALYSIS OF POSE ESTIMATION

The standard model adopted for imaging 3D scenes by CCD and other cameras is perspective projection. A ray from the camera focal point to a 3D point intersects the image plane at the image location of the 3D point under perspective projection. The optical axis is defined as the perpendicular line from the focal point to the imaging plane and the image center is defined as the point where the optical axis pierces the image plane. Two important camera parameters which often need to be calibrated are the focal length and the image center. In this section, we study the effect of errors in estimates of the image center and focal length on pose determination. The pose determination algorithms used in the experiments are described in Section 3. The conclusions drawn, however, are independent of the particular algorithm used.

The image center is often assumed to lie at the center of the image frame. This default center has been reported to be off by as much as 30 pixels for some standard camera and frame grabber combinations [16]. Calibration techniques using either lasers or high precision calibration plates have been used to locate the center to within a few pixels [5, 16, 26]. Is this precise calibration necessary? The analysis presented here shows that it depends on three factors:

1. The particular 3D output or inference one is interested in,
2. The level of accuracy desired in the results,
3. The amount of noise in the input data.

The goal in pose determination is to find the rotation and translation matrices which map the world coordinate system to the camera coordinate system. Given the rotation (or orientation) and translation, the location of the camera with respect to the world coordinate system can be computed. We will show that for *small field-of-view* imaging systems, an error in the estimation of the camera center does not affect the location of the camera significantly. The rotation or orientation is affected, however, and the amount of error in the orientation is linearly related to the error in the estimate of the center.

Finally, in the last subsection of this section, the effect of incorrect estimation of the focal length on the pose determination problem is studied. We show that incorrect estimates of the focal length only significantly affects the z component (i.e., parallel to the optical axis) of the translation. The x and y components of the translation and the rotation are not affected significantly. However, the location of the camera in world coordinates will be affected since the z component of the translation changes. Again, experimental results on real data are presented to support the theoretical claims.

5.1. Errors in the Pose Determination Problem from Center Offsets

The question asked is this: given two input data sets to the pose refinement problem (the first with the correct image center and the second with an offset image center), how are the two resulting poses related? The only difference between the two data sets is a constant offset of all the image pixels in one data set by the amount the center estimate is offset. Associated with each of the input data sets is a camera coordinate frame. The result of the pose refinement process is to determine the rigid body transformation between the world coordinate frame and the camera coordinate frame. Let "W" represent the world coordinate frame, "C1" the camera coordinate frame with the correct center, and "O1" the camera coordinate frame with the offset center; then

$$X_{c1} = R_{c1}(X_w) + T_{c1}. \tag{49}$$

In this equation, the rotation R_{c1} and translation T_{c1} relate a 3D point X_{c1} in the first camera coordinate frame C1 to its coordinates X_w in the world coordinate frame "W". Points in the camera coordinate frame O1 are related to points in the world coordinate frame "W" by equation

$$X_{O1} = R_{O1}(X_w) + T_{O1}. \qquad (50)$$

We will find the relationship between the two camera coordinate frames "C1" and "O1". As noted earlier the only difference between the image data associated with the two frames is a constant shift of all the pixels. Let these be ΔC_x and ΔC_y in the X and Y image frame directions, respectively; these shifts correspond to the offset of the image center for the second image data set relative to the first image. The displacement of image points between two frames due to rigid motion is given by the equation

$$\alpha = \frac{x_1 y \Omega_x}{f} - \left(f + \frac{x x_1}{f}\right) \Omega_y + y\Omega_z + \frac{(fT_x - xT_z)}{Z} \qquad (51)$$

$$\beta = \left(f + \frac{y y_1}{f}\right) \Omega_x - \frac{y_1 x \Omega_y}{f} - x\Omega_z + \frac{(fT_y - yT_z)}{Z} \qquad (52)$$

where

α and β are the image displacements in the x and y axes, respectively,

$(\Omega_x, \Omega_y, \Omega_z)$ are the small angle approximation to rotation about the X, Y, and Z axes, respectively,

(T_x, T_y, T_z) is the translation along the (X, Y, Z) axis respectively,

Z is the depth of the point in the first coordinate frame,

f is the focal length of the camera in pixels.

(x, y) is the location of the point in the first image frame (C1) and (x_1, y_1) is the location of the point in the second image frame (O1).

Between the two frames C1 and O1, $\alpha = \Delta C_x$ and $\beta = \Delta C_y$, i.e., both are constant for all points in the image. What transformation can account for this constant shift? If we assume the field of view of the camera is small, then second-order terms such as xx_1 and $x_1 y$ can be neglected. If the scene being imaged is not a frontal plane, i.e., Z is not constant for all points, then the only transformations that can cause a constant change for a general set of points are the rotations Ω_x and Ω_y about the X and Y axes; everything else (i.e., Ω_z, T_x, T_y, and T_x) will be zero. The following two equations express this relationship:

$$\alpha = \Delta C_x = -f\Omega_y \qquad (53)$$

$$\beta = \Delta C_y = f\Omega_x \qquad (54)$$

Let the rotation operator Δ_R represent the overall rotation composed of the rotations Ω_x and Ω_y about the X and Y axes. The two coordinate frames C1 and O1 are therefore hypothesized to be related by a rotation Δ_R:

$$X_{o1} = \Delta_R(X_{c1}). \qquad (55)$$

Combining Eq. (55) with Eq. (49) we get

$$X_{o1} = \Delta_R R_{c1}(X_w) + \Delta_R(T_{c1}). \qquad (56)$$

Comparing Eq. (56) with Eq. (50) we see that

$$R_{o1} = \Delta_R R_{c1} \text{ and } T_{o1} = \Delta_R(T_{c1}). \qquad (57)$$

The above equations reflect how the orientation R_{o1} and location of the world origin in camera coordinates T_{o1} are altered with incorrect knowledge of the center. The location of the camera origin in world coordinates T_w is given by the equation

$$T_{wc1} = -R^{T}_{c1}(T_{c1}) \text{ for camera frame C1.} \qquad (58)$$

$$T_{wo1} = -R^{T}_{o1}(T_{o1}) \text{ for camera frame O1.} \qquad (59)$$

Using Eq. (57) and the above equation for T_{wo1} we get

$$T_{wo1} = -R^{T}_{c1}\Delta^{T}_{R}\Delta_R(T_{c1}) = -R^{T}_{1}(T_{c1}) = T_{wc1} \qquad (60)$$

Therefore an error in estimating the image center significantly affects the location of the camera in world coordinates only if the second order terms in the motion displacement Eqs. (51) and (52) are large. For small field-of-view imaging systems, the location of the camera is not affected since the second-order terms are negligibly small.

The orientation of the robot, on the other hand, is affected; the amount depends on the values of $(\Delta C_x, \Delta C_y)$. For instance, for a camera with field of view 24° and a 512×512 image, a 30-pixel offset in the camera center in either x or y coordinate would cause a rotation error of 1.427° about the corresponding axis (using equations (54,54)). Whether changes in orientation of this order are significant or not depends on the application.

Finally, in the case of frontal planes, the depth value Z is the same for all points. Therefore, in the motion displacement equations (51) and (52) both the translation components T_x and T_y and rotation terms Ω_x and Ω_y can account for the constant displacement. In this case, the model of change in pose as given in Eq. (55) may not be correct. However, the reader is reminded that frontal planes are typically a degenerate case for pose. Even if we have a correct estimate of center, since Z is constant, there could be an incorrect pose related to the correct pose by a transformation composed of translation components T_x and T_y and rotation components Ω_x and Ω_y. The image transformations caused by rotation (Ω_x, Ω_y) can be canceled by the transformation due to translation (T_x, T_y) in Eqs. (51) and (52) leading to approximately zero values of α and β and therefore more than one pose can explain the same input data. The same observation has been made for the structure from motion problem by other researchers [22]. The above model will also break down for large field-of-view imaging systems (e.g., beyond 45° field of view), when the second-order effects cannot be ignored.

5.2. Experimental Results

In Section 3, we described algorithms for pose estimation given correspondences for 3D model and 2D image points and lines. We show results from our pose algorithms for two image sets with different errors in the location of the image center. The images (512 × 484 pixels) were acquired using a SONY B/W camera (model AVC-D1) interfaced to a Gould frame grabber. The field of view of the imaging system is approximately 24.0°. For each set of image data, a new data set was created by adding a constant pixel offset to the x and y coordinates of the image data of the original set.

The first image (Fig. 24) is of a hallway. The large door in the image is 40 ft distant from the camera. Figure 24 shows the first set of input image lines to the pose algorithm. Two more sets of input data were created by adding center offsets of (10,10) and (20,20) pixels, respectively, to the assumed center. The results for location of the camera in world coordinates for the three different center offsets is given in Table 7 under the heading "Hallway image." *The final location (in feet) in world coordinates (for scenes and images as shown in the figure above) changes only by a few tenths of an inch.*

TABLE 7

Location of Camera in World Coordinates as Computed by the Pose Refinement Algorithm for Two Sets of Real Image Data with Different Center Offsets

Center offset X	Center offset Y	Location in world L_x	L_y	L_z
		Hallway image		
(pixels)	(pixels)	(ft)	(ft)	(ft)
Measured location		40.00	4.00	3.57
0	0	39.98	4.09	3.57
10	10	40.00	4.09	3.58
20	20	40.02	4.09	3.58
		Box image		
(pixels)	(pixels)	(mm)	(mm)	(mm)
0	0	418.23	260.52	381.37
10	0	417.94	260.49	381.72
10	10	417.27	260.56	380.85
20	20	416.68	260.71	380.51

The second image, shown in Fig. 25, is from the Umass box sequence. The box is about 600 mm distant from the camera. The 15 points marked by crosses in Fig. 25 were provided as input to the pose refinement algorithm. Three new image data sets were created by adding center offsets of (10,0), (10,10), and (20,20), respectively. The results of locating the camera for these different data sets are shown in Table 7 under the heading "Box image." As can be seen from the table, the location of the camera changes by only 1 or 2 mm for different center offsets. Although results from only two images are presented here, the above behavior has been observed for numerous other images.

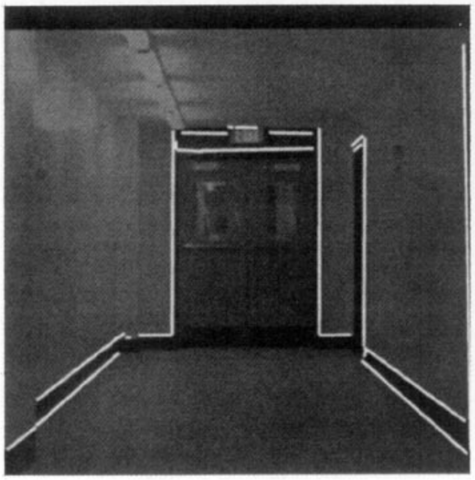

FIG. 24. Hallway image with input 2D-image lines.

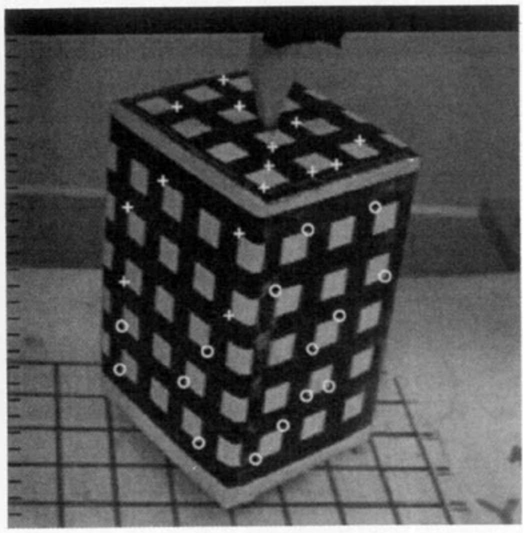

FIG. 25. Box image.

5.3. Inaccurate Estimates of the Focal Length

The focal length of the lens supplied by lens manufacturers is generally quite accurate. However, when the lens is focused on points close to the camera (i.e., when the camera is not focused at infinity) the *effective* focal length of the system must be established by a calibration procedure [26]. In this section, the effects of incorrect estimates of the focal length on the output of the pose refinement process are examined.

The image projection (x, y) of a world point X_w given an estimate of translation T_c and rotation R_c is

$$x = f\frac{(R_c(X_w) + T_c)_x}{(R_c(X_w) + T_c)_z} \tag{61}$$

$$y = f\frac{(R_c(X_w) + T_c)_y}{(R_c(X_w) + T_c)_z}. \tag{62}$$

Dividing these two equations, we obtain

$$\frac{x}{y} = \frac{(R_c(X_w) + T_c)_x}{(R_c(X_w) + T_c)_y}. \tag{63}$$

The rotation operator can be represented as a (3 × 3) matrix,

$$R = \begin{bmatrix} s_1 \\ s_2 \\ s_3 \end{bmatrix}, \tag{64}$$

where s_i, $i = 1, 2, 3$ are the vectors corresponding to the rows of the rotation matrix R_c. Substituting (64) into (63), Eq. (63) can be rewritten as

$$\frac{x}{y} = \frac{(s_1 \cdot X_w + T_{cx})}{(s_2 \cdot X_w + T_{cy})}. \tag{65}$$

This is a linear equation in the pose parameters s_1, s_2, T_{cx}, and T_{cy} which can be rewritten as

$$x(s_2 \cdot X_w + T_{cy}) - y(s_1 \cdot X_w + T_{cx}) = 0. \tag{66}$$

One such equation is obtained for each world/image point correspondence. Given five or more point correspondences, we can therefore solve the system of equations and get estimates of the parameters s_1, s_2, T_{cx}, and T_{cy}. The rotation parameters s_1, s_2, however, have quadratic constraints and therefore the system of equations must be solved by nonlinear techniques. Tsai [26] uses the same system of equations in his camera calibration algorithm. Since the rotation matrix is an orthonormal matrix, estimates of its first two rows s_1 and s_2 can be used to obtain the third row s_3, using the symmetric and other orthonormal properties of the matrix. The only pose refinement parameter not determined by the system of equations is therefore the translation along the optical axis T_z.

Our goal in this section is to examine the effect of incorrect estimates of the focal length on the output of a pose refinement algorithm. Equations (65) and (66) do not depend on the focal length and consequently an incorrect estimate of the focal length would not affect the solution of these equations. Therefore, an incorrect estimate of focal length should affect only the T_z parameter of the pose; all other parameters should not change since their estimation does not depend on knowledge of the focal length.

In practice we may minimize other error functions to do pose refinement. Based on the above analysis we hypothesize that an incorrect estimate of the focal length would only significantly affect the T_z component of the pose parameters for other pose refinement methods as well. That is, methods where the pose is not estimated by solving the system of equations defined by (66) but by some other system of equations. This hypothesis has been supported by experiments using both synthetic and real data and the pose refinement algorithms described in Section 3. Results of some of these experiments are shown in Table 8.

The experiments were performed using the synthetic, hallway and box image data sets described earlier. In each case, the pose refinement algorithm was applied using both the correct focal length and incorrect estimates of the focal length. The incorrect estimates of the focal length were obtained by multiplying the correct focal length by a scale. Thus, in Table 8, entries in rows with focal length scale 1.0 correspond to experiments with the correct focal length and entries with rows corresponding to scale not equal to 1.0 correspond to experiments with incorrect focal lengths. Both the translation and rotation results of the pose are shown in Table 8. The rotation is shown by its angle–axis representation. The axis vector is a unit vector. As can be seen from Table 8, the only large change in any of the pose parameters for any of the experiments is in the T_z component of the translation.

Although poses can be obtained whose projection fits the original image data fairly well in the case of incorrect estimates of the image center, this is not the case for incorrect estimates of focal length. Changing the focal length causes the projection of 3D points to be dilated or contracted by a constant amount while changing the T_z component of the translation causes the image projections to dilate or contract based on their depth from the camera. As we have seen, however, the minimum of the pose error functions, given incorrect estimates of focal length, leads only to a significant change in T_z. This property of poor fits makes it comparatively easier to calibrate

TABLE 8
Rotation and Translation as Computed by the Pose Refinement Algorithm for the Same Sets of Images with Different Focal Lengths

				Rotation			
Focal length scale	Translation			Angle (deg)	Axis		
	T_x	T_y	T_z		A_x	A_y	A_z
			Synthetic data				
1.000	4.004	−3.994	60.011	120.015	−0.577	0.577	0.577
0.928	4.013	−4.002	58.048	120.016	−0.577	0.577	0.578
			Hallway image				
1.000	4.055	−3.942	39.926	119.808	−0.576	0.574	0.582
0.928	4.023	−3.962	37.382	119.344	−0.579	0.574	0.580
1.045	4.072	−3.932	41.509	120.066	−0.575	0.574	0.583
			Box image				
1.000	−8.256	74.647	620.313	132.833	−0.178	0.952	−0.250
1.100	−8.344	74.801	684.354	132.627	−0.177	0.952	−0.249

imaging systems for focal length as compared to calibrations for the image center.

REFERENCES

1. P. J. Besl, J. B. Birch, and L. T. Watson, Robust window operators, in *Proceedings 2nd IEEE International Conference on Computer Vision, Tampa, Florida, Dec. 1989*, pp. 591–600.
2. J. R. Beveridge, R. Weiss, and E. Riseman, Optimization of 2-Dimensional model matching, in *Proceedings IEEE International Conference on Pattern Recognition, Atlantic City, NJ, June 1990.*
3. A. Blake and A. Zisserman, *Visual Reconstruction*, MIT Press, Cambridge, MA, 1987.
4. M. Dhome, M. Richetin, J. T. Lapreste, and G. Rives, Determination of the attitude of 3-D objects from a single perspective view, *IEEE Trans. Pattern Anal. Mach. Intell.* **11**(12), 1989.
5. O. D. Faugeras and G. Toscani, Camera calibration for 3D computer vision, in *Proceedings International Workshop on Machine Vision and Machine Intelligence, Tokyo, Japan, Feb. 2–5, 1987.*
6. C. Fennema, A. Hanson, E. Riseman, J. R. Beveridge, and R. Kumar, Model-directed mobile robot navigation, *IEEE Trans. Syst. Man Cybernet.* **20**(6), 1990.
7. M. A. Fischler and R. C. Bolles, Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography, *Comm. ACM* **24**, 1981, 381–395.
8. S. Ganapathy, Decomposition of transformation matrices for robot vision, in *Proceedings 1st IEEE Conference on Robotics, 1984*, pp. 130–139.
9. R. M. Haralick, C. N. Lee, K. Ottenberg, and M. Nolle, Analysis and solutions of the three point perspective pose estimation problem, in *Proceedings IEEE Conference on Computer Vision and Pattern Recognition, Lahiana, Maui, Hawaii, June 1991*, pp. 592–598.
10. R. M. Haralick, H. Joo, C. N. Lee, X. Zhuang, V. G. Vaidya, and M. B. Kim, Pose estimation from corresponding point data, *IEEE Trans. Syst. Man Cybernet.* **19**(6), 1989.
11. B. K. P. Horn, Closed-form solution of absolute orientation using unit quarternions, *J. Opt. Soc. Amer.* **4**, 1987, 629–642.
12. B. K. P. Horn, Relative orientation, *Internat. J. Comput. Vision* **4**, 1990, 59–78.
13. P. J. Huber, *Robust Statistics*, Wiley, New York, 1981.
14. D. Y. Kim, J. J. Kim, P. Meer, D. Mintz, and A. Rosenfeld, Robust computer vision: A least median of squares based approach, in *Proceedings DARPA Image Understanding Workshop, Morgan Kaufman Publishers, Palo Alto, CA, May 1989.*
15. R. Kumar, Model dependent inference of 3D information from a sequence of 2D images, Ph.D. Thesis, Computer Science Department, University of Massachusetts, Amherst, MA, Feb. 1992.
16. R. K. Lenz and R. Y. Tsai, Techniques for calibration of the scale factor and image center for high accuracy 3-D machine vision metrology, *IEEE Trans. Pattern Anal. Mach. Intell.* **10**(5), 1988, 713–719.
17. G. Li, Robust regression, in *Exploring Data Tables, Trends and Shapes* (D. C. Hoaglin, F. Mosteller, and J. W. Tukey, Eds.), pp. 281–343, Wiley, New York, 1985.
18. Y. Liu, T. S. Huang, and O. D. Faugeras, Determination of camera location from 2D to 3D line and point correspondences, in *Proceedings IEEE Conference on Computer Vision and Pattern Recognition, 1988*, pp. 82–88.
19. D. G. Lowe, *Perceptual Organization and Visual Recognition*, Kluwer Academic, Hingham, MA, *1985*.
20. D. G. Lowe, Integrated treatment of matching and measurement errors for robust model-based motion tracking, in *Proceedings IEEE 3rd International Conference on Computer Vision, Osaka, Japan, Dec. 1990.*
21. D. G. Lowe, Fitting parametrized three-dimensional models to images, *IEEE Trans. Pattern Anal. Mach. Intell.* **13**(5), May 1991.
22. R. Manmatha, R. Dutta, E. M. Riseman, and M. Snyder, Issues in extracting motion parameters and depth from approximate translational motion, in *Proceedings IEEE Workshop on Visual Motion, March 1989*, pp. 264–272.

23. A. Mitchie, S. Seida, and J. K. Aggarwal, Using constancy of distance to estimate position and displacement in space, *IEEE Trans. Pattern Anal. Mach. Intell.* **10**(4), July 1988.

24. F. Mosteller and J. W. Tukey, *Data Analysis and Regression*, Addison–Wesley, Reading, MA, 1977.

25. P. J. Rousseeuw and A. M. Leroy, *Robust Regression and Outlier Detection*, Wiley, New York, 1987.

26. R. Y. Tsai, An efficient and accurate camera calibration technique for 3D machine vision, in *Proceedings IEEE Conference on Computer Vision and Pattern Recognition, 1986*, pp. 364–374.

27. L. R. Williams and A. R. Hanson, Translating optical flow into token matches and depth from looming, in *Proceedings 2nd International Conference on Computer Vision, Tampa, Florida, 1989*, pp. 441–448.

28. P. R. Wolf, *Elements of Photogrammetry*, McGraw–Hill, New York, 1974.

29. A. D. Worrall, K. D. Baker, and G. D. Sullivan, Model based perspective inversion, *Image Vision Comput.* **7**(1), 1989, 17–23.

30. J. J. Wu, R. E. Rink, T. M. Caelli, and V. G. Gourishankar, Recovery of the 3D location and motion of a rigid object through camera image (an extended kalman filter approach), *Internat. J. Comput. Vision* **3**, 1988, 373–394.

31. V. J. Yohai, High breakdown-point and high efficiency robust estimates for regression, *Ann. Statist.* **15**(20), 1987.