Determining motion from 3D line segment matches: a comparative study

Zhengyou Zhang and Olivier D Faugeras

Motion estimation is a very important problem in dynamic scene analysis. Although it is easier to estimate motion parameters from 3D data than from 2D images, it is not trivial, since the 3D data we have are almost always corrupted by noise. A comparative study on motion estimation from 3D line segments is presented. Two representations of line segments and two representations of rotation are described. With different representations of line segments and rotation, a number of methods for motion estimation are presented, including the extended Kalman filter, a general minimization process and the singular value decomposition. These methods are compared using both synthetic and real data obtained by a trinocular stereo. It is observed that the extended Kalman filter with the rotation axis representation of rotation is preferable. Note that all methods discussed can be directly applied to 3D point data.

Keywords: Motion estimation, motion from stereo, noisy system, nonlinear system, minimization

Motion analysis is a very important research field in Computer Vision and Robotics. Its applications include mobile robot navigation, scene segmentation and world model construction[1]. Until recently, most research efforts were directed towards motion analysis of a sequence of monocular images[2,3]. With the development of passive stereo vision systems[4] and range finder systems[5], some work related to motion analysis from 3D data has recently emerged.

The motion analysis problem is usually divided into two subproblems: correspondence of primitives and estimation of motion. Typically used primitives are points, line segments and planar patches. For the correspondence problem the reader is referred to References 1; 6–9. Although it is easier to estimate motion parameters from 3D data than from 2D (monocular) images, it is not trivial since the 3D data we have are almost always corrupted by noise. A number of methods are proposed to combat the noise. To our knowledge, no work has been carried out to compare those methods. We believe that this work is important for researchers working on motion analysis to choose an appropriate approach based on efficiency, accuracy and robustness. In this paper, we present a comparative study of different methods of determining motion from correspondences of 3D line segments. For the problem of determining motion from 3D point correspondences, the reader is referred to References 7, 8, 10 and 11. Also, a method to determine motion from planar patches is presented by Faugeras and Herbert[8].

The paper is organized as follows. First, two representations for line segments are presented, and the problem we address is formulated. Second, representations of motion are described. Third, we show how to estimate motion using the extended Kalman filter, minimization techniques and a singular value decomposition technique. Finally we provide the results of our comparative study on different methods.

PROBLEM STATEMENT

In this section, we give two representations of 3D line segments and the transformation of line segments under rigid motion. Finally, we formalize the problem we should solve.

Line segment representations

3D line segments can be reconstructed from stereo[4] or extracted from range data. A common approach is to represent a 3D line segment L by its endpoints, noted as M_1 and M_2. Equivalently, L can be represented by two vectors $(\mathbf{l}, \mathbf{m})$, $\mathbf{l}$ is the non-normalized direction vector of L and $\mathbf{m}$, the midpoint of L, that is:

$$\mathbf{l} = M_2 - M_1, \quad \mathbf{m} = (M_1 + M_2)/2. \tag{1}$$

INRIA Sophia-Antipolis, 2004 route des Lucioles, F-06565 Valbonne Cedex, France

0262–8856/91/001010–10

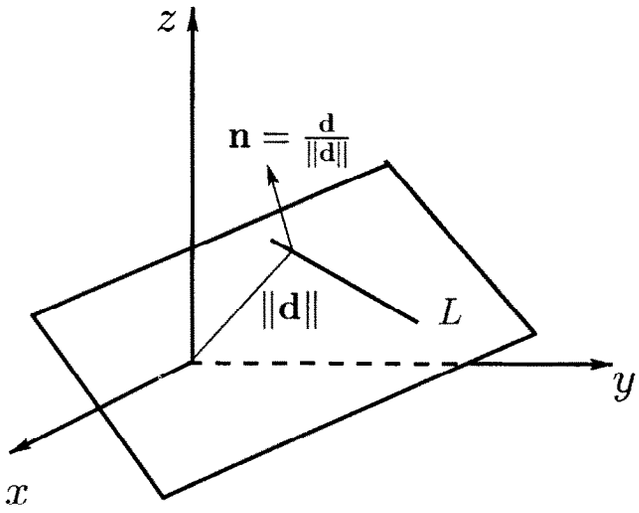

Figure 1. 3D line representation

Besides the geometric parameters, the uncertainty is also manipulated in our stereo system. Suppose that W_1 and W_2 are the covariance matrices of M_1 and M_2, respectively, and suppose that M_1 and M_2 are independent, we can compute the covariance matrix of $(\mathbf{l}, \mathbf{m})$ as follows:

$$W_L = \begin{bmatrix} W_1 + W_2 & (W_2 - W_1)/2 \\ (W_2 - W_1)/2 & (W_1 + W_2)/4 \end{bmatrix} \quad (2)$$

which is a 6×6 matrix.

We can also represent a 3D line segment by its infinite supporting line. There are a number of ways to represent a 3D line, one of which is to represent the line by two vectors $(\mathbf{u}, \mathbf{d})$, where $\mathbf{u}$ is its unit direction vector, the norm of $\mathbf{d}$ gives the distance from the origin to the line, and the direction of $\mathbf{d}$ is parallel to the normal $\mathbf{n}$ of the plane containing the line and the origin (see Figure 1). We can compute $(\mathbf{u}, \mathbf{d})$ from the above representation $(\mathbf{l}, \mathbf{m})$. Indeed, we have:

$$\mathbf{u} = \mathbf{l}/\|\mathbf{l}\| \quad \text{and} \quad \mathbf{d} = \mathbf{u} \wedge \mathbf{m} = (\mathbf{l} \wedge \mathbf{m})/\|\mathbf{l}\| \quad (3)$$

where $\wedge$ is the cross product. In reality, $\mathbf{m}$ may be any point on the line. As we see later, we use this representation to derive an analytical solution of motion.

We should point out that segments addressed in this article are *oriented*, which is obtained in stereo through the information about the intensity contrast.

3D line segment transformation

If a 3D line segment undergoes a rigid displacement $(\mathbf{R}, \mathbf{t})$ (see below) and if we use the first representation (see equation (1)), let $(\mathbf{l}, \mathbf{m})$ be the parameters before transformation and $(\mathbf{l}', \mathbf{m}')$, those after transformation, the following relations hold:

$$\mathbf{l}' = \mathbf{R}\mathbf{l} \quad \text{and} \quad \mathbf{m}' = \mathbf{R}\mathbf{m} + \mathbf{t} \quad (4)$$

As we know that a segment can be differently segmented in successive views, and that its direction is relatively more stable, we insist on having the transformed segment parallel to the segment in the second view which yields[12]:

$$\mathbf{l}' \wedge \mathbf{R}\mathbf{l} = 0 \quad \text{and} \quad \mathbf{l}' \wedge (\mathbf{m}' - \mathbf{R}\mathbf{m} - \mathbf{t}) = 0 \quad (5)$$

We shall use these two equations to compute the motion parameters. If we note:

$$\mathbf{f}(\mathbf{p}, \mathbf{x}) = \begin{bmatrix} \mathbf{l}' \wedge \mathbf{R}\mathbf{l} \\ \mathbf{l}' \wedge (\mathbf{m}' - \mathbf{R}\mathbf{m} - \mathbf{t}) \end{bmatrix} \quad (6)$$

Equation (5) becomes:

$$\mathbf{f}(\mathbf{p}, \mathbf{x}) = 0 \quad (7)$$

where $\mathbf{x} = [\mathbf{l}^T, \mathbf{m}^T, \mathbf{l}'^T, \mathbf{m}'^T]^T$ (the superscript ^{T} denotes the transpose of a vector or a matrix) is a 12 dimensional vector which we call *measurement vector*, and $\mathbf{p}$ is one of the motion parametrizations described below. In filtering terminology, $\mathbf{p}$ is called *state vector*. So our problem can be formulated as follows:

Given *n measurement vectors:* $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$,
i.e. given a noisy system:

$$\mathbf{f}(\mathbf{p}, \mathbf{x}_i) = 0, \quad \textit{for } i = 1, \ldots, n,$$

Recover *the motion parameters* $\mathbf{p}$.

MOTION REPRESENTATION

Any rigid motion can be uniquely decomposed into a rotation around an axis passing through the origin of the coordinate system, and a translation. The translation is supposed after the rotation. A translation can be simply represented by a 3D vector $\mathbf{t} = [t_x\ t_y\ t_z]^T$. A rotation can be represented by a 3×3 matrix $\mathbf{R}$ called the *rotation matrix*, which is an orthogonal matrix satisfying:

$$\mathbf{R}\mathbf{R}^T = \mathbf{I} \quad (8)$$

where $\mathbf{I}$ is the 3×3 identity matrix. This representation gives a simple way of representing a 3D rotation, but leads to a high dimensional space of constraints. Several other representations of rotation are available[13], and we present here two of them: using the rotation axis and using a quaternion.

Using the rotation axis

A rotation can be defined as a three dimensional vector $\mathbf{r} = [a\ b\ c]^T$ whose direction is that of the rotation axis and whose norm is equal to the rotation angle.

For convenience, we note $\tilde{\mathbf{v}}$ as the antisymmetric matrix defined by $\mathbf{v}$. More precisely, if $\mathbf{v} = [x\ y\ z]^T$, then

$$\tilde{\mathbf{v}} = \begin{bmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{bmatrix} \quad (9)$$

In fact, for any three dimensional vectors $\mathbf{u}$ and $\mathbf{v}$, we hve $\mathbf{u} \wedge \mathbf{v} = \tilde{\mathbf{u}}\mathbf{v}$.

The relation between $\mathbf{R}$ and $\mathbf{r}$ is the following Rodrigues formula:

$$\mathbf{R} = e^{\tilde{\mathbf{r}}} = \mathbf{I}_3 + \mathrm{f}(\theta)\tilde{\mathbf{r}} + \mathrm{g}(\theta)\tilde{\mathbf{r}}^2 \tag{10}$$

where $\theta = \sqrt{a^2 + b^2 + c^2}$ is the rotation angle, $\mathrm{f}(\theta) = \sin\theta/\theta$ and $\mathrm{g}(\theta) = (1 - \cos\theta)/\theta^2$.

Thus, the parameter vector $\mathbf{p}$ in equation (7) is in this representation a six dimensional vector, noted as $\mathbf{s}$:

$$\mathbf{s} = \begin{bmatrix} \mathbf{r} \\ \mathbf{t} \end{bmatrix} \tag{11}$$

Using a quaternion

Quaternions have been found useful in robotics and vision[14]. A quaternion $\mathbf{q}$ can be considered as being either a 4-dimensional vector $[\lambda_0\ \lambda_1\ \lambda_2\ \lambda_3]^T$ or as a pair (α, γ), where α is a real number equal to λ_0, and γ is the vector $[\lambda_1\ \lambda_2\ \lambda_3]^T$. We define the multiplication $\times$ of two quaternions $\mathbf{q}$ and $\mathbf{q}'$ as follows:

$$\mathbf{q} \times \mathbf{q}' = (\alpha\alpha' - \gamma \cdot \gamma', \alpha\gamma' + \alpha'\gamma + \gamma \wedge \gamma') \tag{12}$$

The conjugate and the magnitude of a quaternion $\mathbf{q}$ are defined as follows:

$$\bar{\mathbf{q}} = (\alpha, -\gamma), \tag{13}$$

$$|\mathbf{q}|^2 = \mathbf{q} \times \bar{\mathbf{q}} = (\alpha^2 + \|\gamma\|^2, \mathbf{0}) = (\|\mathbf{q}\|^2, \mathbf{0}) \tag{14}$$

A real number x is identified with the quaternion $(x, \mathbf{0})$ and a 3D vector $\mathbf{v}$ is identified with the quaternion $(0, \mathbf{v})$.

A rotation can then be represented by two quaternions $\mathbf{q} = (\alpha, \gamma)$ and $-\mathbf{q}$, with $|\mathbf{q}| = 1$. The relation between this representation and the rotation axis one is:

$$\alpha = \cos(\theta/2) \quad \text{and} \quad \gamma = \sin(\theta/2)\mathbf{u} \tag{15}$$

where $\theta = \|\mathbf{r}\|$ and $\mathbf{u} = \mathbf{r}/\|\mathbf{r}\|$. Note that there are two quaternions for one rotation. It is not surprising since a rotation of angle θ around an axis $\mathbf{u}$ is the same as a rotation of angle $2\pi - \theta$ around the axis $-\mathbf{u}$. Usually, the rotation angle between two successive views does not go beyond π, so we can impose that the first element α of a quaternion $\mathbf{q}$ must be positive. Thus the mapping between rotation and quaternion is unique under this new constraint.

The relation between $\mathbf{R}$ and a unit quaternion $\mathbf{q} = [\lambda_0, \lambda_1, \lambda_3]^T$ is given as follows:

$$\mathbf{R} = \begin{bmatrix} \lambda_0^2 + \lambda_1^2 - \lambda_2^2 - \lambda_3^2 & 2(\lambda_1\lambda_2 - \lambda_0\lambda_3) & 2(\lambda_1\lambda_3 + \lambda_0\lambda_2) \\ 2(\lambda_1\lambda_2 + \lambda_0\lambda_3) & \lambda_0^2 - \lambda_1^2 + \lambda_2^2 - \lambda_3^2 & 2(\lambda_2\lambda_3 - \lambda_0\lambda_1) \\ 2(\lambda_1\lambda_3 - \lambda_0\lambda_2) & 2(\lambda_2\lambda_3 + \lambda_0\lambda_1) & \lambda_0^2 - \lambda_1^2 - \lambda_2^2 + \lambda_3^2 \end{bmatrix}$$

The product $\mathbf{Rv}$ can be identified as the product of quaternions:

$$\mathbf{Rv} = \mathbf{q} \times \mathbf{v} \times \bar{\mathbf{q}} \tag{16}$$

Thus, the parameter vector $\mathbf{p}$ in equation (7) is in this representation a 7-dimensional vector, noted as $\mathbf{s}_q$:

$$\mathbf{s}_q = \begin{bmatrix} \mathbf{q} \\ \mathbf{t} \end{bmatrix} \tag{17}$$

under the constraint $\|\mathbf{q}\| = 1$.

Derivation of rotation matrix

At this point, we have two parametrizations for a rotation. In this section, we give the first derivatives of the rotation matrix with respect to each of its parametrizations, which are used in the methods described below. More precisely, we are interested in computing the derivative of $\mathbf{Rv}$ with respect to $\mathbf{r}$ and that with respect to $\mathbf{q}$, where $\mathbf{v} = [v_0\ v_1\ v_2]^T$ is an arbitrary 3D vector.

The derivative of $\mathbf{Rv}$ with respect to $\mathbf{r}$ is:

$$\frac{\partial \mathbf{Rv}}{\partial \mathbf{r}} = \frac{\cos(\theta) - \mathrm{f}(\theta)}{\theta^2}(\tilde{\mathbf{r}}\tilde{\mathbf{v}})\,\mathbf{r}^T + \frac{\sin(\theta) - 2\theta\mathrm{g}(\theta)}{\theta^3}(\tilde{\mathbf{r}}(\tilde{\mathbf{r}}\mathbf{v}))\,\mathbf{r}^T - \mathrm{f}(\theta)\tilde{\mathbf{v}} + \mathrm{g}(\theta)(-(\widetilde{\mathbf{r}\mathbf{v}}) + (\mathbf{r}.\mathbf{v})\mathbf{I}_3 - \mathbf{v}\mathbf{r}^T). \tag{18}$$

where $\mathrm{f}(\theta)$ and $\mathrm{g}(\theta)$ have the same definitions as above.

We define $E(\mathbf{R}, \mathbf{v})$ to be the 3×3 matrix $\partial\mathbf{Rv}/\partial\mathbf{r}$.

The derivative of $\mathbf{Rv}$ with respect to $\mathbf{q}$ is simpler. Indeed, we have:

$$\frac{\partial \mathbf{Rv}}{\partial \mathbf{q}} = \begin{bmatrix} d_0 & d_1 & d_2 & d_3 \\ -d_3 & -d_2 & d_1 & d_0 \\ d_2 & -d_3 & -d_0 & d_1 \end{bmatrix} \tag{19}$$

where

$$\begin{aligned} d_0 &= 2(\lambda_0 v_0 - \lambda_3 v_1 + \lambda_2 v_2) \\ d_1 &= 2(\lambda_1 v_0 + \lambda_2 v_1 + \lambda_3 v_2) \\ d_2 &= 2(-\lambda_2 v_0 + \lambda_1 v_1 + \lambda_0 v_2) \\ d_3 &= 2(-\lambda_3 v_0 - \lambda_0 v_1 + \lambda_1 v_2) \end{aligned} \tag{20}$$

We define $Q(\mathbf{R}, \mathbf{v})$ to be the 3×4 matrix $\partial\mathbf{Rv}/\partial\mathbf{q}$.

The derivative of $\mathbf{f}(\mathbf{p}, \mathbf{x})$ (see equation (6)) with respect to $\mathbf{s}$ can be easily computed as follows:

$$\frac{\partial \mathbf{f}(\mathbf{p}, \mathbf{x})}{\partial \mathbf{s}} = \begin{bmatrix} \tilde{\mathbf{I}}' E(\mathbf{R}, \mathbf{I}) & 0 \\ -\mathbf{I}' E(\mathbf{R}, \mathbf{m}) & -\tilde{\mathbf{I}}' \end{bmatrix} \tag{21}$$

and the derivative of $\mathbf{f}(\mathbf{p}, \mathbf{x})$ with respect to $\mathbf{s}_q$ is:

$$\frac{\partial \mathbf{f}(\mathbf{p}, \mathbf{x})}{\partial \mathbf{s}_q} = \begin{bmatrix} \tilde{\mathbf{I}}' Q(\mathbf{R}, \mathbf{I}) & 0 \\ -\mathbf{I}' Q(\mathbf{R}, \mathbf{m}) & -\tilde{\mathbf{I}}' \end{bmatrix} \tag{22}$$

The tilde '~' above is defined as in equation (9).

ESTIMATING MOTION USING EXTENDED KALMAN FILTER

The Kalman filter is a powerful tool to deal with a linear noisy system. The reader is assumed familiar with the filter equations. (For details of this filter, the reader is referred to References 15 and 16.)

However, the Kalman filter is not directly applicable to our problem, because equation (7) is non-linear. So we use th so-called extended Kalman filter[17–19], i.e., we first linearize equation (7), then we apply the Kalman filter to the linearized system. The linearized system is as follows[1]:

$$\mathbf{p}_i = \mathbf{p}_{i-1} \tag{23}$$

$$\mathbf{y}_i = M_i \mathbf{p}_i + \mathbf{w}_i \tag{24}$$

where

$$\mathbf{y}_i = -\mathbf{f}(\hat{\mathbf{p}}_{i-1}, \hat{\mathbf{x}}_i) + \frac{\partial \mathbf{f}(\hat{\mathbf{p}}_{i-1}, \hat{\mathbf{x}}_i)}{\partial \mathbf{p}} \hat{\mathbf{p}}_{i-1}$$

$$M_i = \frac{\partial \mathbf{f}(\hat{\mathbf{p}}_{i-1}, \hat{\mathbf{x}}_i)}{\partial \mathbf{p}}$$

$$\mathbf{w}_i = \frac{\partial \mathbf{f}(\hat{\mathbf{p}}_{i-1}, \hat{\mathbf{x}}_i)}{\partial \mathbf{x}} (\mathbf{x}_i - \hat{\mathbf{x}}_i)$$

$\hat{\mathbf{p}}_{i-1}$ is the current estimate of $\mathbf{p}$ before processing the linearized system, and $\hat{\mathbf{x}}_i$ is the current measurement. The derivative of $\mathbf{f}(\mathbf{p}, \mathbf{x})$ with respect to $\mathbf{p}$ can be computed using either equation (21) or equation (22). What we need to compute yet is the derivative of $\mathbf{f}(\mathbf{p}, \mathbf{x})$ with respect to $\mathbf{x}$, which is the same for both motion parametrizations:

$$\frac{\partial \mathbf{f}(\mathbf{p}, \mathbf{x})}{\partial \mathbf{x}} = \begin{bmatrix} \tilde{\mathbf{l}}'\mathbf{R} & 0 & -(\widetilde{\mathbf{R}\mathbf{l}}) & 0 \\ 0 & -\tilde{\mathbf{l}}'\mathbf{R} & \widetilde{\mathbf{R}\mathbf{m}} + \tilde{\mathbf{t}} - \widetilde{\mathbf{m}}' & \tilde{\mathbf{l}}' \end{bmatrix} \tag{25}$$

The expectation and the covariance of the new measurement noise $\mathbf{w}_i$ are easily derived from that of $\mathbf{x}_i$ as:

$$E[\mathbf{w}_i] = 0 \quad \text{and} \quad W_i \stackrel{\text{def}}{=} E[\mathbf{w}_i \mathbf{w}_i^T] = \left[\frac{\partial \mathbf{f}(\hat{\mathbf{p}}_{i-1}, \hat{\mathbf{x}}_i)}{\partial \mathbf{x}} \right] \Lambda_i \left[\frac{\partial \mathbf{f}(\hat{\mathbf{p}}_{i-1}, \hat{\mathbf{x}}_i)}{\partial \mathbf{x}} \right]$$

where Λ_i is the covariance matrix of $\mathbf{x}_i$. If we suppose that the matched segments L_i and L_i' are independent, then:

$$\Lambda_i = \begin{bmatrix} W_{L_i} & 0 \\ 0 & W_{L_i'} \end{bmatrix}$$

An important remark is that when we use the quaternion to represent the rotation, we have added the constraint $\|\mathbf{q}\| = 1$, i.e. $\mathbf{q}^T\mathbf{q} = 1$ in equation (7) as an additional measurement.

ESTIMATING MOTION USING MINIMIZATION

We can restate the motion estimation problem as a minimization problem, i.e.:

Given *n measurement vectors*: $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$,
Recover *the motion parameters* $\mathbf{p}$ *so that*

$$\sum_{i=1}^{n} [\mathbf{f}(\mathbf{p}, \mathbf{x}_i)^T \mathbf{f}(\mathbf{p}, \mathbf{x}_i)]$$

is minimized.

Here $\Sigma_{i=1}^{n}[\mathbf{f}(\mathbf{p}, \mathbf{x}_i)^T \mathbf{f}(\mathbf{p}, \mathbf{x}_i)]$ is called the *objective function* in the minimization problem which is denoted by $\mathcal{F}(\mathbf{p})$, i.e.:

$$\mathcal{F}(\mathbf{p}) = \Sigma_{i=1}^{n}[(\mathbf{l}_i' \wedge \mathbf{R}\mathbf{l}_i)^T (\mathbf{l}_i' \wedge \mathbf{R}\mathbf{l}_i) + (\mathbf{l}_i' \wedge (\mathbf{m}_i' - \mathbf{R}\mathbf{m}_i - \mathbf{t}))^T (\mathbf{l}_i' \wedge (\mathbf{m}_i' - \mathbf{R}\mathbf{m}_i - \mathbf{t}))] \tag{26}$$

There exist many routines in mathematical libraries (like Nag[20]) to minimize $\mathcal{F}(\mathbf{p})$ with or without constraints. The minimization process can be speeded up if we also supply the first derivations, which can be easily obtained using equation (21) or (22).

We can adopt the weighted least-squares method to take into account the uncertainty of measurements. That is, we first use the general minimization algorithms to obtain a better estimate of motion by minimizing the simple criterion (equation (26)). Then we compute the covariance of $\mathbf{f}(\mathbf{p}, \mathbf{x}_i)$ using the first order approximation, denoted by W_i. We again use the minimization algorithm to obtain a new estimate by minimizing:

$$\sum_{i=1}^{n} [\mathbf{f}(\mathbf{p}, \mathbf{x}_i)^T \mathbf{W}_i^{-1} \mathbf{f}(\mathbf{p}, \mathbf{x}_i)] \tag{27}$$

The new estimate is generally better than the previous one, but more time is needed to compute the weighting. We have not implemented the weighted least-squares technique using the general minimization algorithms. However, we note that the EKF presented in the previous section can be considered as a recursive implementation of the weighted least-squares technique.

ANALYTICAL SOLUTION: SVD

In this section, we present an analytical method to recover the motion parameters. The second representation of line segments (see equation (3)) and the quaternion representation of rotation are used. Faugeras and Hébert[8] have already proposed an analytical method from point and plane correspondences. The method described here is directly inspired from theirs.

The relation between a line segment $(\mathbf{u}, \mathbf{d})$ and the transformed line segment $(\mathbf{u}', \mathbf{d}')$ is:

$$\mathbf{u}' = \mathbf{R}\mathbf{u} \tag{28}$$

$$\mathbf{d}' = \mathbf{R}\mathbf{d} + \mathbf{u}' \wedge \mathbf{t} \tag{29}$$

The first one is evident, while the second one can be easily verified using the definition of $\mathbf{d}$:

$$\mathbf{d}' \triangleq \mathbf{u}' \wedge \mathbf{m}' = \mathbf{u}' \wedge (\mathbf{R}\mathbf{m} + \mathbf{t}) = \mathbf{R}\mathbf{u} \wedge \mathbf{R}\mathbf{m} + \mathbf{u}' \wedge \mathbf{t}$$
$$= \mathbf{R}(\mathbf{u} \wedge \mathbf{m}) + \mathbf{u}' \wedge \mathbf{t} = \mathbf{R}\mathbf{d} + \mathbf{u}' \wedge \mathbf{t} \tag{30}$$

Due to the fact that the orientation of a segment is more conservative than other parameters (for example, $\mathbf{d}$ or $\mathbf{m}$), we divide the motion determination problem into two subproblems:

1. Determine first the rotation using equation (28)

under the following criterion:

$$\mathbf{Min} \sum_{i=1}^{n} \|\mathbf{u}'_i - \mathbf{R}\mathbf{u}_i\|^2 \tag{31}$$

2. Determine then the translation using equation (29) under the following criterion:

$$\mathbf{Min} \sum_{i=1}^{n} \|\mathbf{d}'_i - \mathbf{R}^*\mathbf{d}_i - \mathbf{u}'_i \wedge \mathbf{t}\|^2 \tag{32}$$

where $\mathbf{R}^*$ is the rotation matrix recovered in the first step.

Determining rotation

By using equation (16), we can restate the minimization problem of equation (31) in quaternion notation as:

$$\mathbf{Min} \sum_{i=1}^{n} |\mathbf{u}'_i - \mathbf{q} \times \mathbf{u}_i \times \bar{\mathbf{q}}|^2 \tag{33}$$

subject to the constraint $|\mathbf{q}| = 1$. Here $\times$ is the quaternion multiplication. By multiplying each term in equation (33) with $|\mathbf{q}|^2$, we get

$$\mathbf{Min} \sum_{i=1}^{n} |\mathbf{u}'_i \times \mathbf{q} - \mathbf{q} \times \mathbf{u}_i|^2 \tag{34}$$

From the definition of the product of two quaternions, we can express $\mathbf{u}'_i \times \mathbf{q} - \mathbf{q} \times \mathbf{u}_i$ as a linear function of $\mathbf{q}$. Indeed, there exists a matrix A_i such that:

$$\mathbf{u}'_i \times \mathbf{q} - \mathbf{q} \times \mathbf{u}_i = A_i \mathbf{q} \tag{35}$$

where

$$A_i = \begin{bmatrix} 0 & (\mathbf{u}_i - \mathbf{u}'_i)^T \\ -(\mathbf{u}_i - \mathbf{u}'_i) & (\tilde{\mathbf{u}}_i + \tilde{\mathbf{u}}'_i) \end{bmatrix}$$

Then, equation (33) can be further restated as:

$$\mathbf{Min} \sum_{i=1}^{n} \mathbf{q}^T A_i^T A_i \mathbf{q} = \mathbf{Min}\, \mathbf{q}^T A \mathbf{q} \tag{36}$$

where $A = \Sigma_{i=1}^{n} A_i^T A_i$ and $\|\mathbf{q}\| = 1$. The matrix A can be computed incrementally.

Since A is a symmetric matrix, the solution to this problem is the four dimensional vector $\mathbf{q}_{\min}$ corresponding to the smallest eigenvalue of A.

Determining translation

We can determine translation using the standard minimization technique. Let the derivative of equation (32) with respect to $\mathbf{t}$ be zero, we have:

$$\sum_{i=1}^{n} 2(\mathbf{d}'_i - \mathbf{R}^*\mathbf{d}_i - \tilde{\mathbf{u}}'_i\mathbf{t})^T \tilde{\mathbf{u}}'_i = 0 \tag{37}$$

that is

$$\left(\sum_{i=1}^{n} \tilde{\mathbf{u}}'_i (\tilde{\mathbf{u}}'_i)^T\right) \mathbf{t} = \sum_{i=1}^{n} \tilde{\mathbf{u}}'^T_i (\mathbf{d}'_i - \mathbf{R}^*\mathbf{d}_i) \tag{38}$$

Table 1. Abbreviations of different methods

Abbrev.	*Rep. of rot.*	*Approach*
EKF-AXIS	Rotation axis	Extended Kalman filter
EKF-QUAT	Quaternion	Extended Kalman filter
MIN-AXIS	Rotation axis	Gauss-Newton minimiz.
MIN-QUAT	Quaternion	Seq. quad. programming
EIGEN	Quaternion	Singular value decomp.

If $(\Sigma_{i=1}^{n} \mathbf{u}'_i (\mathbf{u}'_i)^T)$ is a full rank matrix, we have explicitly the translation vector:

$$\mathbf{t} = \left(\sum_{i=1}^{n} \tilde{\mathbf{u}}'_i (\tilde{\mathbf{u}}'_i)^T\right)^{-1} \left(\sum_{i=1}^{n} \tilde{\mathbf{u}}'^T_i (\mathbf{d}'_i - \mathbf{R}^*\mathbf{d}_i)\right) \tag{39}$$

If not, $\mathbf{t}$ is unrecoverable. It can be shown that $(\Sigma_{i=1}^{n} \tilde{\mathbf{u}}'_i (\tilde{\mathbf{u}}'_i)^T)$ is always of full rank if two of $\mathbf{u}'_i (i = 1 \ldots n)$ are different (non-parallel).

EXPERIMENTAL RESULTS

The objective of our comparative study is to investigate the applicability of different methods to stereo data. Both synthetic and real stereo data are used to compare the methods described in the preceding sections. For simplicity, we use some abbreviations to refer to different methods (see Table 1). Note that we have not implemented the weighted least-squares method in using the general minimization techniques.

Results with synthetic data

The synthetic data we used contain 26 segments. One of the endpoints of each segment is fixed as the centre of a sphere with radius 100 units. The other endpoints are on the surface of the sphere. We choose them so that the sphere surface is quasi-uniformity sample by them. In other words, the orientation of segments are uniformly distributed in the space. Thus we obtain a set of noise free 3D line segments in one position. Then we apply a motion, which equals $[0.4, 0.2, 0.5, 200.0, -150.0, 300.0]^T$ under the rotation axis representation, to this set and we obtain another set. Finally, independent Gaussian noise with mean zero and standard deviations σ_x, σ_y and σ_z is added to the 3D coordinates of each endpoint in x, y and z directions to obtain the noisy measurements.

Before going through the experimental results, it is useful to examine the shape of the objective function of the minimization (equation (26)). The noise level of the measurements is: $\sigma_x = \sigma_y = 2$, $\sigma_z = 6$. Figure 2 displays the objective function for two matches of noisy line segments ($n = 2$ in equation (26)). In Figure 2a, the objective function is plotted with respect to the x and z components of the rotation vector while fixing the translation at $[200.0, -150.0, 300.0]^T$ and the y component of the rotation vector at 0.2. The x and z components, r_x and r_z, vary around their real values, i.e. $-1.6 \leqslant r_x \leqslant 2.4$ and $-1.5 \leqslant r_z \leqslant 2.5$. In Figure 2b, the objective function is plotted with respect to the x and z components of the translation vector while fixing the rotation at $[0.4, 0.2, 0.5]^T$ and the y

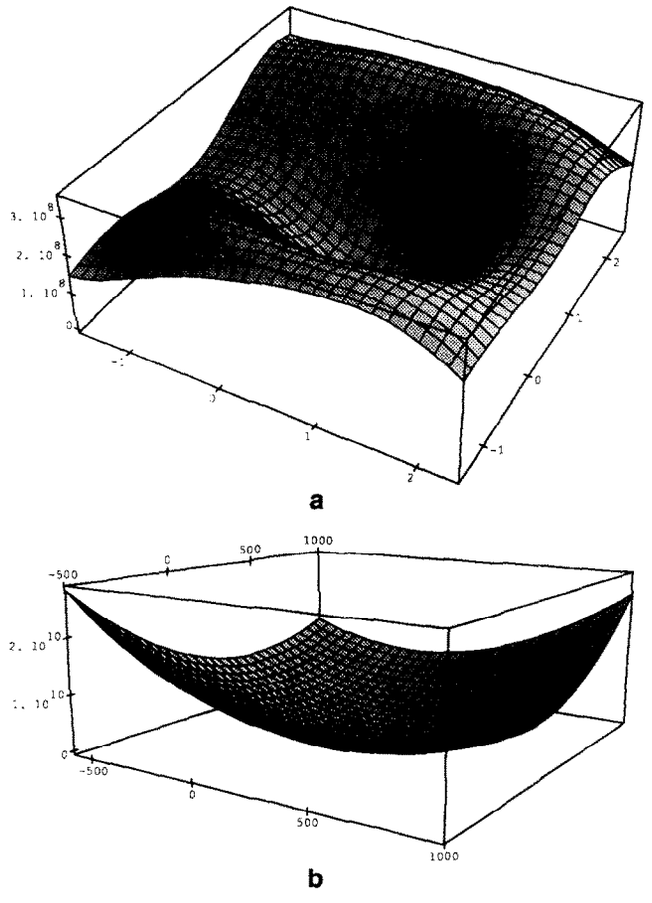

Figure 2. Objective function for two matches of noisy line segments

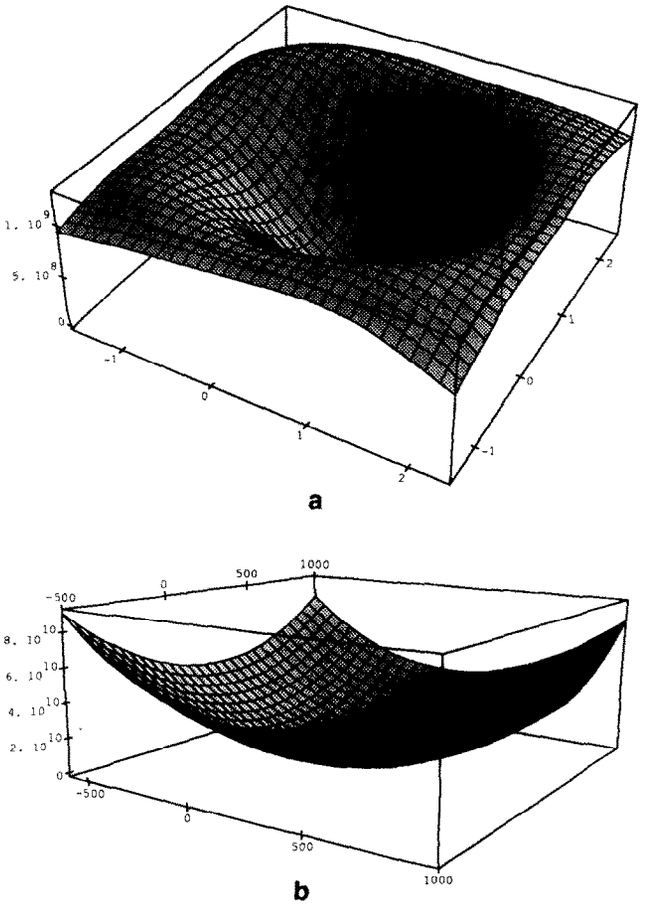

Figure 3. Objective function for ten matches of noisy line segments

component of the translation vector at -150.0. The x and z components, t_x and t_z, vary around their real values, i.e. $-600.0 \leq t_x \leq 1000.0$ and $-500.0 \leq t_z \leq 1000.0$. Figure 3 shows the same thing as Figure 2, except the objective function is computed using ten matches ($n = 10$).

From those drawings, we see that the objective function with respect to the rotational parameters is smooth, and that it has only one local minimum in a rather wide range near the true value. This implies that in an iterative minimization procedure we can find the good solution if we start with an initial guess not *very* far from the true value, as we can observe later. If the initial guess differs from the real value about 0.6 radians in each direction (about 60 degrees of difference in rotation angle), the iterative minimization procedure can be expected to converge to the good solution. The objective function with respect to the translational parameters is also smooth, and it has only one minimum. This implies that we can always find the good solution even if the initial guess in translation is very bad. The difference between the objective functions plotted in Figures 2 and 3 is that the later is more symmetric in different directions than the former. This is to say that the effects of Gaussian errors in measurements can be reduced by using more measurements, and we can expect that more data used, better estimation can be obtained.

In the following, motion error is given in two parts: rotation error and translation error. If we use the rotation axis representation, let $\mathbf{r}$ be the real rotation parameter ($[0.4, 0.2, 0.5]^T$ in our case) and $\hat{\mathbf{r}}$ be the estimated rotation parameter, then the rotation error is defined as:

$$e_r = \|\mathbf{r} - \hat{\mathbf{r}}\| / \|\mathbf{r}\| \times 100\% \tag{40}$$

If we use the quaternion representation, we first transform it into the rotation axis representation, and then compute the error in the same way. Similarly, the translation error is defined as:

$$e_t = \|\mathbf{t} - \hat{\mathbf{t}}\| / \|\mathbf{t}\| \times 100\% \tag{41}$$

where $\mathbf{t}$ is the real translation parameter ($[200.0, -150.0, 300.0]^T$ in our case) and $\hat{\mathbf{t}}$ is the estimated one.

Since the system is nonlinear, recursive methods may give different solutions (even a wrong solution) with different initial estimates. Figures 4 and 5 show the motion errors of the four recursive methods with respect to different initial estimates. The noise level of the measurements is: $\sigma_x = \sigma_y = 2$, $\sigma_z = 6$. In **EKF-AXIS** and **EKF-QUAT**, five iterations of **EKF** are applied. In **MIN-AXIS** and **MIN-QUAT**, usually more than 30 iterations are needed to get a stable solution. The abscissa coordinates (from -25 to 25) correspond to different initial estimates. More precisely, the abscissa i corresponds to the initial estimate:

$$[0.4\ 0.2\ 0.5\ 200.0\ -150.0\ 300.0]^T + i[0.05\ 0.05\ 0.05\ 15.00\ 15.00\ 15.00]^T$$

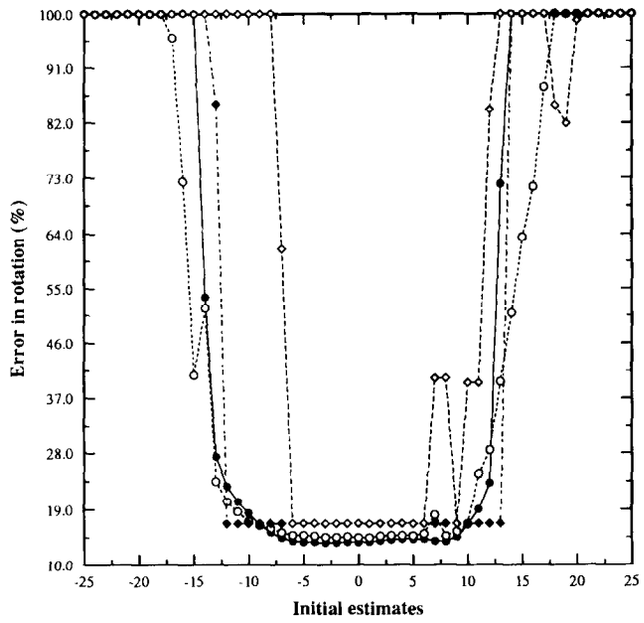

Figure 4. Comparison between EKF-AXIS, EKF-QUAT, MIN-AXIS and MIN-QUAT: error in rotation versus different initial estimates. ●: EKF-AXIS; ○: EKF-QUAT; ◆: MIN-AXIS; ◇: MIN-QUAT

For example, the abscissa -5 corresponds to the initial estimate $[0.15, -0.05, 0.25, 125.0, -225.0, 225.0]^T$. The errors are the average of 20 tries. From all these results, we can say that these methods (except **MIN-QUAT**) converge in a rather wide range (from -12 to 12 for EKF methods, i.e. from $[-0.2, -0.4, -0.1, 20, -330, 120]^T$ to $[1.0, 0.8, 1.1, 380, 30, 480]^T$). We find also that the methods using the quaternion representation (especially **MIN-QUAT**) is less stable than the methods using the rotation axis representation. The instability of the **EKF-QUAT** method can be observed at $i = 7$, but it is not as serious as that of **MIN-QUAT**. We observe also this instability of the quaternion representation in other experiments[1]. Another remark is that if **MIN-AXIS** and **MIN-QUAT** do not diverge, they give exactly the same solution.

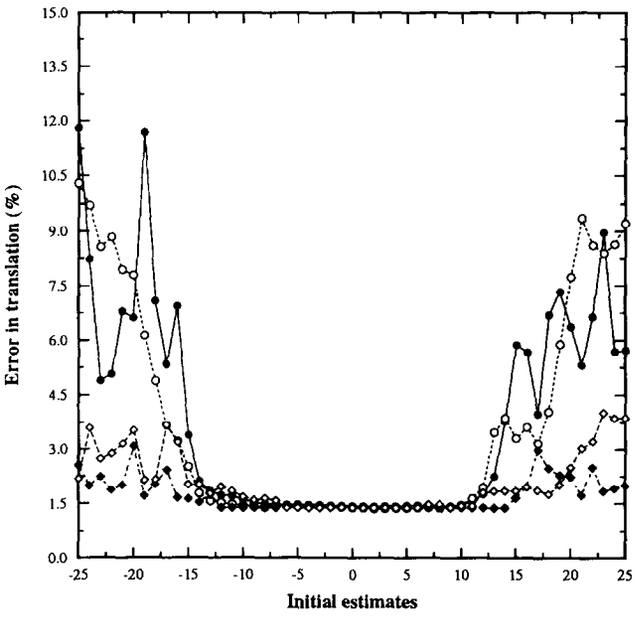

Figure 5. Comparison between EKF-AXIS, EKF-QUAT, MIN-AXIS and MIN-QUAT: error in translation versus different initial estimates. ●: EKF-AXIS; ○: EKF-QUAT; ◆: MIN-AXIS; ◇: MIN-QUAT

Table 2. Comparison of different methods: user time and motion error

Methods	*User time (second)*	*Rot. error (%)*	*Trans. error (%)*
MIN-QUAT	122.8	17.73	1.17
MIN-AXIS	57.7	17.73	1.17
EKF-QUAT	29.7	14.91	1.15
EKF-AXIS	27.7	14.26	1.16
EIGEN	0.07	20.72	1.12

Table 2 shows the comparison on run time (on SUN 3/60 workstation), rotation error and translation error *versus* different methods. The results are the average of ten tries. Two line segment correspondences are used. The EKF is iterated five times. $\sigma_z = \sigma_y = 2$, $\sigma_z = 6$. From Table 2, we observe that using a general minimization routine is very time expensive (for example, 122.8 seconds for **MIN-QUAT**. This is because more than 30 iterations are needed to get a stable solution) and that the **EIGEN** method is very efficient. EKF gives smaller motion errors than other methods with a reasonable run time. This is expected, since EKF takes into account the different uncertainty distribution of measurements and the others treat equally each measurement and each component of a measurement. Another remark is that using the quaternion representation is more time consuming than using the rotation axis representation. This is for two reasons:

1. The quaternion representation has one parameter more than the rotation axis representation.
2. There is a constraint for the quaternion representation, and in **EKF-QUAT** we add this constraint as an additional measurement.

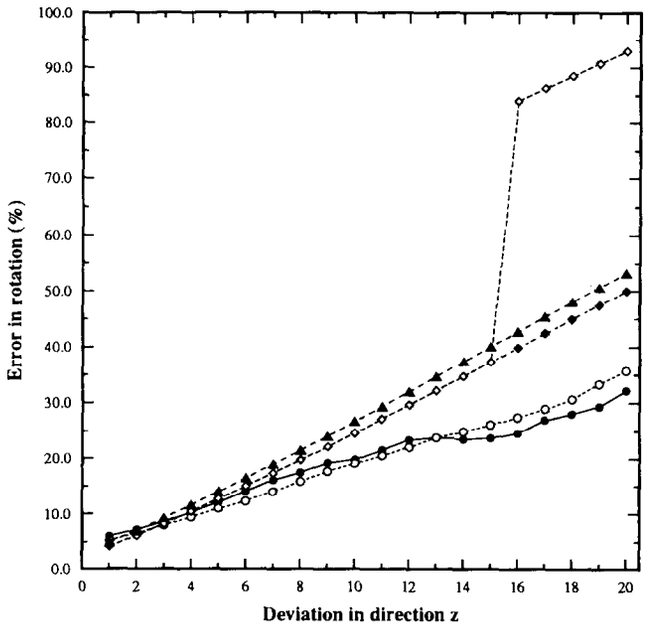

Figure 6. Comparison: error in rotation versus σ_z while σ_x and σ_y are fixed at 1. ●: EKF-AXIS; ○: EKF-QUAT; ◆: MIN-AXIS; ◇: MIN-QUAT; ▲: EIGEN

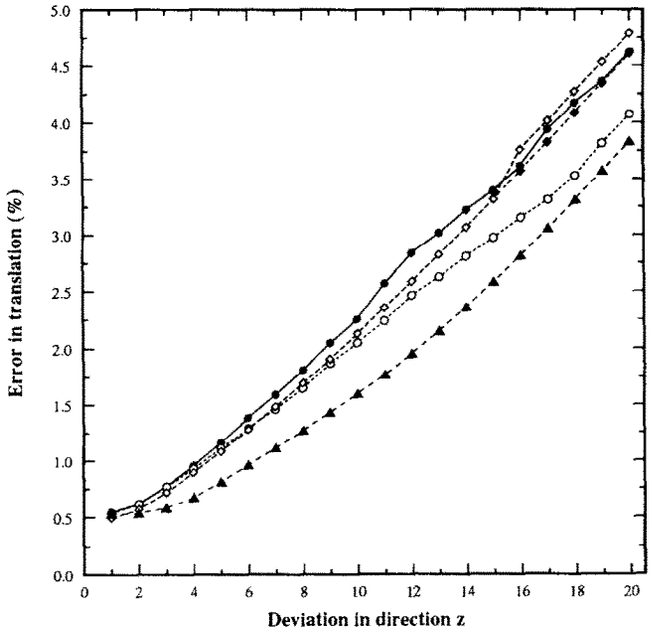

Figure 7. Comparison: error in translation versus σ_z while σ_x and σ_y are fixed at 1. ●: EKF-AXIS; ○: EKF-QUAT; ◆: MIN-AXIS; ◇: MIN-QUAT; ▲: EIGEN

One can expect that the weighted least-squares method using general minimization techniques gives the best estimation, but it would take more time than **MIN-QUAT** or **MIN-AXIS**.

Figures 6 and 7 show the motion error while the standard deviation in z direction (σ_z) varies from 1 to 20 and σ_x and σ_y are fixed at 1. The initial estimate for the recursive methods are all zero. Five iterations of EKF are applied. The error is the average of ten tries. Two correspondences are used. We can observe that error in rotation varies almost linearly with the deviation σ_z, but the slopes of the curves corresponding to **EKF-AXIS** amd **EKF-QUAT** are much smaller than the others. This shows the advantage to take into account the uncertainty, especially when the uncertainty distribution is not uniform. **MIN-QUAT** gives the same error as **MIN-AXIS**, but it diverges when σ_z becomes big. From Figure 7, we see that all methods, except **EIGEN**, give almost the same error in translation.

Results with real data

Up to now, the experiments we have carried out used only the synthetic measurements with the same uncertainty distribution. However, the uncertainty in 3D positions of line segments obtained by stereo triangulation varies with the distance. It increases with its distance to the cameras. Furthermore, the uncertainty of a reconstructed 3D point is different in different coordinate directions: a nearby point has a fairly compact uncertainty; a distant point has a bigger uncertainty in the range than in the horizontal and vertical directions. In this section, we compare the different methods using real data.

Figures 8 and 9 are two stereo frames constructed by our mobile robot in two different positions. Each figure contains two pictures. Figures 8a and 9a are the

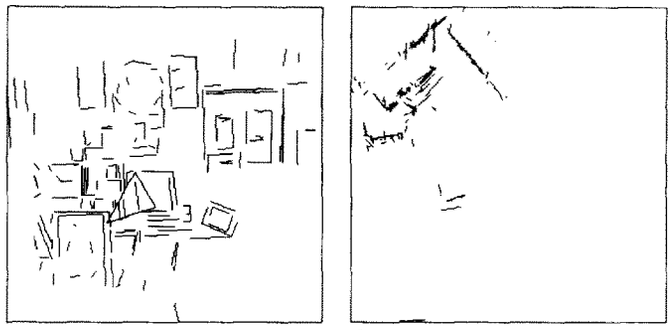

Figure 8. (a) Front and (b) top views of the first 3D scene

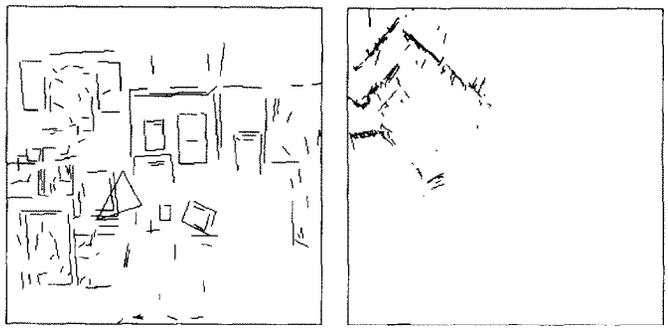

Figure 9. (a) Front and (b) top views of the second 3D scene

projections of the 3D scene on a plane perpendicular to the ground plane in front of the cameras (called front view) and Figures 8b and 9b are the projections on the ground plane (called top view). The triangle in each frame represents the optical centres of the cameras of our trinocular stereo system. There is a large displacement between these two positions (about 10 degrees of rotation and 75 centimetres of translation). We applied the **EKF-AXIS** on these two scenes using 105 correspondences (recovered by the algorithm described in Reference 9). After two iterations of EKF, the motion parameter is given as $[-5.26e-03,\ -1.64e-01,\ 1.45e-03,\ 3.09e+02,\ -1.52e+00,\ 6.89e+02]^T$. The translation is in millimetres. The result is shown in Figure 10. We applied the estimated motion on the first scene and superimposed the transformed scene (in dashed lines) on the second scene (in solid lines). The displacement between the two positions can be recognized from the shift of the triangle.

Five arbitrary correspondences are used to compare the different methods. The initial estimate is zero for

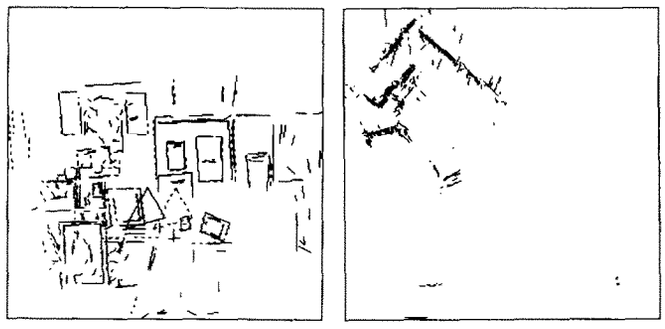

Figure 10. Superposition of two scenes with the result of EKF-AXIS using 105 line segment correspondences

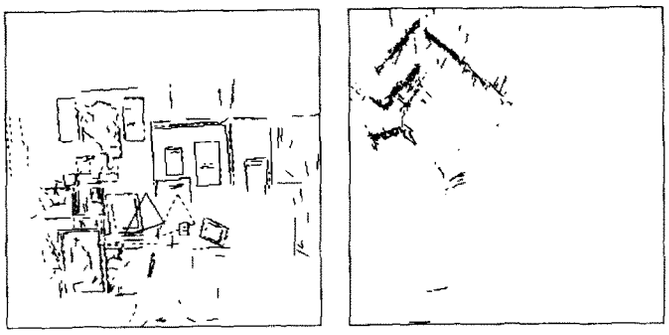

Figure 11. Superposition of two scenes with the result of MIN-AXIS using 5 correspondences

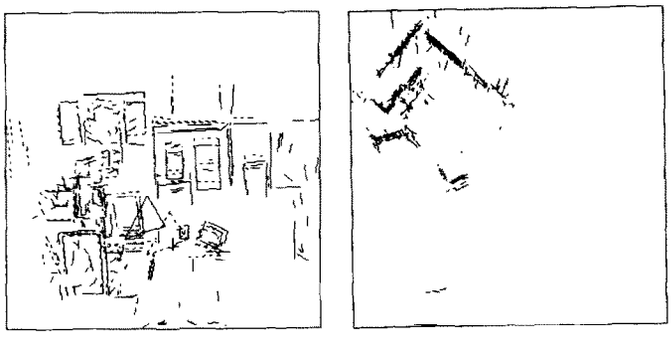

Figure 12. Superposition of two scenes with the result of EKF-QUAT using 5 correspondences

the four recursive methods. Three iterations of EKF are applied in **EKF-AXIS** and **EKF-QUAT**. The result of **EKF-AXIS** is shown in Figure 11, that of **EKF-QUAT** in Figure 12, that of **MIN-AXIS** in Figure 13 and that of **EIGEN** in Figure 14. **MIN-QUAT** gives exactly the same result as **MIN-AXIS**. All these results is compared with the above one and is displayed in Table 3. We arrive that the EKF is much better than the others

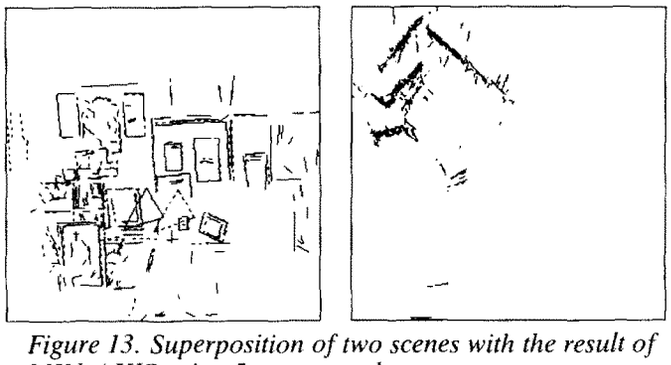

Figure 13. Superposition of two scenes with the result of MIN-AXIS using 5 correspondences

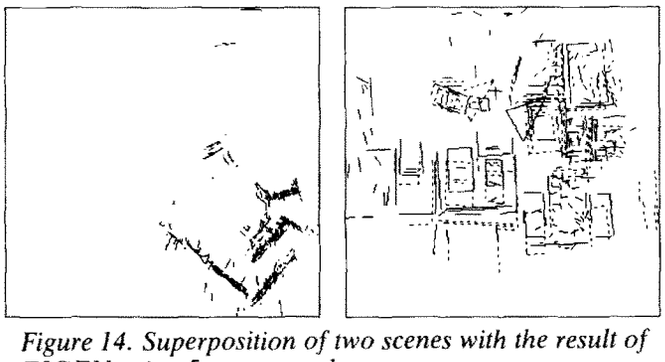

Figure 14. Superposition of two scenes with the result of EIGEN using 5 correspondences

Table 3. Comparison using the real data

Methods	*Rotation error*	*Translation error*
EKF-AXIS	2.25	1.77
EKF-QUAT	2.32	1.37
MIN-AXIS	34.09	22.78
MIN-QUAT	34.09	22.78
EIGEN	53.48	37.74

when 3D data come from stereo. The reason is that the EKF takes into account that the fact that 3D measurements from stereo have different error distribution, and the others do not.

CONCLUSION

In this paper we have presented a number of methods for determining 3D motion from 3D line segment correspondences. Two representations of 3D line segments and two representations of rotation were described. With different representations of 3D line segment and rotation, we showed how to determine motion using the extended Kalman filter, a general minimization process, and the singular value decomposition. These methods have been compared using both synthetic data and real data obtained by a trinocular stereo. Due to space limitations, only part of the results were provided in this paper (see Reference 1 for further details).

From our experiments, we observe that:

1. Uncertainty on measurements should be taken into account, especially when measurements have different uncertainty distributions.
2. When measurements have small uncertainty (less than 2% of segment length), general minimization algorithms give the best results. But when the uncertainty becomes larger, the general minimization algorithms do not give better results than EKF. Furthermore, they are more time consuming.
3. In the general minimization algorithm or EKF, using the quaternion representation is more time consuming and does not give better results than using the rotation axis representation. On the contrary, we observed that using the quaternion representation is less stable.
4. Recursive methods require an initial guess of the solution. When the initial estimate is far from the true one, recursive methods may give a wrong solution. In our experiments, we observe that the recursive methods can converge to the true solution when the initial estimate varies in a wide range from the true one.
5. Using an iterated extended Kalman filter can reduce the effects of non-linearity. Even when few correspondences are available, EKF converges to the true estimate after only five or six iterations.
6. Using the quaternion representation of rotation, we can use the singular value decomposition to obtain the analytical solution of motion. The method is efficient and does not need an initial motion estimate. However, the result is not very significant.
7. EKF can incorporate new measurements incrementally.

We conclude that the extended Kalman filter with the rotation axis representation is preferable, especially when measurements have different distribution of uncertainty like in stereo. The **EKF-QUAT** also gives good results, but it is more time consuming and less stable than the **EKF-AXIS**.

We note that all methods presented in this paper can be directly applied to 3D point data. In that case, at least three 3D point correspondences (non-collinear) are needed. Every three non-collinear points can form two non-parallel segments.

ACKNOWLEDGEMENTS

The authors would like to thank the anonymous referees for their useful comments.

REFERENCES

1 **Zhang, Z** Motion Analysis from a Sequence of Stereo Frames and its Applications PhD thesis, University of Paris-Sud, France (1990)

2 **Nagel, H** 'Image sequences – ten (octal) years-from phenomenology towards a theoretical foundation' *Proc. 8th ICPR* Paris, France (October 1986) pp 1174–1185

3 **Aggarwal, J K and Nandhakumar, N** 'On the computation of motion from sequences of images – a review' *Proc. IEEE* Vol 76 No 8 (August 1988) pp 917–935

4 **Ayache, N and Lustman, F** 'Fast and reliable passive trinocular stereovision' *Proc. First Int. Conf. Computer Vision* London, UK (June 1987)

5 **Faugeras, O D *et al*** 'Toward a flexible vision system' in Pugh, A (ed) *Robot Vision* IFS, UK (1983) pp 129–142

6 **Chen, H H and Huang, T S** 'An algorithm for matching 3-D line segments with application to multiple-object motion estimation' *Proc. IEEE Comput. Soc. Workshop on Computer Vision* (November 30–December 2 1987) pp 151–156

7 **Chen, H H and Huang, T S** 'Maximal matching of 3-D points for multiple-object motion estimation' *Patt. Recogn.* Vol 21 No 2 (1988) pp 75–90

8 **Faugeras, O D and Herbert, M** 'The representation, recognition, and locating of 3D shapes from range data' *Int. J. Robotics Res.* Vol 5 No 3 (1986) pp 27–52

9 **Zhang, Z, Faugeras, O D and Ayache, N** 'Analysis of a sequence of stereo scenes containing multiple moving objects using rigidity constraints' *Proc. Second Int. Conf. Comput. Vision* Tampa, Florida (December 1988) pp 177–186

10 **Huang, T S** 'Motion analysis' in *AI Encyclopedia* John Wiley (1986) pp 620–632

11 **Arun, K S, Huang, T S and Blostein, S D** 'Least-squares fitting of two 3-D point sets' *IEEE Trans PAMI* Vol 9 No 5 (September 1987) pp 698–700

12 **Ayache, N and Faugeras, O D** 'Building, registrating and fusing noisy visual maps' *Proc. ICCV '87*, London, UK (June 1987) pp 73–82

13 **Faugeras, O D** 'A few steps towards artificial 3D vision' in Brady, M (ed) *Robotics Research* MIT Press, USA (1988)

14 **Horn, B K P** *Robot Vision* McGraw-Hill Book Co (1986)

15 **Jazwisky, A M** *Stochastic Processes and Filtering Theory* Academic Press (1970)

16 **Maybeck, P S** *Stochastic Models, Estimation and Control* (Vol 1) Academic Press (1979)

17 **Maybeck, P S** *Stochastic Models, Estimation and Control* (Vol 2) Academic Press (1982)

18 **Faugeras, O D, Ayache, N and Faverjon, B** 'Building visual maps by combining noisy stereo measurements' *Proc. Conf. on Robotics & Automation* IEEE (April 1986)

19 **Ayache, N and Faugeras, O D** 'Maintaining representations of the environment of a mobile robot' *Int. Symposium on Robotics Res.* Santa-Cruz, California (1987)

20 *FORTRAN Library Manual*, Numerical Algorithms Group, mark 11 edition (1984)