International Journal of Computer Vision, 7:3, 211-241 (1992)

Three-Dimensional Motion Computation and Object Segmentation in a Long Sequence of Stereo Frames

ZHENGYOU ZHANG AND OLIVIER D. FAUGERAS
INRIA Sophia-Antipolis, 2004 route des Lucioles, 06565 Valbonne Cedex, France

Received August 8, 1990. Revised November 26, 1991.

Abstract
We address the problem of computing the three-dimensional motions of objects in a long sequence of stereo frames. Our approach is bottom-up and consists of two levels. The first level deals with the tracking of 3D tokens from frame to frame and the estimation of their kinematics. The processing is completely parallel for each token. The second level groups tokens into objects based on their kinematic parameters, controls the processing at the low level to cope with problems such as occlusion, disappearance, and appearance of tokens, and provides information to other components of the system. We have implemented this approach using 3D line segments obtained from stereo as the tokens. We use classical kinematics and derive closed-form solutions for some special, but useful, cases of motions. The motion computation problem is then formulated as a tracking problem in order to apply the extended Kalman filter. The tracking is performed in a prediction-matching-update loop in which multiple matches can be handled. Tokens are labeled by a number called its support of existence which measures their adequation to the measurements. If this number goes beyond a threshold, the token disappears. The individual line segments can be grouped into rigid objects according to the similarity of their kinematic parameters. Experiments using synthetic and real data have been carried out and the results found to be quite good.

1 Introduction

The problem of analyzing sequences of images to extract three-dimensional motion and structure has been at the heart of the research in computer vision for many years. It is very important since its success or failure will determine whether or not vision can be used as a sensory process in *reactive* systems. There are of course many possibilities for attacking the problem and many more remain to be explored. We discuss a few of them.

In fact, image sequence analysis is a rather vague term and can cover several meanings. Our definition is that, given one or several sequences of images acquired from one or several cameras whose relative positions are known and which are rigidly moving in an unknown environment containing a number of mobile rigid objects, we must determine the various relative motions (cameras and objects) and the structure of the scene.

There has been a tremendous amount of work on the analysis of monocular sequences of images. This work has basically followed two main paths: optical flow and token tracking. The philosophy of optical flow is to work in two steps. First estimate from the variation of image intensities the projection of the three-dimensional velocities and second, compute those velocities (in fact the kinematic screw that defines them) and the depth from the optical flow. The reason for splitting the process of recovery into those two steps can be traced back to the work of Gibson [1950] and Koenderink [Koenderink & van Doorn 1975; 1978; Koenderink 1986]. Because of fundamental difficulties such as the aperture effect, researchers have only been able to partially solve the first step [Horn & Schunk 1981; Nagel 1983, 1986; Hildreth 1984], and the results of the second step have been reported to be of very poor quality. These and other reasons such as the fact that the relationship between the optical flow and the projected velocity field is a bit uncertain [Faugeras 1990] and the optical flow

can only be reliably estimated near image discontinuities, have led researchers to explore another route, to track tokens.

The philosophy of the token tracking approach is also to work in two steps. First detect reliable tokens such as curves, corners, from the spatial variations of image intensities, assuming that they correspond to markings on the three-dimensional objects. Second, track them over time and recover the depth and three-dimensional velocities of the corresponding 3D tokens. This tracking is performed by building a kinematic model for the two-dimensional tokens. There are two main possibilities, either to work directly in 3D or to work in 2D. The advantage of the first option is that one works in the "right place" where it makes more sense to model the kinematics of objects but at the cost of being very sensitive to noise. The advantage of the second approach is that one works in the "easy space" where measurement are made but at the cost of not computing directly the values we are really interested in computing. This is where the largest amount of work has been performed [Sethi and Jain 1987; Crowley et al., 1988; Gambotto 1989; Hwang 1989; Deriche & Faugeras 1990; Broida & Chellappa 1986; Broida & Chellappa 1989; Weng et al., 1987; Dickmanns 1987; Dickmanns and Graefe, 1988a,b].

Another possibility for performing this computation is to consider that the two-dimensional tracking gives us matches between different frames and estimate the three-dimensional motions from those matches. This last problem has also received considerable attention. A lot of work has been published on algorithms for recovering motion and structure from n point matches, p line matches, between q views, where typically n is 5, p is 6, and q is 2 or 3 [Ullman 1979; Tsai & Huang 1981; Huang & Tsai 1981; Longuet-Higgins 1981; Yen & Huang, 1983; Tsai & Huang, 1984; Zhuang & Haralick, 1985; Liu & Huang, 1986; Liu & Huang, 1988; Aggarwal & Wang, 1987; Faugeras & Maybank, 1990). These results are theoretically very interesting but are limited to the estimation of the motion of a single object and to the reconstruction of the structure of the scene up to a scale factor unless considerable a priori information is available. Also, due to the complexity of image formation and the nonlinear relation between 3D motion and changes in the images, the solutions have been reported to be very sensitive to noise, and thus have so far been of little practical use except, perhaps, for calibration.

There has also been a large amount of work on stereo [Baker & Binford, 1981; Grimson, 1981, 1985; Nishihara 1984; Ohta & Kanade 1985; Pollard et al. 1985; Marr & Poggio 1976; 1979; Yachida 1986; Ayache & Lustman 1987; Kitamura & Yachida 1990] which can be seen as the analysis of two or three sequences of images (if we use binocular or trinocular stereo), each sequence being limited to only one image. The main problem has been, and remains, to establish correspondences between the images and to reconstruct a depth map which is as dense as possible.

Much less has been done on the analysis of several, simultaneously acquired, sequences of images. Clearly the amount of information is much higher and one would hope that this would allow us to solve the problem in a more robust fashion. It is not obvious, however, to decide how to proceed and build upon existing techniques, for example those developed for the analysis of monocular sequences and stereo. In [Zhang et al., 1988; Zhang & Faugeras 1991; Zhang & Faugeras 1992], we have proposed an algorithm based upon the hypothesize-and-verify paradigm to match 3D line segments and to compute 3D displacements between two 3D frames obtained from stereo. In order to reduce the complexity of the method, we have made the assumption that objects are rigid. This algorithm has been extended to deal with the case where several mobile objects are present.

The solution we explore in this paper is the following. We assume that we can do reliable stereo at a reasonable rate, let us say five times a second to fix ideas. We then *match* the set of sequentially reconstructed three-dimensional representations and estimate motion. Therefore, we do three-dimensional motion from three-dimensional structure.

Of course, there are many details that need to be filled in:

1. What are the three-dimensional representations that are used?
2. How do we match them?
3. How do we estimate motion from the matches?

The answer to point number 1 is that we use an edge-based stereo algorithm that has been developed in the past [Ayache & Lustman 1987] and put into hardware [Faugeras et al. 1988b]. It can deliver three-dimensional line segments at the rate of 5Hz. Therefore, our representation is quite simple and consists of sets of three-dimensional line segments. It is not clear how crucial

this assumption is for the whole system. We believe that many of the ideas described here can be used for other simple geometric primitives, other tokens, even though in the details of the current implementation the line segment assumption plays an important role.

The answer to points number 2 and 3 is that we build a model of the kinematics of each token, assuming that it is attached to a moving rigid object. We use this model to *predict* the appearance of the token in the three-dimensional visual map obtained at the next time instant. We use this prediction to *verify* whether the token is present and *match* it, if possible, to a real token. The match is then used to *update* the kinematic model. The whole process is implemented as a Kalman filter. Therefore, matching and estimation of motion are intimately related in this approach. One interesting feature is that we actually integrate the kinematic equations by making the assumption that the motion we observe can be well approximated on a short-time scale by constant angular velocity and constant linear acceleration. This is in general true only if the time-sampling frequency is high enough so that the accelerations can be neglected between two sampling times.

We also tackle the following problems that arise when we deal with (long) sequences of images:

- *Occlusion*: A moving object may be partially or totally occluded by the background or by other objects.
- *Disappearance*: A moving object in the current field of view may move partially or totally out of it in the next frames.
- *Appearance*: A previously unseen object may partially or totally come into view.

Clearly, occlusion is related to disappearance and appearance, since when we talk about the occlusion of an object, we mean that some of its features disappear for a moment and may eventually reappear in the future. Those three events are due to regular transformations of the scene. We must add to them a fourth which is due to the failure of the algorithms that produce the description:

- *Absence*: When features that should be present are not, due to the failure of the feature extraction (or reconstruction) process.

These remarks bring forward an interesting aspect of the problem, namely that there are always two kinds of tokens: those that have been seen for a sufficiently long time so that the system has been able to build a good model of their kinematics, and those that have just entered the field of view and for which no kinematics information is available. The first kind of tokens is easily dealt with since it is likely that the prediction stage will help to cut down heavily the number of tentative candidates to a match in the next frame. For the second kind, a computational explosion is likely to happen: in order to find the right match, we may have to explore a large number of possibilities and if we make the wrong choice we will lose track of the token. Therefore our system can be seen as operating in two modes, the first one called the *continuous* mode and the second called the *bootstrapping* mode.

The continuous mode applies to tokens for which the system has built up a kinematic model with low uncertainty. The model at time t is used to predict the position and orientation of the token in the scene at time $t + \Delta t$. Since the uncertainty of the model is small, the search for corresponding tokens can be restricted to a small zone around the predicted token.

The bootstrapping mode assumes no knowledge of the kinematics of the token, i.e., it assumes that it is not moving, with a large uncertainty. Its position and orientation at time $t + \Delta t$ are predicted to be the same as those at time t but, since the uncertainty of the model is large, the search for corresponding tokens is conducted in a larger zone than in the previous mode, leading to the possibility of many candidates.

One interesting feature of both modes is that they use the idea of *least-commitment* and, instead of forcing a decision, may make multiple correspondence choices and use the time continuity to throw away later the ones which are not confirmed by the measurements.

Another feature of our approach is that we can detect multiple motions by grouping tokens that have similar kinematic models, thus obtaining a segmentation of the scene into "objects" (i.e., sets of tokens) moving rigidly.

We are not the first ones to investigate this problem from that viewpoint. Young and Chellappa [1988] describe the computer simulation of a system that uses a number of noisy 3D points assumed to belong to the same rigid object to estimate its motion. In their work, the problem of obtaining the matches from frame to frame and the problem of multiple objects are not addressed.

2 Statement of the Problem

We address the motion-tracking problem that arises in the context of a mobile vehicle navigating in an

unknown environment where other mobile bodies such as humans or robots may also be moving. A stereo rig mounted on the mobile vehicle provides a sequence of 3D maps of the environment. The current stereo system is trinocular [Ayache and Lustman 1987], and the 3D tokens we are using are line segments produced by significant intensity discontinuities in the images. Although the framework to solve the motion-tracking problem developed here arises in this specific context, we believe it should be applicable in other contexts; in particular we could use other 3D primitives, for example, points, combinations of points and lines, curves.

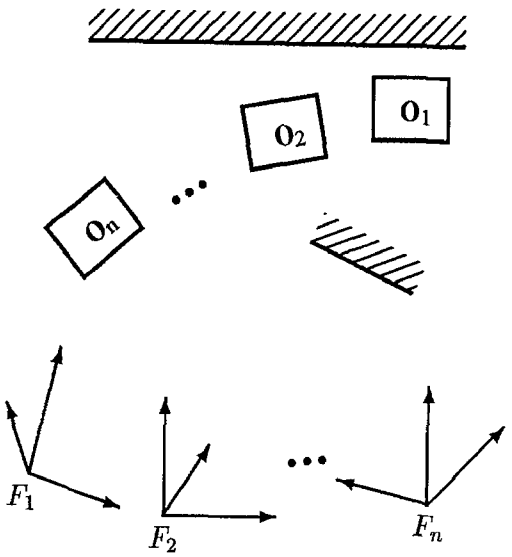

Fig. 1. Illustration of the motion-tracking problem.

The situation is illustrated in figure 1. The static environment is represented by hatched regions. Only one moving object is drawn, represented by a square. The mobile robot is represented by a frame of reference, which is the one of the stereo system. This reference frame is attached to the mobile robot and its numerical parameters are determined in the camera calibration phase [Faugeras & Toscani 1986]. In the figure, the object undergoes a general motion from right to left, and the robot moves from left to right. We want to solve the following problems:

1. Find the positions of static and moving objects in each stereo frame,
2. Determine the motion of the robot as well as motions of the moving objects with respect to the static environment.

As stated earlier, in order to resolve these issues we have to handle problems in dynamic scene analysis such as occlusion, appearance, disappearance, and absence of features.

Those problems can be solved at the level of objects: *object tracking*, or at the level of features that constitute objects: *token tracking*. In the object-tracking approach, the scene must first be segmented into objects, and this in general requires high-level knowledge about the characteristics of objects such as rigidity and geometry (planar world). This approach is in general difficult. In some special cases, such as Radar imagery and in the experiment reported by Gordon [1989] using tennis balls, objects can be easily detected and can be replaced by points (usually their centers of gravity). In the token-tracking approach, no such knowledge is required and the tracking process can be carried out in parallel for each token. For this reason and also for the following, we track 3D line segments instead of 3D objects:

- Objects can be later identified by grouping line segments with similar motion,
- After tracking individual line segments, one can detect multiple moving objects, articulated objects, or even deformable objects based on common motion characteristics.

Because of this, the hypothesis that assumes that objects are moving rigidly can be somewhat relaxed in the analysis of long sequences.

3 A Framework to Solve the Motion-Tracking Problem

To clarify the presentation, we call the 3D line segments being tracked the *tokens* and the currently observed 3D line segments the *scene tokens*.

3.1 Outline of the Motion-Tracking Algorithm

Our motion-tracking algorithm consists of two levels (see figure 2). In the figure, single arrows represent data flow, and the double arrow represents the flow of control.

The low level is called the *tracking team*. A token being tracked can be considered as one of the team members. As we discussed earlier, the token-tracking

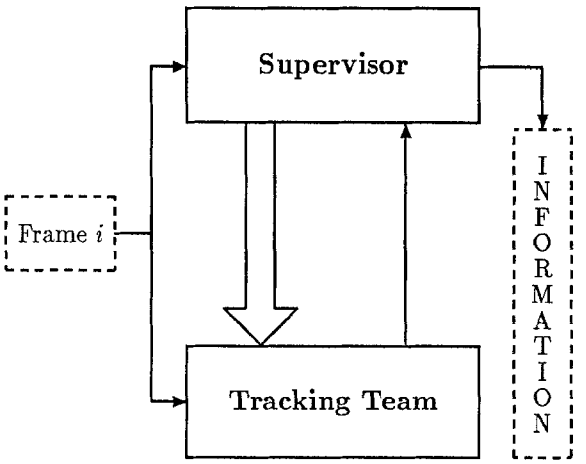

Fig. 2. An architecture for motion tracking.

process uses a model of the 3D kinematics, instead of a model of the evolution of the token parameters. A token is then characterized by its *position* and *orientation* in the current frame, its *kinematic parameters*, and a positive number called *support*. The support indicates the degree of support for the existence of the token, which will be described in section 9.

When a new frame is acquired, each token being tracked searches the whole frame for a correspondence. The search space can be considerably reduced by using a kinematic model and information from previous frames: before the new frame is available, one can predict the occurrence of each token in the new frame based on the kinematic model. When the new frame is obtained, one only needs to look for scene tokens in the neighborhood of the predicted position.

When a match is found, the parameters of the token kinematic model are updated, the position parameters are replaced by those of its match[1], and the support parameter is also updated. The prediction and update of the position and kinematic parameters are done by using an extended Kalman filter. The matching process is based on the Mahalanobis distance. Due to occlusion or absence of some scene tokens in the current frame, a token may not find any match in the neighborhood of its predicted position in the current frame. Of course, this phenomenon may also occur due to the disappearance of the token. To handle the occlusion and absence of scene tokens, it is necessary to hypothesize the existence of the token and continue to change its kinematic parameters according to the kinematic model and update its support.

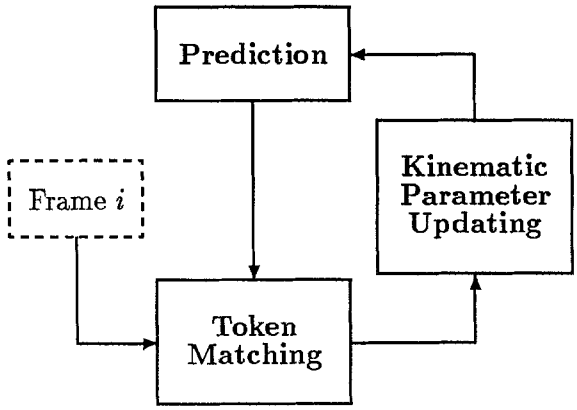

Fig. 3. Block diagram of the toaken tracking algorithm.

Figure 3 shows the block diagram of the algorithm, and the details will be described in the following sections. As we can observe, the above process can be performed independently for each token to be tracked, and this allows a completely parallel implementation.

The high level is called the *supervisor*. It has three main functions:

- Grouping tokens with similar kinematic parameters as a single object. If there exist multiple moving objects, they can be segmented on the grounds that they undergo different motions. We describe later the details about how to group tokens.
- Monitoring the tracking team by detecting the following events:

1. **Appearances**: When a new token appears, that is, when a scene token in the current frame cannot be matched with any token being tracked, then the supervisor activates an additional token in the tracking team. This new token starts the same process as the others.
2. **False matches**: When a token loses the support of its existence (see section 9), the supervisor then deactivates this token. Unusually such tokens have been activated due to false matching in the previous frames. In a parallel implementation, the processor occupied by this token would be freed, and could be used by some new token.
3. **Disappearances**: When a token moves out of the field of view, the supervisor deactivates this token. We can easily determine whether a tracked token is out of the field of view by projecting it onto one of the camera planes. Just as in the

previous case, in a parallel implementation, the processor occupied by this token would be freed, and could be reutilized.

4. **Multiple matches**: A token being tracked may find multiple matches in the current frame with a criterion defined a priori (the Mahalanobis distance, for example), especially when several scene tokens are near to each other. A common way to solve this problem is to choose the scene token that is the nearest to the predicted position (*best-first search*), as in [Crowley et al. 1988; Deriche and Faugeras 1990]. This may lead to unpredictable results. A more robust approach is to keep tracking the token using several nearest-scene tokens in the current frame; thus a token can be split. This approach can be called *beam search*. In our implementation, we choose the two nearest scene tokens to the predicted position in the sense of the Mahalanobis distance (see section 8), if both their distances are less than some threshold. The token updates its kinematic parameters by incorporating the nearest scene token. If the second-nearest-scene token exists, then the token reports it to the supervisor. The supervisor activates an additional token by integrating the original token and the matched one. The beam-search strategy is utilized in other research fields, such as in the HARPY speech understanding system [Lowerre & Reddy 1980]. This strategy has been found to be efficient as well as robust.
5. **About changes**: A potential capacity of the supervisor to monitor the tracking team is to detect abrupt changes in the motion of a token due, for example, to collision, and to reinitialize its kinematic parameters.

— Providing information to other components of the global system. For example, in an active tracking application, one may need to control the motion of the robot or adjust the camera parameters to adapt the changing situation based on the information provided by the motion-tracking algorithm. The information may include the kinematics of the robot (ego-motion) and the kinematics and relative positions of the moving objects.

3.2 A Pedagogical Example

Figure 4 shows an example of how the tracking team works. At t_2, token 1 is split in two (token 1 and token 1') due to ambiguous matches. At t_3, token 1' cannot find a correspondence in the current frame, and it makes a hypothetical extension to cope with the occlusion problem. But because too many such hypothetical extensions are made consecutively, it loses its support for existence at t_5 and is then deactivated. At t_5, token 1 cannot find its correspondence in the current frame, and it makes a hypothetical extension. It finds its correspondence at t_6. Thus the occlusion problem is handled gracefully.

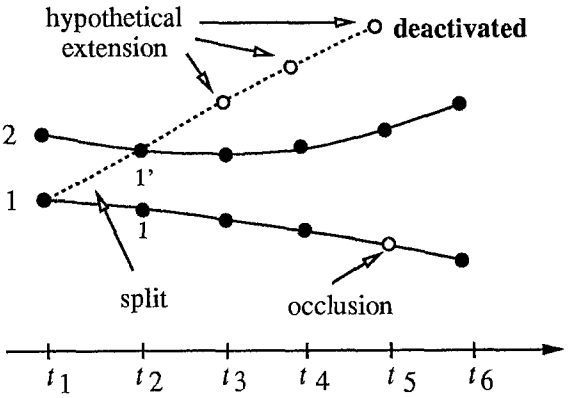

Fig. 4. An example of motion tracking.

4 Representation of 3D Line Segments

Our stereo system reconstructs a set of 3D line segments about its environment. These segments may correspond to the contours of objects, to shadows, or to region markings. Line segments addressed here are *oriented* thanks to the intensity contrast. In this section, we propose a new representation for a line segment which we think is well adapted for the task at hand.

4.1 Motivation

It is a consensus among many researchers in the field that uncertainty should be *explicitly* represented and manipulated in computer vision and robotics applications [Ayache & Faugeras 1989; Durrant-Whyte 1988]. We can think of uncertainty as follows. Let us model the features as random and consider their probability density functions. In practice those functions are very hard to estimate and one is usually satisfied with the first few moments, usually the first two, the mean vector and the covariance matrix. This does not imply that the features are modeled as Gaussian but only that we

neglect their higher-order moments. The question of whether this poses problems, for example with the Kalman filter, is answered in section 6.

A related question is the following. Suppose that we have a random feature vector $\mathbf{x}$ with mean $\mathbf{x}_0$ and covariance matrix Λ_x to which we apply a nonlinear function $\mathbf{f}$ to produce a new random feature $\mathbf{y}$. The question is to compute the mean and covariance matrix of $\mathbf{y}$. One way to do this is to compute the Taylor series expansion of $\mathbf{f}$ in the vicinity of $\mathbf{x}_0$. If we perform this expansion up to the second order, we obtain

$$\begin{aligned}\mathbf{y} &= \mathbf{f}(\mathbf{x}) \\ &= \mathbf{f}(\mathbf{x}_0) + \mathbf{f}'(\mathbf{x}_0)(\mathbf{x} - \mathbf{x}_0) + O(\|\mathbf{x} - \mathbf{x}_0\|^2)\end{aligned}$$

This shows that, up to the first order, we have

$$E(\mathbf{y}) = \mathbf{f}(\mathbf{x}_0) = \mathbf{f}(E(\mathbf{x}))$$

and

$$\begin{aligned}\Lambda_{\mathbf{y}} &= E\left[(\mathbf{y} - E(\mathbf{y}))(\mathbf{y} - E(\mathbf{y}))^T\right] \\ &= \mathbf{f}'(\mathbf{x}_0)\Lambda_{\mathbf{x}}\mathbf{f}'(\mathbf{x}_0)^T \qquad (1)\end{aligned}$$

Of course these may be poor approximations if the second order term is not negligible. For example, up to the third order, the mean is given by[2]

$$E(y) = f(x_0) + \frac{1}{2}f''(x_0)\sigma_x^2$$

This formula clearly shows that the fact that the second-order term can or cannot be neglected depends upon both the magnitude of the second-order derivative of f and of the variance of x.

In this article, we assume that the first-order approximation is sufficient either because the second-order derivatives are small compared to the first-order derivatives, or because the second-order moments are small, or both.

A line segment is usually represented by its endpoints M_1 and M_2, which require 6 parameters, and their covariance matrixes Λ_1 and Λ_2. Λ_1 and Λ_2 are estimated by stereo triangulation from point correspondences [Ayache 1988]. Equivalently, a line segment can be represented by its direction vector $\mathbf{v}$, its length l, and its midpoint M, and their covariance matrixes.

But we cannot directly use these parameters in most cases. As explained in section 4.2.2, the endpoints or the midpoint of a segment are not reliable. Thus, instead of the line segment, the infinite line supporting the segment is usually used, as in [Kim & Aggarwal 1987]. In an earlier version of our algorithm for motion analysis of two stereo views [Ayache & Faugeras 1987a; Faugeras et al. 1988a; Zhang et al. 1988], a line segment is treated in a mixed way. The infinite supporting line is used in estimating motion and the line segment is used in matching.

Many representations have been proposed in the literature for a line segment [Ayache & Faugeras 1987b; Roberts 1988]. The main problem is that *the uncertainty on the line parameterization does not reflect that of the segment that the line supports* (see Zhang [1990] for more details). A segment with big uncertainty may yield a small uncertainty in the line parametrization, for example, if its uncertainty is in the direction of the line.

4.2 Our Representation

Because of the deficiencies of the previous representations for a line or a line segment, we use a five-parameter representation for a line segment: two for the orientation, three for the position of a segment. This is a trade-off between an infinite line and a line segment. If we add the length, a line segment can then be fully specified. Special attention is given to the representation of uncertainty.

4.2.1 Representing the Orientation by Its Euler Angles ϕ and θ. Let us consider the spherical coordinates. Let $\mathbf{u} = [u_x, u_y, u_z]^t$ be a unit vector of orientation, we have:

$$\begin{cases} u_x = \cos\phi \sin\theta \\ u_y = \sin\phi \sin\theta \\ u_z = \cos\theta \end{cases} \qquad (2)$$

with $0 \le \phi < 2\pi$, $0 \le \theta \le \pi$.

From $\mathbf{u}$, we can compute ϕ and θ:

$$\phi = \begin{cases} \arccos \dfrac{u_x}{\sqrt{1 - u_z^2}} & \text{if } u_y \ge 0 \\ 2\pi - \arccos \dfrac{u_x}{\sqrt{1 - u_x^2}} & \text{otherwise} \end{cases}$$

$$\theta = \arccos u_z \qquad (3)$$

If we denote $[\phi, \theta]^t$ by ψ, then the mapping between ψ and $\mathbf{u}$ is 1-to-1, except when $\theta = 0$. When $\theta = 0$, ϕ is not defined. This will show in the covariance matrix of ψ as a very large entry for the variance of ϕ, indicating that the ϕ-measurement cannot be trusted very much [Zhang 1990].

Another problem with this representation is the discontinuity in ϕ when a segment is nearly parallel to the plane $y = 0$. In that case, the angle ϕ may jump from the interval $[0, \pi/2)$ to the interval $(3\pi/2, 2\pi)$, or vice versa. This discontinuity must be dealt with in matching and fusion.

In the following, we assume that the direction vector $\mathbf{v} = [x, y, z]^t$ and its covariance matrix Λ_v of a given segment are known. We want to compute ψ and its covariance matrix Λ_ψ from $\mathbf{v}$ and Λ_v. ϕ and θ are simply given by

$$\phi = \begin{cases} \arccos \dfrac{x}{\sqrt{x^2 + y^2}} & \text{if } y \geq 0 \\ 2\pi - \arccos \dfrac{x}{\sqrt{x^2 + y^2}} & \text{otherwise} \end{cases}$$

$$\theta = \arccos \frac{z}{\sqrt{x^2 + y^2 + z^2}} \tag{4}$$

Since the relation between ψ and $\mathbf{v}$ is not linear, we use the first order approximation of equation (1) to compute the covariance matrix Λ_ψ from $\Lambda_\mathbf{v}$. That is

$$\Lambda_\psi = \frac{\partial \psi}{\partial \mathbf{v}} \Lambda_\mathbf{v} \frac{\partial \psi^t}{\partial \mathbf{v}} \tag{5}$$

where the Jacobian matrix

$$\frac{\partial \psi}{\partial \mathbf{v}} = \begin{bmatrix} \dfrac{\partial \phi}{\partial x} & \dfrac{\partial \phi}{\partial y} & \dfrac{\partial \phi}{\partial z} \\ \dfrac{\partial \theta}{\partial x} & \dfrac{\partial \theta}{\partial y} & \dfrac{\partial \theta}{\partial z} \end{bmatrix}$$

4.2.2 Modeling the Midpoint of a 3D Line Segment. We choose the midpoint as the three parameters to localize a segment, but a special treatment on the covariance is introduced to characterize the uncertainty in the location of a segment.

The reason for this is that the way the uncertainty of the endpoints of a three-dimensional segment is computed takes only into account the uncertainty of the pixel coordinates due to the edge detection process and the uncertainty of the calibration of the stereo rig [Ayache & Faugeras 1989]. But it does not take into account the uncertainty due to the variations, in the different images of the stereo triplet, of the polygonal approximations of corresponding contours. There are two main sources for these variations.

The first is purely algorithmic: because of noise in the images and because we sometimes approximate significantly curved contours with line segments, the polygonal approximation may vary from frame to frame inducing a variation in the segments' endpoints that has not been accounted for. The second is physical: because of partial occlusion in the scene, a segment can be considerably shortened or lengthened and this has also to be taken into account in the modeling of uncertainty.

In an attempt to cope with all this, we model the midpoint $\mathbf{m}$ of a segment M_1M_2 as

$$\mathbf{m} = \mathbf{M} + n\mathbf{u} \tag{6}$$

where $\mathbf{M} = (M_1 + M_2)/2$, $\mathbf{u}$ is the unit direction vector of the segment, and n is a random scalar. Equation (6) says in fact that the midpoint has some extra uncertainty attached to it. It may vary randomly along the line supporting it in successive views.

The random variable n in equation 6 is modeled as zero-mean with deviation σ_n, a positive scalar. If a segment is reliable, σ_n may be chosen to be a small number; if not, it may be chosen to be a big number. In our implementation, σ_n is related to the length l of the segment, that is, $\sigma_n = \kappa l$, where κ is some constant. That is to say that the longer a segment is, the bigger the deviation σ_n is. That is reasonable since a long segment is much more likely to be broken into smaller segments in other views. In our experiments, $\kappa = 0.2$.

In order to compute the covariance of $\mathbf{m}$, we should first compute $\mathbf{u}$ and $\Lambda_\mathbf{u}$. The unit direction vector $\mathbf{u}$ and its covariance $\Lambda_\mathbf{u}$ can be computed from a nonnormalized direction vector $\mathbf{v}$ and its covariance matrix $\Lambda_\mathbf{v}$ from equation (1). Indeed, we have

$$\mathbf{u} = \frac{\mathbf{v}}{\|\mathbf{v}\|}$$

$$\Lambda_\mathbf{u} = \frac{\partial \mathbf{u}}{\partial \mathbf{v}} \Lambda_\mathbf{v} \frac{\partial \mathbf{u}^t}{\partial \mathbf{v}} \tag{7}$$

where $\partial \mathbf{u}/\partial \mathbf{v}$ is a 3×3 matrix

$$\frac{\partial \mathbf{u}}{\partial \mathbf{v}} = \frac{\mathbf{I}_3}{\|\mathbf{v}\|} - \frac{\mathbf{v}\mathbf{v}^t}{\|\mathbf{v}\|^3}$$

Note that the covariance matrix $\Lambda_\mathbf{u}$ is singular (the determinant is zero). This is due to the fact that the three components of $\mathbf{u}$ are not independent since $\|\mathbf{u}\| = 1$.

At this point, the covariance of $\mathbf{m}$ can be computed. We start with the covariance of $n\mathbf{u}$. Since n and $\mathbf{u}$ are independent of each other, we have

$$E[n\mathbf{u}] = E[n]E[\mathbf{u}] = 0 \tag{8}$$

$$\begin{aligned}\Lambda_{n\mathbf{u}} &= E[(n\mathbf{u})(n\mathbf{u})^t] = E[n^2]E[\mathbf{u}\mathbf{u}^t] \\ &= \sigma_n^2(\Lambda_{\mathbf{u}} + \bar{\mathbf{u}}\bar{\mathbf{u}}^t),\end{aligned} \tag{9}$$

where $\bar{\mathbf{u}} = E[\mathbf{u}]$. Now we have

$$E[\mathbf{m}] = E[\mathbf{M}] + E[n\mathbf{u}] = E[\mathbf{M}]$$

and

$$\begin{aligned}\Lambda_{\mathbf{m}} &= E[(\mathbf{m} - E[\mathbf{m}])(\mathbf{m} - E[\mathbf{m}])^t] \\ &= E[(\mathbf{M} - E[\mathbf{M}])(\mathbf{M} - E[\mathbf{M}])^t] \\ &\quad + E[(n\mathbf{u})(n\mathbf{u})^t] + E[n(\mathbf{M} - E[\mathbf{M}])\mathbf{u}^t]\end{aligned}$$

Since n is independent of $\mathbf{M}$ and $\mathbf{u}$ and has zero-mean, the last two terms are equal to 0 and

$$\Lambda_{\mathbf{m}} = \Lambda_M + \Lambda_{n\mathbf{u}}$$

We consider that M_1 and M_2 are independent, therefore

$$\Lambda_M = \frac{\Lambda_1 + \Lambda_2}{4}$$

and this completes the computation of Λ_M, up to the first order.

If we add another parameter l to denote the length of the segment, we can then exactly represent a line segment. This ends our modeling of a line segment. See [Zhang & Faugeras 1990a; Zhang 1990] for more details.

5 Kinematic Model

A common approach to model the motion kinematics is to divide the motion into two parts: a rotation about a point (called the *center of rotation*) and a translation of the center of rotation. The rotation is often assumed to be constant angular velocity or constant precession. The trajectory of the rotation center is assumed to be well approximated by the first k terms of a polynomial ($k \geq 0$). See [Broida & Chellappa 1986, 1989; Weng et al. 1987; Young & Chellappa 1988] for such a modeling. We show [Zhang 1990] that in case of constant angular velocity, that modeling is a special case of the one described in this section. Webb & Aggarwal [1982] used the *fixed-axis assumption* to recover the 3D structure of moving rigid and jointed objects from several single-camera views. The fixed-axis assumption is stated as follows: Every rigid object movement consists of a translation plus a rotation about an axis that is fixed in direction for short periods of time. In this section, we describe the kinematics of the well-known classical model of rigid bodies and then derive the closed-form solutions for some special motions.

5.1 The Classical Kinematic Model

Given a Cartesian system of reference $Oxyz$ for rigid bodies in which a rigid body is in motion. Choose a point on the solid, noted by P. Consider any point M of the solid, then its velocity $\mathbf{v}_M$ is the sum of the velocity $\mathbf{v}_P$ of the point P and the rotation around the point P (see figure 5), that is

$$\mathbf{v}_M(t) = \mathbf{v}_P(t) + \omega(t) \times \vec{PM} \tag{10}$$

where $\omega(t)$ is called the *angular velocity*, and $\times$ denotes the cross-product of two vectors.

The above equation is true for any point P. For simplicity, we choose the origin as the point P, that is, $P = O$, and we have the kinematic model as follows:

$$\mathbf{v}_M(t) = \mathbf{v}(t) + \omega(t) \times \vec{OM} \tag{11}$$

The kinematics of any point M of the body is completely characterized by $\mathbf{v}(t)$, the velocity of the point of the solid coinciding with the origin of the reference system, and $\omega(t)$, the angular velocity of the point M around the origin. The pair $(\omega(t), \mathbf{v}(t))$ is called the kinematic screw of the solid.

Let us replace $\vec{OM}$ in equation (11) by $\mathbf{p}(t)$, and $\mathbf{v}_M(t)$ by $\dot{\mathbf{p}}(t)$, where $\dot{\mathbf{p}}(t)$ denotes the time derivative

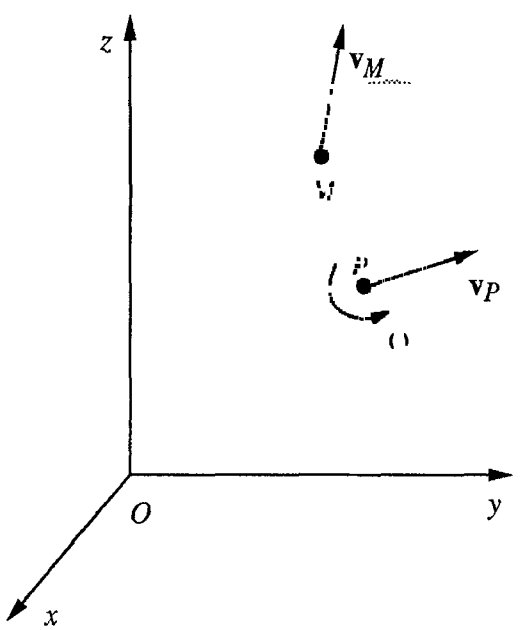

Fig. 5. Illustration of the classical kinematics for rigid bodies.

of $\mathbf{p}(t)$, that is, $d\mathbf{p}(t)/dt$. For the sake of clarity, we write the time as a subscript. For instance, $\mathbf{p}(t)$ is written as $\mathbf{p}_t$. If we denote by $\tilde{\mathbf{v}}$ the antisymmetric matrix associated with $\mathbf{v} = [v_1, v_2, v_3]^t$, that is,

$$\tilde{\mathbf{v}} = \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix} \tag{12}$$

then we have $\mathbf{v} \times \mathbf{u} = \tilde{\mathbf{v}}\mathbf{u}$. Equation (11) can therefore be rewritten as a first-order differential equation in $\mathbf{p}_t$:

$$\dot{\mathbf{p}}_t = \tilde{\omega}_t \mathbf{p}_t + \mathbf{v}_t \tag{13}$$

It is very difficult to get the solution of equation (13) for a general motion. In the appendix of this paper, we show that a closed form can be obtained if the angular velocity is constant and the translational velocity is described by a polynomial of degree $n (n \geq 0)$. We give in the following section the closed form of the kinematic models for two special motions.

5.2 Closed-form Solutions for Two Special Motions

In the case of constant angular and translational velocities, we have a simple closed-form of the solution (theorem 1). Let $\omega_t = \omega$ and $\mathbf{v}_t = \mathbf{v}$.

Theorem 1. *Trajectory in the case of constant angular and translational velocities:*

The trajectory of a point $\mathbf{p}_t$ *given by equation (13) is given, in the case of constant angular velocity* ω *and translational velocity* $\mathbf{v}$, *as*

$$\mathbf{p}_t = W\mathbf{p}_0 + V\mathbf{v} \tag{14}$$

where

$$W = \mathbf{I}_3 + \frac{\sin(\theta\,\Delta t)}{\theta}\tilde{\omega} + \frac{1 - \cos(\theta\,\Delta t)}{\theta^2}\tilde{\omega}^2 \tag{15}$$

$$V = \mathbf{I}_3 \Delta t + \frac{1 - \cos(\theta\,\Delta t)}{\theta^2}\tilde{\omega} + \frac{\theta\,\Delta t - \sin(\theta\,\Delta t)}{\theta^3}\tilde{\omega}^2 \tag{16}$$

and $\theta = \|\omega\|$, $\Delta t = t - t_0$, $\mathbf{I}_3$ *is the* 3×3 *identity matrix, and* $\mathbf{p}_0 = \mathbf{p}_{t_0}$. ■

See the appendix for the proof. From theorem 1, we can observe that when $\omega = 0$ (i.e., pure translation), then

$$\mathbf{p}_t = \mathbf{p}_0 + \mathbf{v}(t - t_0) \tag{17}$$

This is the well-known equation for a point moving on a straight line with constant velocity.

When angular velocity and translational acceleration are constant, we have the following equations:

$$\omega_t = \omega$$
$$\mathbf{v}_t = \mathbf{v} + \mathbf{a}(t - t_0) \tag{18}$$

where ω denotes the constant angular velocity, $\mathbf{v}$ denotes the translational velocity at $t = t_0$, and $\mathbf{a}$ denotes the constant translational acceleration. The trajectory of a point in this case is defined by the following theorem.

Theorem 2. *Trajectory with constant angular velocity and translational acceleration:*

The trajectory of a point $\mathbf{p}_t$ *given by equation (13) is given, in the case of constant angular velocity* ω *and constant translational acceleration* $\mathbf{a}$, *as*

$$\mathbf{p}_t = W\mathbf{p}_0 + V\mathbf{v} + A\mathbf{a} \tag{19}$$

where W is the same as in equation (15), V is the same as in equation (16), and

$$A = \frac{\Delta t^2}{2}\mathbf{I}_3 + \frac{\theta\,\Delta t - \sin(\theta\,\Delta t)}{\theta^3}\tilde{\omega} + \frac{(\theta\,\Delta t)^2 - 2(1 - \cos(\theta\,\Delta t))}{2\theta^4}\tilde{\omega}^2 \tag{20}$$

and $\theta = \|\omega\|$, $\Delta t = t - t_0$ *and* $\mathbf{I}_3$ *is the* 3×3 *identity matrix.* ■

See the appendix for the proof. From theorem 2, we observe that when $\omega = 0$ (i.e., pure translation), then

$$\mathbf{p}_t = \mathbf{p}_0 + \mathbf{v}(t - t_0) + \mathbf{a}\frac{(t - t_0)^2}{2} \tag{21}$$

This is the well-known equation for a point moving on a straight line with constant acceleration.

6 Extended Kalman Filter

In this section, we adapt the extended Kalman filter (EKF) [Maybeck 1982] formulation to our problem. It will then be used in the next section to solve our motion-tracking problem. For more details, the reader is referred to [Maybeck 1979; Maybeck 1982].

In practice, the individual state variables of a dynamic system cannot be determined exactly by direct measurements; instead, we usually find that the measurements we make are functions of the state variables and that

these measurements are corrupted by random noise. The system itself may also be subjected to random disturbances. We want to estimate the state variables from the noisy observations.

If we denote the state vector by $\mathbf{s}$ and denote the measurement vector by $\mathbf{x}$, a dynamic system (in discrete-time form) can be described by

$$\mathbf{s}_{i+1} = \mathbf{h}_i(\mathbf{s}_i) + \mathbf{n}_i, \qquad i = 0, 1, \ldots \tag{22}$$

$$\mathbf{f}_i(\mathbf{x}_i, \mathbf{s}_i) = 0, \qquad i = 0, 1, \ldots \tag{23}$$

Equation (22) is the state equation, and equation (23) the measurement equation. In equation (22), $\mathbf{n}_i$ is the vector of random disturbance of the dynamic system and is usually modeled as white noise:

$$E[\mathbf{n}_i] = 0 \quad \text{and} \quad E[\mathbf{n}_i \mathbf{n}_j^t] = \delta_{ij} Q_i$$

Where $\delta_{ij} = 1$ if $i = j$ and 0 otherwise is the Kronecker symbol. The measurement $\mathbf{x}_i$ is corrupted by additive random noise, that is

$$E[\mathbf{x}_i] = \bar{\mathbf{x}}_i \quad \text{and} \quad E[(\mathbf{x}_i - \bar{\mathbf{x}}_i)(\mathbf{x}_i - \bar{\mathbf{x}}_i)^t] = \Lambda_{\mathbf{x}_i}$$

We assume also that there is no correlation between the noise process of the system and that of the observation, that is

$$E[(\mathbf{x}_i - \bar{\mathbf{x}}_i)\mathbf{n}_i^t] = 0$$

When $\mathbf{h}_i$ and $\mathbf{f}_i$ are linear functions, we write $\mathbf{s}_{i+1} = H_i \mathbf{s}_i + \mathbf{n}_i$ and $\mathbf{x}_i = F_i \mathbf{s}_i$, and the standard Kalman filter [Maybeck 1979] is directly available.

The performance of the Kalman filter in the linear case have been completely characterized as, for example, in [Maybeck 1979]. If we assume that the $\mathbf{n}_i$ and the $\mathbf{x}_i$ are Gaussian then, among *all* possible estimators, the Kalman filter provides the one with minimum variance, that is, which minimizes

$$E[(\mathbf{s}_i - \hat{\mathbf{s}}_i)^t (\mathbf{s}_i - \hat{\mathbf{s}}_i)]$$

If we do not assume Gaussianness, among all possible *linear* estimators (those that are computed as linear functions of the measurements), the Kalman filter also computes the one with minimum variance. Note that in this case there may exist nonlinear estimators that yield better results, that is, a smaller variance.

In that sense, in the case of linear state and measurement equations, the Gaussian assumption is unnecessary.

If $\mathbf{h}_i(\mathbf{s}_i)$ is not linear or if a linear relationship between $\mathbf{x}_i$ and $\mathbf{s}_i$ does not exist, the so-called *extended Kalman filter* (EKF) can be applied. The EKF approach is to apply the standard Kalman filter (for *linear* systems) to *nonlinear* systems with additive white noise by continually updating a *linearization* around the previous state estimate, starting with an initial guess. In other words, we only consider linear Taylor approximations of the state equation at the previous state estimate and of the measurement equation at the corresponding predicted state. This approach gives a simple and efficient algorithm to handle a nonlinear model. However, convergence to a reasonable estimate may *not* be achieved if the initial guess is poor or if the disturbances are so large that the linearization is inadequate to describe the system. Contrary to the linear case, there are no optimality results for the EKF; but note that the Gaussian assumption is also unnecessary.

The measurement equation $\mathbf{f}_i(\mathbf{x}_i, \mathbf{s}_i) = 0$ is first linearized to obtain a new measurement equation

$$\mathbf{y}_i = M_i \mathbf{s}_i + \mathbf{v}_i \tag{24}$$

where $\mathbf{y}_i$ is the new measurement vector, $\mathbf{v}_i$ is the noise vector of the new measurement, and M_i is the linearized transformation matrix. They are given by

$$M_i = \frac{\partial \mathbf{f}_i(\mathbf{x}_i, \mathbf{s}_i)}{\partial \mathbf{s}_i}$$

$$\mathbf{y}_i = -\mathbf{f}_i(\mathbf{x}_i, \mathbf{s}_i) + \frac{\partial \mathbf{f}_i(\mathbf{x}_i, \mathbf{s}_i)}{\partial \mathbf{s}_i} \mathbf{s}_i$$

$$E[\mathbf{v}_i] = 0$$

$$E[\mathbf{v}_i \mathbf{v}_i^t] = \frac{\partial \mathbf{f}_i(\mathbf{x}_i, \mathbf{s}_i)}{\partial \mathbf{x}_i} \Lambda_{\mathbf{x}_i} \frac{\partial \mathbf{f}_i(\mathbf{x}_i, \mathbf{s}_i)^t}{\partial \mathbf{x}_i} \triangleq V_i$$

The partial derivatives are evaluated at $\mathbf{s}_i = \hat{\mathbf{s}}_{i|i-1}$ and $\mathbf{x}_i = \bar{\mathbf{x}}_i$. The extended Kalman filter equations are given as follows:

Algorithm: Extended Kalman Filter

State prediction:

$$\hat{\mathbf{s}}_{i|i-1} = \mathbf{h}_i(\hat{\mathbf{s}}_{i-1})$$

Prediction of the state covariance matrix:

$$P_{i|i-1} = \frac{\partial \mathbf{h}_i}{\partial \mathbf{s}_i} P_{i-1} \frac{\partial \mathbf{h}_i^t}{\partial \mathbf{s}_i} + Q_{i-1}$$

Kalman gain matrix:

$$K_i = P_{i|i-1} M_i^t (M_i P_{i|i-1} M_i^t + V_i)^{-1}$$

Update of the state estimation:

$$\hat{\mathbf{s}}_i = \hat{\mathbf{s}}_{i|i-1} + K_i(\mathbf{y}_i - M_i \hat{\mathbf{s}}_{i|i-1})$$

$$= \hat{\mathbf{s}}_{i|i-1} - K_i \mathbf{f}_i(\bar{\mathbf{x}}_i, \hat{\mathbf{s}}_{i|i-1})$$

Update of the state covariance matrix:

$$P_i = (\mathbf{I} - K_i M_i) P_{i|i-1}$$

Initialization:

$$P_{0|0} = \Lambda_{\mathbf{s}_0} \qquad \hat{\mathbf{s}}_{0|0} = E[\mathbf{s}_0]$$

7 Formulation of Motion-Tracking Problem for EKF Approach

In this section, we formulate the motion-tracking problem in such a way that we can apply the extended Kalman filter formulation of the preceding section. The token is assumed to undergo a motion with constant angular velocity and constant translational acceleration (see theorem 2). We are given a sequence of stereo frames taken at $t_0, t_1, \ldots t_{i-1}, t_i, \ldots$, such that the interval between t_{i-1}, and t_i is constant and is denoted by Δt.

7.1 State Transition Equation

Let the angular velocity at time t_i be ω_i, the translational velocity $\mathbf{v}_i$ and the translational acceleration $\mathbf{a}_i$. Define the state vector as

$$\mathbf{s}_i = [\omega_i \ \mathbf{v}_i \ \mathbf{a}_i]^t \tag{25}$$

The state transition equation can be written as:

$$\mathbf{s}_i = H\mathbf{s}_{i-1} + \mathbf{n}_{i-1} \tag{26}$$

where

$$H = \begin{bmatrix} \mathbf{I}_3 & 0 & 0 \\ 0 & \mathbf{I}_3 & \mathbf{I}_3 \Delta t \\ 0 & 0 & \mathbf{I}_3 \end{bmatrix}$$

We then replace $\partial \mathbf{h}_i / \partial \mathbf{s}_i$ in the EKF Algorithm by H, since the transition function is linear. The $\mathbf{n}_{i-1}$ in equation (26) is the random disturbance, with

$$\mathrm{E}[\mathbf{n}_{i-1}] = 0 \qquad \text{and} \qquad \Lambda_{\mathbf{n}_{i-1}} = Q_{i-1}$$

$\mathbf{n}_i$ is used first to model noise due to, for example, vibration of objects during motion. But our constant-acceleration kinematic model is also in general only an approximation. By adding $\mathbf{n}_{i-1}$ in the dynamic model, we can partially take the approximation error into account.

7.2 Measurement Equations

As described in section 4, a segment S is represented by ψ and $\mathbf{m}$ and their covariance matrixes. Suppose a match $\{S_1, S_2\}$ is given, where S_1 occurs at time t_{i-1} and S_2 at time t_i. We define the measurement vector as

$$\mathbf{x} = [\psi_1^t \ \mathbf{m}_1^t \ \psi_2^t \ \mathbf{m}_2^t]^t \tag{27}$$

From theorem 2, we have the following equation:

$$\mathbf{m}_2 = W\mathbf{m}_1 + V\mathbf{v} + A\mathbf{a}$$

Let $\mathbf{u}_1$ be the unit direction vector of segment S_1 and $\mathbf{u}_2$ that of segment S_2. We have the following relation:

$$\mathbf{u}_2 = W\mathbf{u}_1$$

If we define two functions g and h to relate ψ and $\mathbf{u}$ together (see equations (2) and (3)) so that

$$\psi = g(\mathbf{u}) \qquad \text{and} \qquad \mathbf{u} = h(\psi) \tag{28}$$

then we have the following measurement equation

$$\mathbf{f}(\mathbf{x}, \mathbf{s}) = \begin{bmatrix} g(Wh(\psi_1)) - \psi_2 \\ W\mathbf{m}_1 + V\mathbf{v} + A\mathbf{a} - \mathbf{m}_2 \end{bmatrix} = \mathbf{0} \tag{29}$$

This is a 5-dimensional vector equation. In the following, the first two elements in $\mathbf{f}(\mathbf{x}, \mathbf{s})$ are denoted by $\mathbf{f}_1$ and the last three elements by $\mathbf{f}_2$.

The relation between $\mathbf{s}$ and $\mathbf{x}$ described by equation (29) is not linear. In order to apply the EKF algorithm, it is necessary to compute the derivatives of $\mathbf{f}(\mathbf{x}, \mathbf{s})$ with respect to $\mathbf{s}$ and $\mathbf{x}$. It is easy to show that

$$\frac{\partial \mathbf{f}}{\partial \mathbf{s}} = \begin{bmatrix} \dfrac{\partial \mathbf{f}_1}{\partial \omega} & \mathbf{0} & \mathbf{0} \\ \dfrac{\partial \mathbf{f}_2}{\partial \omega} & V & A \end{bmatrix} \tag{30}$$

$$\frac{\partial \mathbf{f}}{\partial \mathbf{x}} = \begin{bmatrix} \dfrac{\partial \mathbf{f}_1}{\partial \psi_1} & \mathbf{0} & -\mathbf{I}_2 & \mathbf{0} \\ \mathbf{0} & W & \mathbf{0} & -\mathbf{I}_3 \end{bmatrix} \tag{31}$$

where

$$\frac{\partial \mathbf{f}_1}{\partial \omega} = \frac{\partial g}{\partial \mathbf{u}_1'} \frac{\partial (W\mathbf{u}_1)}{\partial \omega}$$

$$\frac{\partial \mathbf{f}_2}{\partial \omega} = \frac{\partial (W\mathbf{m}_1)}{\partial \omega} + \frac{\partial (V\mathbf{v})}{\partial \omega} + \frac{\partial (A\mathbf{a})}{\partial \omega}$$

$$\frac{\partial \mathbf{f}_1}{\partial \psi_1} = \frac{\partial g}{\partial \mathbf{u}_1'} W \frac{\partial h}{\partial \psi_1}$$

with $\mathbf{u}_1 = h(\psi_1)$ and $\mathbf{u}_1' = W\mathbf{u}_1$. $\partial g/\partial \mathbf{u}_1'$ can be computed from formula (3). $\partial h/\partial \psi_1$ can be computed from formula (2). Since $\partial(W\mathbf{u}_1)/\partial\omega$ has the same formula as $\partial(W\mathbf{m}_1)/\partial\omega$, we then only need to compute $\partial(W\mathbf{u}_1)/\partial\omega$, $\partial(V\mathbf{v})/\partial\omega$, and $\partial(A\mathbf{a})/\partial\omega$.

After some simple computations, we get

$$\begin{aligned}
&\frac{\partial(W\mathbf{u})}{\partial\omega} \\
&= -\frac{\sin(\theta\,\Delta t)}{\theta}\,\tilde{\mathbf{u}} \\
&\quad + \frac{\theta\,\Delta t\cos(\theta\,\Delta t) - \sin(\theta\,\Delta t)}{\theta^3}\,(\tilde{\omega}\mathbf{u})\omega^t \\
&\quad + \frac{\theta\,\Delta t\sin(\theta\,\Delta t) - 2(1-\cos(\theta\,\Delta t))}{\theta^4}\,(\tilde{\omega}(\tilde{\omega}\mathbf{u}))\omega^t \\
&\quad + \frac{1-\cos(\theta\,\Delta t)}{\theta^2}\,[-\tilde{\tilde{\omega}}\mathbf{u} + (\omega\cdot\mathbf{u})\mathbf{I}_3 - \mathbf{u}\omega^t]
\end{aligned} \tag{32}$$

$$\begin{aligned}
&\frac{\partial(V\mathbf{v})}{\partial\omega} \\
&= -\frac{1-\cos(\theta\,\Delta t)}{\theta^2}\,\tilde{\mathbf{v}} \\
&\quad + \frac{\theta\,\Delta t\sin(\theta\,\Delta t) - 2(1-\cos(\theta\,\Delta t))}{\theta^4}\,(\tilde{\omega}\mathbf{v})\omega^t \\
&\quad + \frac{3\sin(\theta\,\Delta t) - \theta\,\Delta t\,(2+\cos(\theta\,\Delta t))}{\theta^5}\,(\tilde{\omega}(\tilde{\omega}\mathbf{v}))\omega^t \\
&\quad + \frac{\theta\,\Delta t - \sin(\theta\,\Delta t)}{\theta^3}\,[-\tilde{\tilde{\omega}}\mathbf{v} + (\omega\cdot\mathbf{v})\mathbf{I}_3 - \mathbf{v}\omega^t]
\end{aligned} \tag{33}$$

$$\begin{aligned}
&\frac{\partial(A\mathbf{a})}{\partial\omega} \\
&= -\frac{\theta\,\Delta t - \sin(\theta\,\Delta t)}{\theta^3}\,\tilde{\mathbf{a}} \\
&\quad + \frac{3\sin(\theta\,\Delta t) - \theta\,\Delta t\,(2+\cos(\theta\,\Delta t))}{\theta^5}\,(\tilde{\omega}\mathbf{a})\omega^t \\
&\quad + \frac{4(1-\cos(\theta\,\Delta t)) - (\theta\,\Delta t)^2 - \theta\,\Delta t\sin(\theta\,\Delta t)}{\theta^6} \\
&\quad \times (\tilde{\omega}(\tilde{\omega}\mathbf{a}))\omega^t \\
&\quad + \frac{(\theta\,\Delta t)^2 - 2(1-\cos(\theta\,\Delta t))}{2\theta^4} \\
&\quad \times [-\tilde{\tilde{\omega}}\mathbf{a} + (\omega\cdot\mathbf{a})\mathbf{I}_3 - \mathbf{a}\omega^t]
\end{aligned} \tag{34}$$

where $\omega\cdot\mathbf{u}$ denotes the inner product of the two vectors ω and $\mathbf{u}$.

When a token matches a segment in the current frame, we use the above formalism to update its kinematic parameters. The same process is applied to each token.

8 Matching Segments

In this section, we describe how to match a token being tracked to a segment in the current frame. The matching technique is based on the Mahalanobis distance, which can be considered as an Euclidean distance in parameter space weighted by uncertainty.

8.1 Prediction of a Token

Let $\mathbf{z} = [\psi^t, \mathbf{m}^t]^t$ be the parameter vector of the token being tracked. The token kinematic parameters are $[\omega^t, \mathbf{v}^t, \mathbf{a}^t]^t$. We can use them to *predict* its parameter vector $\hat{\mathbf{z}} = [\hat{\psi}^t, \hat{\mathbf{m}}^t]^t$ at the next time instant:

$$\begin{cases} \hat{\psi} = g(Wh(\psi)) \\ \hat{\mathbf{m}} = W\mathbf{m} + V\mathbf{v} + A\mathbf{a} \end{cases} \tag{35}$$

where the functions g and h are defined as in equation (28) in section 7. Due to noise from multiple sources, it is very unlikely that a segment can be found with exactly the parameters $\hat{\mathbf{z}} = [\hat{\psi}^t, \hat{\mathbf{m}}^t]^t$ and we have to design a matching strategy.

8.2 Matching Criterion

Let $\{\ldots, [\psi_i^t, \mathbf{m}_i^t], \ldots\}$ be the set of observed segments in the scene and $[\hat{\psi}^t, \hat{\mathbf{m}}^t]^t$ be the expected segment. All segments have their measures of uncertainty attached (covariance matrixes): $\{\ldots, \Lambda_i, \ldots\}$ and Λ_{token}. Λ_{token} is the covariance matrix of the predicted parameters $\hat{\mathbf{z}} = [\hat{\psi}^t, \hat{\mathbf{m}}^t]^t$ of the token being tracked, whose computation is given below. The *Mahalanobis distance* between the expected segment to each segment in the current frame is then given by

$$d_i^M = \mathbf{r}_i^t \Lambda_{\mathbf{r}_i}^{-1} \mathbf{r}_i \tag{36}$$

where

$$\mathbf{r}_i = \begin{bmatrix} \psi_i - \hat{\psi} \\ \mathbf{m}_i - \hat{\mathbf{m}} \end{bmatrix}$$

$$\Lambda_{\mathbf{r}_i} = \Lambda_i + \Lambda_{\text{token}}$$

The variable d_i^M is a scalar random variable following a χ^2 distribution with 5 degrees of freedom.

By looking up the χ^2 distribution table, we can choose an appropriate threshold ϵ on the Mahalanobis distance. For example, we can take $\epsilon = 11.07$ which corresponds to a probability of 95% to have the distance d_i^M less than ϵ if the match is correct. Thus segments in the current frame can be considered as *plausible matches*, if they verify the inequality:

$$d_i^M < \epsilon \tag{37}$$

Using the above technique, a token may have multiple matches in the current frame. This problem has been described in section 3.

Before we compute the Mahalanobis distance (equation (36)), we must take care of the discontinuity of ϕ when a segment is nearly parallel to the plane $y = 0$ (see section 4). The idea is the following. If a segment is represented by $\psi = [\phi, \theta]^t$, it is also represented by $[\phi - 2\pi, \theta]^t$. Therefore, when comparing the representations of two segments S and S', we perform the following tests and actions. If $\phi < \pi/2$ and $\phi' > 3\pi/2$, then set ϕ' to be $\phi' - 2\pi$; else if $\phi > 3\pi/2$ and $\phi' < \pi/2$, then set ϕ to be $\phi - 2\pi$; else do nothing. Notice that adding a constant to a random variable does not affect its covariance matrix.

Note that the fact that ϕ is discontinuous is only an artifact of the representation used. The right way to avoid this kind of problems is to use the notions of manifold and maps [Faugeras 1991] which are outside the scope of this paper.

Now we return to the problem of computing Λ_{token}. Given the original parameters of the token $[\psi^t, \mathbf{m}^t]^t$ and their covariance matrix $\Lambda_{(\psi,\mathbf{m})}$, and given the kinematic parameters $[\omega^t, \mathbf{v}^t, \mathbf{a}^t]^t$ and their covariance matrix $\Lambda_{(\omega,\mathbf{v},\mathbf{a})}$. Under the first order approximation, the covariance matrix of the expected segment is given by

$$\Lambda_{\text{token}} = J_{(\psi,\mathbf{m})}\Lambda_{(\psi,\mathbf{m})}J_{(\psi,\mathbf{m})}^t + J_{(\omega,\mathbf{v},\mathbf{a})}\Lambda_{(\omega,\mathbf{v},\mathbf{a})}J_{(\omega,\mathbf{v},\mathbf{a})}^t \tag{38}$$

From equation (35), the Jacobian matrix with respect to $[\psi^t, \mathbf{m}^t]^t$ is

$$J_{(\psi,\mathbf{m})} = \begin{bmatrix} \dfrac{\partial g}{\partial \mathbf{u}'} W \dfrac{\partial h}{\partial \psi} & \mathbf{0} \\ \mathbf{0} & W \end{bmatrix} \tag{39}$$

and the Jacobian matrix with respect to $[\omega^t, \mathbf{v}^t, \mathbf{a}^t]^t$ is

$$J_{(\omega,\mathbf{v},\mathbf{a})} = \begin{bmatrix} \dfrac{\partial g}{\partial \mathbf{u}'} \dfrac{\partial (W\mathbf{u})}{\partial \omega} & \mathbf{0} & \mathbf{0} \\ \dfrac{\partial (W\mathbf{m})}{\partial \omega} + \dfrac{\partial (V\mathbf{v})}{\partial \omega} + \dfrac{\partial (A\mathbf{a})}{\partial \omega} & V & A \end{bmatrix} \tag{40}$$

where $\mathbf{u} = h(\psi)$ and $\mathbf{u}' = Wh(\psi)$. All derivatives in the above equations are computed as in section 7.

8.3 Reducing the Complexity by Bucketing Technique

Although the complexity of matching one segment is linear in the number of segments present in the current frame, the matching process may be slow, especially when there is a large number of segments. This is because the computation of the Mahalanobis distance is relatively expensive (it involves the inversion of a 5×5 matrix). If we can compute the distances d_i^M of the expected segment to only a subset of segments which are near the expected one, we can considerably speed up the matching process; this can be achieved by the use of *bucketing techniques*; which are now standard in computational geometry [Preparata & Shamos 1986].

We can apply the bucketing techniques either in 3D space or in 2D space. Bucketing in the image plane of one camera is preferred because it is cheaper. The image plane is partitioned into m^2 square windows (buckets) W_{ij} (in our implementation, $m = 16$). To each window W_{ij} we attach the list of segments $\{S_k\}$ intersecting W_{ij}. The key idea of bucketing is that on the average the number of segments intersecting a bucket is much smaller, and in practice constant, than the total number of segments in the frame (see for example [Faugeras et al., 1990] for details). Given a predicted token to be matched, we first compute the buckets which the disk defined by the predicted segment intersects. The disk is defined as follows: its center coincides with the midpoint of the predicted segment and its diameter equals its length plus a number corresponding to the projected uncertainty of its midpoint. Again, the idea is that, on the average, this disk will intersect a small number of buckets, except for a token which just appeared. Since we initialize such a token with a

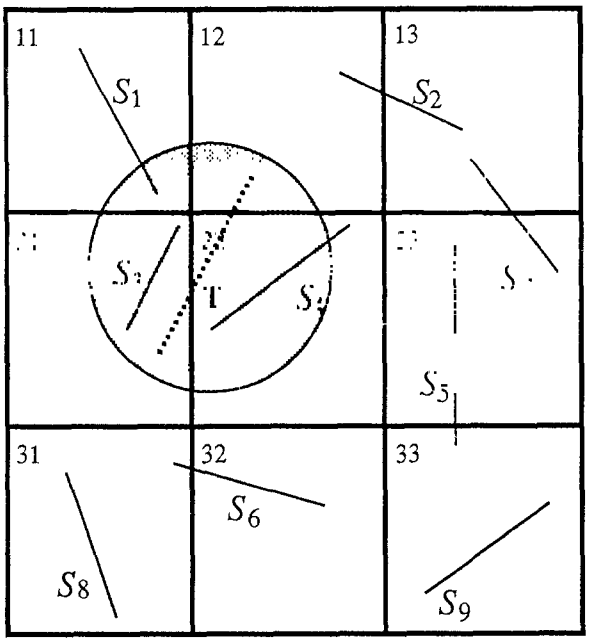

Fig. 6. Bucketing technique.

big uncertainty in motion, its corresponding disk is quite big and may intersect many buckets. The set of potential matching candidates is then the union of the lists of segments intersecting these buckets. This set is now considerably reduced, on the average. Figure 6 illustrates the bucketing technique. In the figure, **T** is the expected segment. Thanks to the bucketing technique, it finds 4 potential matching candidates $\{S_1, S_2, S_3, S_4\}$ instead of 9 in total. After computing the matching criterion described in the above section, there remain only two candidates S_3 and S_4. Note that the computation of buckets can be performed very quickly by an algorithm whose complexity is linear in the number of segments in the current frame.

9 Support of Existence

In this section we describe in detail how the beam-search strategy sketched in section makes tracking much more robust by allowing multiple matches.

Indeed, in practice, a token being tracked may find several correspondences in the current frame. The most common strategy is to choose the nearest segment (as in Crowley et al. [1988]; Deriche & Faugeras [1990]) and to discard the other possibilities.

Our implementation is based on the work of Bar-Shalom and Fortmann [1988] and is much less sensitive to false matches. The idea is to keep open the possibility of accepting several or no matches for any given token. But, if tokens never disappear we may rapidly reach a computational explosion. To avoid this we compute for each token a number that we call its support of existence which measures the adequateness of the token with the measurements: if the token has not found any correspondences in a long time then it is bound either to be the result of a false match that happened in the past or to have disappeared from the scene.

We use the notations of section 8 and denote the sequence of measurements corresponding to the token being tracked up to time t_k as $Z^k \triangleq \{\mathbf{z}(t_1), \ldots, \mathbf{z}(t_k)\}$ in which $\mathbf{z}(t_i) = [\psi^t(t_i), \mathbf{m}^t(t_i)]^t$ is the parameter vector of the segment observed at time t_i. Denote the event that Z^k yields a correct token, that is, that its components $\mathbf{z}(t_i)$ were produced by the same segment moving in space, by $e \triangleq \{Z^k \text{ yields a correct token}\}$. The likelihood function of this sequence yielding a correct token is the joint probability density function (or PDF):

$$L^k(e) = p(Z^k|e) = p[\mathbf{z}(t_1), \ldots, \mathbf{z}(t_k)|e] \tag{41}$$

From the definition of a conditional PDF, $L^k(e)$ can be written as

$$\begin{aligned} L^k(e) &= p[Z^{k-1}, \mathbf{z}(t_k)|e] \\ &= p[\mathbf{z}(t_k)|Z^{k-1}, e]\, p[Z^{k-1}|e] \\ &= \prod_{i=1}^{k} p[\mathbf{z}(t_i)|Z^{i-1}, e] \end{aligned} \tag{42}$$

where Z^0 represents the prior information.

As in section 8, we denote the measurement residual as $\mathbf{r}$, that is, $\mathbf{r}_i = \mathbf{z}(t_i) - \hat{\mathbf{z}}(t_i)$. Then $p(\mathbf{r}_i) = N[\mathbf{r}_i; \mathbf{0}, \mathbf{\Lambda}_{\mathbf{r}_i}]$ with $\mathbf{\Lambda}_{\mathbf{r}_i} = \mathbf{\Lambda}_i + \mathbf{\Lambda}_{\text{token}}$. We use $N[\mathbf{x}; \bar{\mathbf{x}}, \mathbf{\Lambda}]$ to denote the Gaussian density function of the random variable $\mathbf{x}$ with mean $\bar{\mathbf{x}}$ and covariance $\mathbf{\Lambda}$. We now make the admittedly strong assumption that the $\mathbf{r}_i$ are Gaussian and uncorrelated. We thus write:

$$p[\mathbf{z}(t_i)|Z^{i-1}, e] = N[\mathbf{r}_i; \mathbf{0}, \mathbf{\Lambda}_{\mathbf{r}_i}] \tag{43}$$

It follows that under the previous assumption:

$$L^k(e) = \left[\prod_{i=1}^{k} |2\pi\mathbf{\Lambda}_{\mathbf{r}_i}|^{-1/2}\right] \exp\left[-\frac{1}{2}\sum_{i=1}^{k} \mathbf{r}_i^t \mathbf{\Lambda}_{\mathbf{r}_i}^{-1} \mathbf{r}_i\right]$$

Note that $\mathbf{r}_i^t \mathbf{\Lambda}_{\mathbf{r}_i}^{-1} \mathbf{r}_i = d_i^M$ (see equation (36)). The modified log-likelihood function, corresponding to the exponent of $L^k(e)$, is defined as

$$l_k \triangleq -2 \ln \left[\frac{L^k(e)}{\prod_{i=1}^{k} |2\pi \Lambda_{\mathbf{r}_i}|^{-1/2}} \right] = \sum_{i=1}^{k} d_i^M$$

and can be computed recursively as follows:

$$l_k = l_{k-1} + d_k^M$$

The last term has a χ^2 distribution with $n_r = 5$ degrees of freedom. Since the $\mathbf{r}_k$ are assumed to be independent, l_k has a χ^2 distribution with kn_r degrees of freedom.

The statistical test for deciding that Z^k yields a correct token is that the log-likelihood function satisfies

$$l_k \leq \kappa \tag{44}$$

where the threshold κ is obtained from the χ^2 table with kn_r degrees of freedom by setting $\Pr(\chi^2_{kn_r} \leq \kappa) = \alpha$, where α is typically equal to 95%.

In practice, the test (44) cannot be used for long sequences because the likelihood function is dominated by old measurements and responds very slowly to recent ones. In order to limit the "memory" of system, we can multiply the likelihood function at each step by a discount factor $c < 1$. This results in the fading-memory likelihood function:

$$l_k = cl_{k-1} + d_k^M = \sum_{i=1}^{k} c^{k-i} d_i^M$$

The effective memory of l_k is now $(1 - c)^{-1}$, and in steady state l_k is approximately a χ^2 random variable with $n_r(1 + c)/(1 - c)$ degrees of freedom, mean $n_r/(1 - c)$, and variance $2n_r/(1 + c^2)$. See Bar-Shalom and Fortmann [1988] for the proof. In our implementation, $c = 0.75$.

In the above discussion, we assume implicitly that a match is detected at each sampling time. As described earlier, match detection may fail from time to time for a number of reasons. These failures mean that

$$d_k^M \geq \epsilon$$

as described in section 8. Thus, if at time t_k no match if found, the fit between the prediction and the observation is not very good. But note that even in that case we may still have $l_k \leq \kappa$ and the processing of the token will continue. This allows us to cope with problems such as occlusion, disappearance, and absence. Of course if the Mahalanobis distances stay over the threshold ϵ at too many consecutive time instants, that is, if the token too often does not find any good match in the scene, then l_k will go beyond the threshold κ, and the token will be discarded, as expected. In practice we set $d_k^M = \alpha\epsilon$ where $\alpha = 1.2$ in our implementation.

10 Grouping Tokens into Objects

In the previous sections, we have described how to track each token and estimate its kinematic parameters in parallel. In this section, we address the following problem: how can the supervisor group tokens into objects based on their kinematic parameters?

We assume that moving objects in the scene are rigid. From section 5, we know that each token belonging to a rigid object must have the same kinematic parameters with respect to a *common* point. In our algorithm, the kinematic parameters of all tokens are computed with respect to the same point—the origin of the system of reference. We can then define an object as follows: *an object is a set of tokens with the same kinematic parameters*. Under this definition, two different objects are grouped as a single one if they undergo the same motion.

Of course, the estimated kinematic parameters are uncertain, and have attached to them an uncertainty measure—their covariance matrix. One cannot expect to find two tokens having exactly the same kinematic parameters. The Mahalanobis distance is again used in this case to measure the discrepancy between kinematic parameters. Given two kinematic parameter vectors $\mathbf{s}_1$ and $\mathbf{s}_2$ and their covariance matrices $\Lambda_{\mathbf{s}_1}$ and $\Lambda_{\mathbf{s}_2}$, their Mahalanobis distance is given by

$$\delta^M = [\mathbf{s}_1 - \mathbf{s}_2]^t (\Lambda_{\mathbf{s}_1} + \Lambda_{\mathbf{s}_2})^{-1} [\mathbf{s}_1 - \mathbf{s}_2] \tag{45}$$

When $\mathbf{s} = [\omega, \mathbf{v}, \mathbf{a}]^t$, δ^M is a random scalar following a χ^2 distribution with 9 degrees of freedom. By looking up the χ^2 distribution table, we can choose an appropriate threshold ε on the Mahalanobis distance. For example, we can take $\varepsilon = 16.92$ which corresponds to a probability of 95% that the distance δ^M is less than ε. Thus two tokens can be considered to belong to the same object, if the Mahalanobis distance of their kinematic parameters verifies the inequality

$$\delta^M < \varepsilon \tag{46}$$

If two tokens are identified to belong to a single object by the above test, we can compute a better estimate $\mathbf{s}$ of the kinematic parameters for the object from those attached to each token by the Kalman filter or simply by the modified Kalman minimum-variance estimator:

$$\mathbf{s} = \Lambda_s^{-1}(\Lambda_{s_1}^{-1}\mathbf{s}_1 + \Lambda_{s_2}^{-1}\mathbf{s}_2)$$
$$\Lambda_s = (\Lambda_{s_1}^{-1} + \Lambda_{s_2}^{-1})^{-1} \quad (47)$$

where Λ_s is the covariance matrix of the new estimate s. We then try to find more tokens compatible with s and Λ_s and update them by equation (47). When all tokens have been processed, we get

— *an object* with a list of tokens associated with it and its kinematic parameters, and
— *a list of tokens* not yet attached to an object.

The same process is then performed on the list of non-identified tokens to recover new objects, and so forth, until no more objects can be identified. We finally obtain all plausible objects in the scene.

11 Experimental Results

We have implemented the proposed algorithm on a SUN workstation using the C language. Our program can display and control interactively the motion-tracking process. In this section, we provide two experimental results with real data to show the performance of our algorithm. Results with synthetic data have been reported in [Zhang & Faugeras 1990b]. The angular velocity unit is radians/unit-time, and the translational velocity unit is millimeters/unit-time, except when stated otherwise.

11.1 Real Data with Controlled Motion

The data are acquired as follows: the trinocular stereo rig is in front of a rotating table (at about 2.5 meters). The rotating table has two degrees of freedom: vertical translation and rotation around the vertical. The motion is controlled manually. Boxes are put on the table and will be considered by the program as an object. The table undergoes a general motion, combination of a rotation and a translation parallel to the rotation axis. It rotates 3 degrees (clockwise, if viewed from the top), and at the same time, has translation from bottom to top of 50 millimeters between adjacent frames. Thus, the motion corresponds exactly to the constant-velocity model. There are in fact three "objects" in the scene (see figure 7): the second is the static support of the table, and the third is what is below and attached to the rotating table, and undergoes the same translation but no rotation.

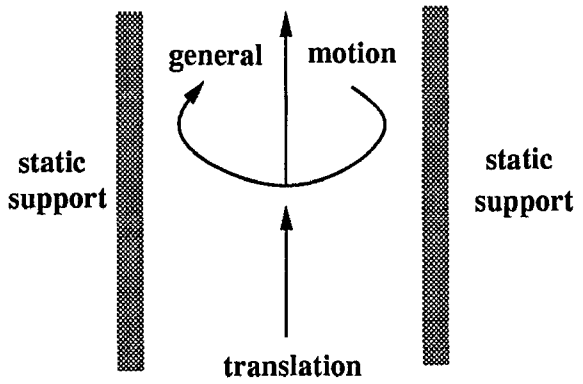

Fig. 7. Description of the rotating table undergoing general motion.

Ten 3D frames have been acquired for this experiment. Each frame contains about 130 3D line segments. Figure 8 displays the images taken by the first camera at t_1 and t_{10}. To show the motion, the first and the second 3D frames are superposed in figure 9 and the first and the last in figure 10, together with a pair of stereograms. We find that the data are very noisy even for the static object and that occlusion, appearance, disappearance and absence problems are very severe. We can also observe that the object in translation is not detected until the third frame is taken. (*Note:* In figures 9 through 18 when there are four pictures, the upper left one is the front view of a 3D frame—that is, its orthographic projection onto a frontoparallel plane—the upper right one is the top view—that is, its orthographic projection onto a horizontal plane—and the lower left and right ones form a stereogram to allow the reader to perceive 3D information by cross-eye fusion. If there are only two pictures, they are a pair of stereograms.)

Each segment in the first frame is initialized as a token to be tracked. Since the motion-tracking algorithm is recursive, some a priori information on the kinematics is required. A reasonable assumption may be that objects do not move, as the interframe motion is expected to be small. The kinematic parameters are thus all initialized to zero, but with fairly large uncertainty: the standard deviation for each angular velocity component is 0.0873 radians/unit-time, and that for each translational velocity component is 150 millimeters/unit-time. The variances on the acceleration components are set to zero (no acceleration).

Those tokens are then predicted for the next instant t_2 and the predicted tokens are compared with those in the new frame. Of course, since we have assumed no motion, the predicted position and orientation of

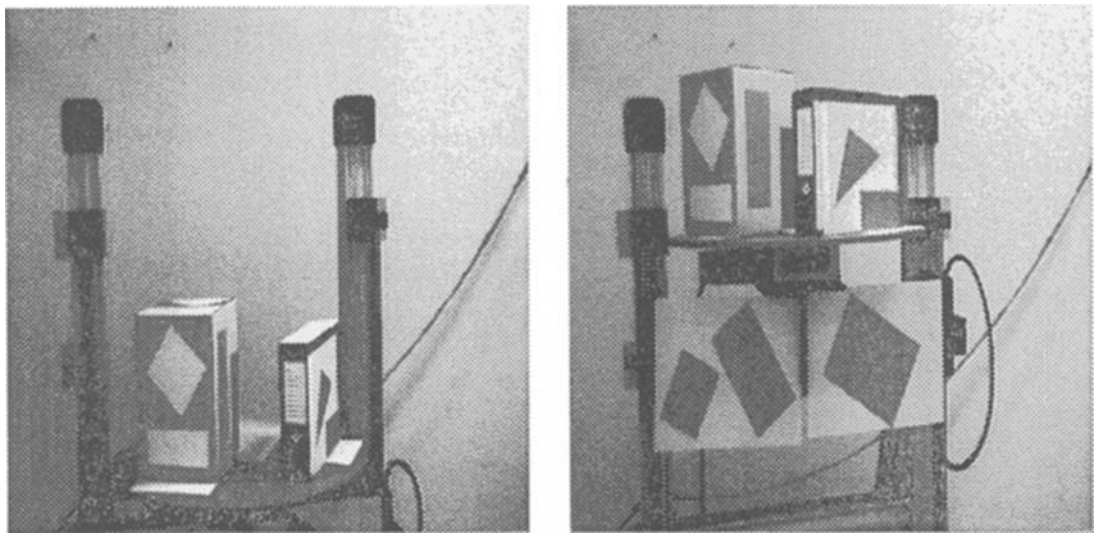

Fig. 8. The images taken by the first camera at t_1 and t_{10}.

each token remains unchanged, but its uncertainty changes and becomes very large. Thus figure 9 displays also the difference between the predicted tokens (in solid lines) and the observed line segments (in dashed lines) at time t_2. As expected, multiple matches occur for most of the tokens. Techniques based only on the best match usually fail at this stage, since the nearest segment is not always the correct match. We retain the two best matches if a token has multiple matches. The token updates its kinematic parameters using its best match. The supervisor initializes a new token by combining the token and its second-best match which is used to estimate its kinematic parameters.

We continue the tracking in the same manner. Figure 11 shows the superposition of the predicted segments for t_3 and the observed segments. As can be observed, more active tokens (in solid lines) exist at this moment: some have been activated due to multiple matches at time t_2 and some just entered the field of view. We observe that segments belonging to the object undergoing general motion coincide well, and so do the segments of the static support. The segments belonging to the translating object do not coincide well, because they just appeared in the field of view and no information about their kinematics is available. Figure 12 shows the superposition between the predicted segments for t_{10} and the observed segments. Almost all segments are well superposed except those in the middle part. Those segments correspond to the outline of the rotating table which is an ellipse. Using line segments to approximate an ellipse is of course difficult, and this is clearly one of the limitations of our system. One can notice that most of the tokens, due to false matching in the preceding instants, have disappeared, because they have lost their supports of existence.

The supervisor successfully segments the scene into 3 groups of objects based on the information of the kinematics of each token. In the following, we show the result of the estimated kinematic parameters of the object undergoing general motion. If we describe the motion in the stereo coordinate system, the kinematic parameter vector is $\mathbf{s} = [0.0, 5.236\text{e} - 02, 0.0, ?, -50, ?]^t$. The x and z components of the translational velocity are not known, since because of the calibration method we have used, the rotation axis (that is, the axis of the cylinder) is not known in the stereo coordinate system. We know the y component of the translational velocity because the rotation axis is parallel to the y axis. Figure 13 shows the variation of the absolute errors in the estimation of the angular velocity. Figure 14 shows only the error in the estimation of the y component of the translational velocity. The results are good: after a few (four or five) frames, the error for the angular velocity is less than 2.5 milliradians/unit-time (compared with 52.36), and the error for the translational velocity is less than 1.5 millimeters/unit-time (compared with 50). The final estimation for this object is

$$\hat{\mathbf{s}} = [4.291\text{e} - 04, 5.328\text{e} - 02, -7.059\text{e} - 04, -7.387\text{e} + 00, -4.951\text{e} + 01, 3.677\text{e} + 01]^t$$

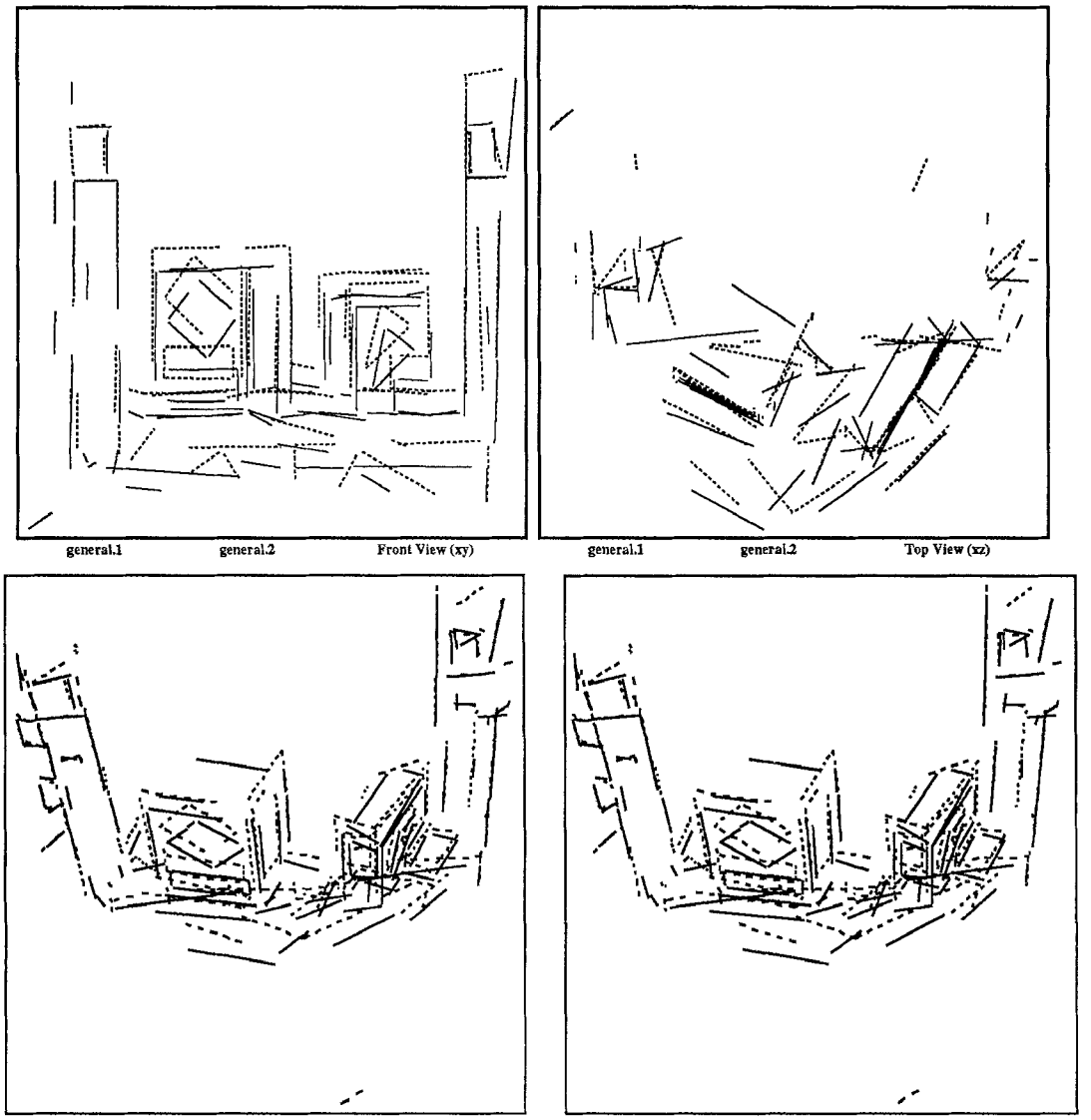

Fig. 9. The superpositon of the first (in solid lines) and the second (in dashed lines) frames.

The relative error of the angular velocity is 2.36%. The angle between the true and estimated axes of rotation is 0.9 degrees. Similar results are observed for the other objects. The final motion estimate corresponding to the translating object is

$$[3.437e-04,\ 1.353e-03,\ -7.303e-04,$$
$$-2.009e-01,\ -4.937e+01,\ 8.361e-01]^t$$

which should be compared with [0, 0, 0, 0, −50, 0]. The final motion estimate of the static object is

$$[-2.321e-04,\ 5.009e-04,\ 2.981e-04,$$
$$2.681e-02,\ -3.399e-01,\ 4.799e-01]^t$$

We now show that the position of the axis of the rotating table in the stereo coordinate system can be computed from the estimated kinematics $\hat{\mathbf{s}}$ and the known nature of the motion. For a point not on that axis, its trajectory is a helix. For a point on that axis, its trajectory is a straight line. Let a point on the axis be $\mathbf{p}$. From equation (11), the velocity of $\mathbf{p}$ is given by

$$\mathbf{v}_\mathbf{p} = \mathbf{v} + \omega \times \mathbf{p}$$

Since the rotation is about the axis of the table, the velocity $\mathbf{v}_\mathbf{p}$ of $\mathbf{p}$ must be parallel to the rotation axis ω, that is $\omega \times \mathbf{v}_\mathbf{p} = 0$. We thus have

$$\omega \times \mathbf{v} + \omega \times (\omega \times \mathbf{p}) = 0 \qquad (48)$$

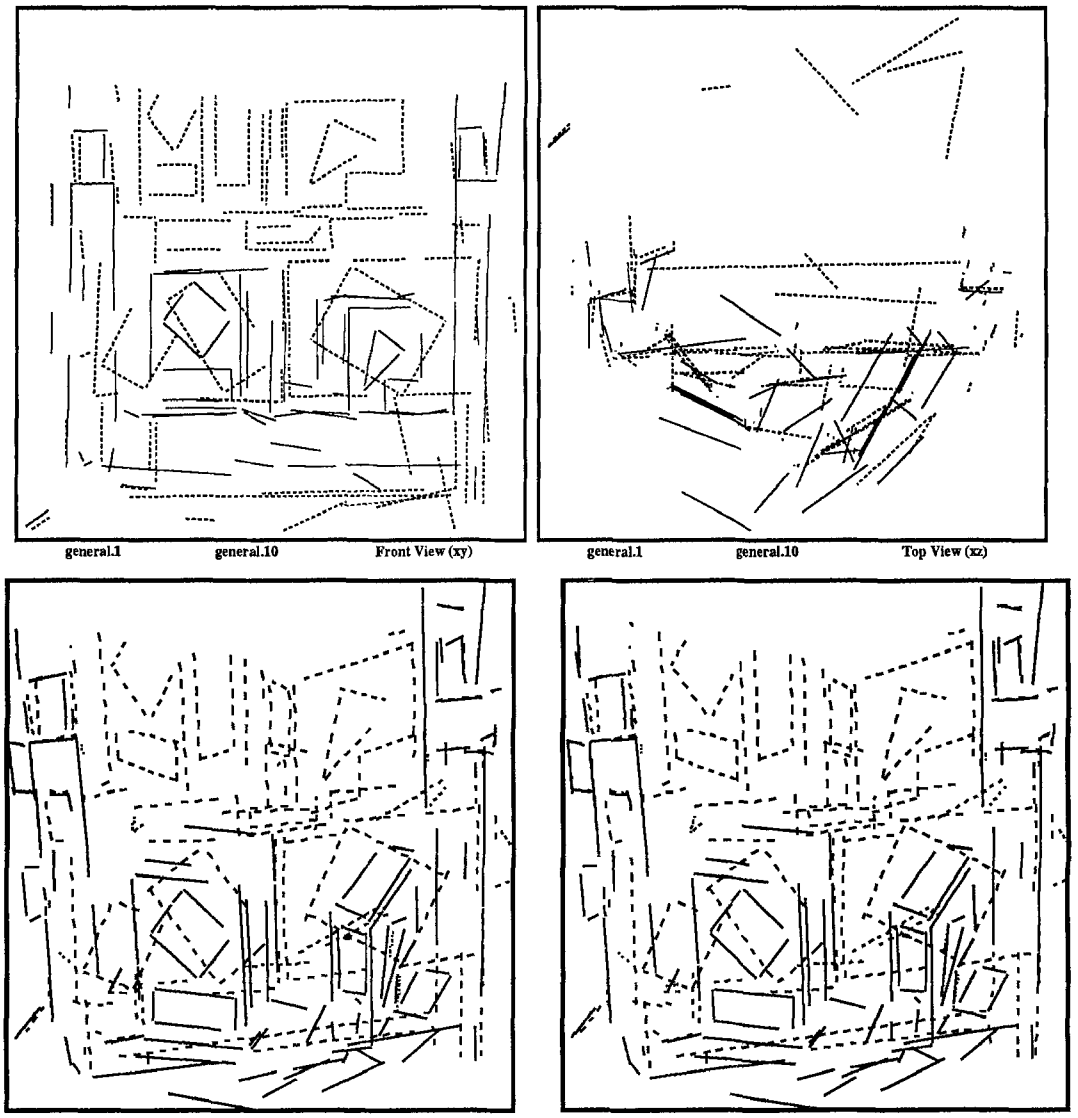

Fig. 10. The superposition of the first (in solid lines) and the last (in dashed lines) frames.

Equation (48) gives only two independent scalar equations, because the three components of a cross-product are not linearly independent. It defines the axis of the table. In order to compute a point on it, we can choose the y component p_y of $\mathbf{p}$ equal to $\mathbf{0}$, and compute its x and z components: p_x and p_z. In this example, we get $p_x = 677.68$ and $p_z = 131.08$. That point is shown at the intersection of the two bold line segments in the top view of figure 12, and seems roughly correct.

11.2 Real Data with Uncontrolled Motion

The data described in this section are acquired as follows. The trinocular stereo rig is mounted on a mobile vehicle, which moves in the laboratory. Objects in the scene are about 2 to 7 meters away from the mobile vehicle. We have taken 15 stereo views while the vehicle was supposed to undergo a translation. Figure 15 displays the images taken by the first camera at t_1 and

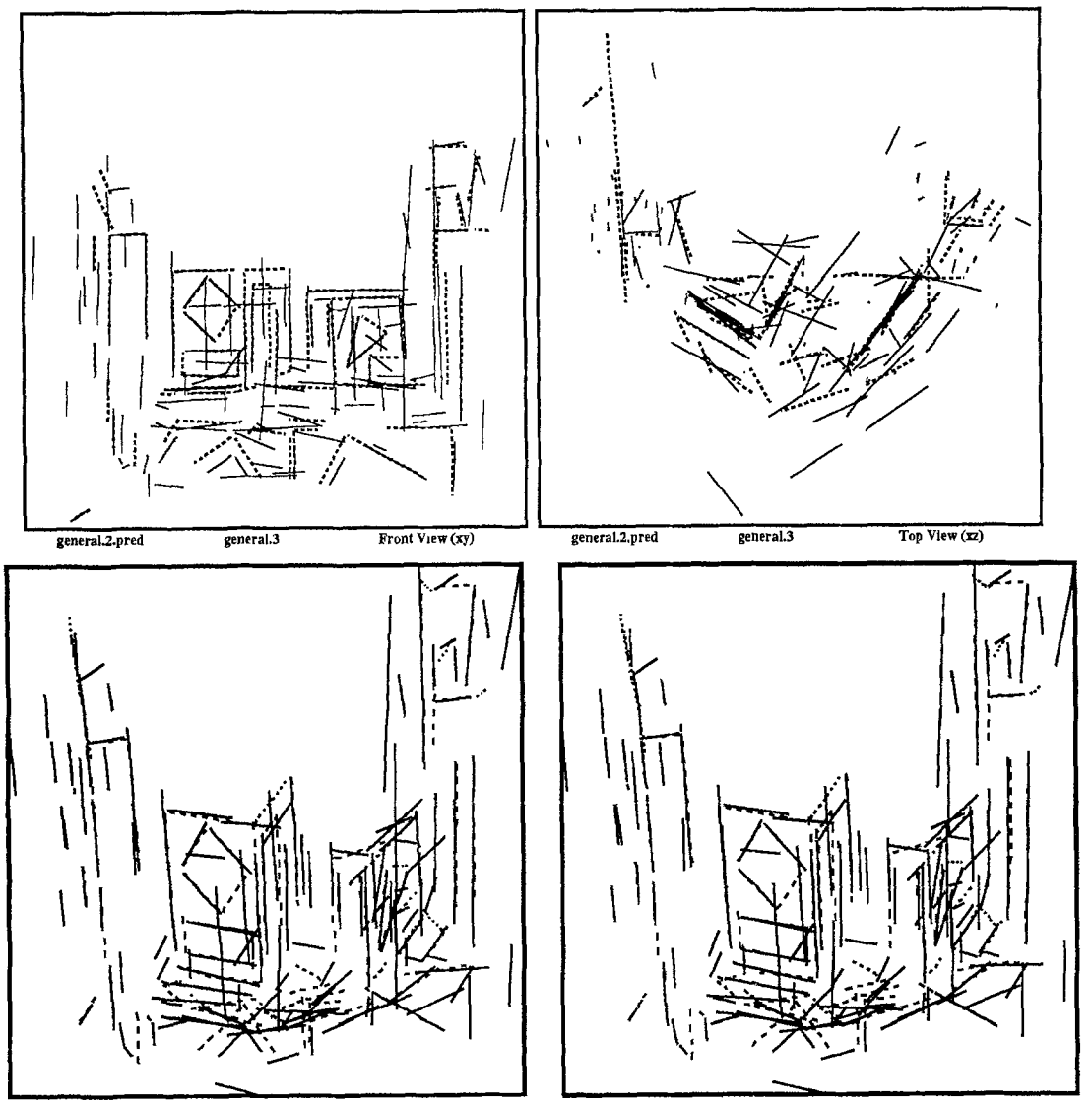

Fig. 11. The superposition of the predicted (in solid lines) and the observed (in dashed lines) segments at time t_3

t_{15}. We have thus reconstructed 15 3D frames using our stereo system, each containing around 85 line segments.

Ideally, the motion of the vehicle is a pure translation. If described in our stereo coordinate system, the translation between successive views is about $[-25, 0, -98]^t$, that is, a displacement of 100 millimeters. However, due to precision of the mechanical system and slipping of the vehicle wheels, it is naive to believe that the vehicle motion is going to be exactly the requested one. To check this, we have applied an algorithm developed earlier for computing motion between two 3D frames [Zhang et al. 1988; Zhang & Faugeras 1992] to every successive 3D view. We have found that there is a random rotation of about one degree between successive views and a difference in translation of up to 30 millimeters in x or z direction. To give an idea, we show in table 1 the x and z components of the computed motions (the sign is omitted). We can remark that the motion realized by the mobile vehicle is not very smooth and does not satisfy either the constant velocity or the constant acceleration model. This is our ground truth data for checking the performances of the methods presented in this paper.

We apply the techniques described in this paper to the above data. As in the previous experiment, each

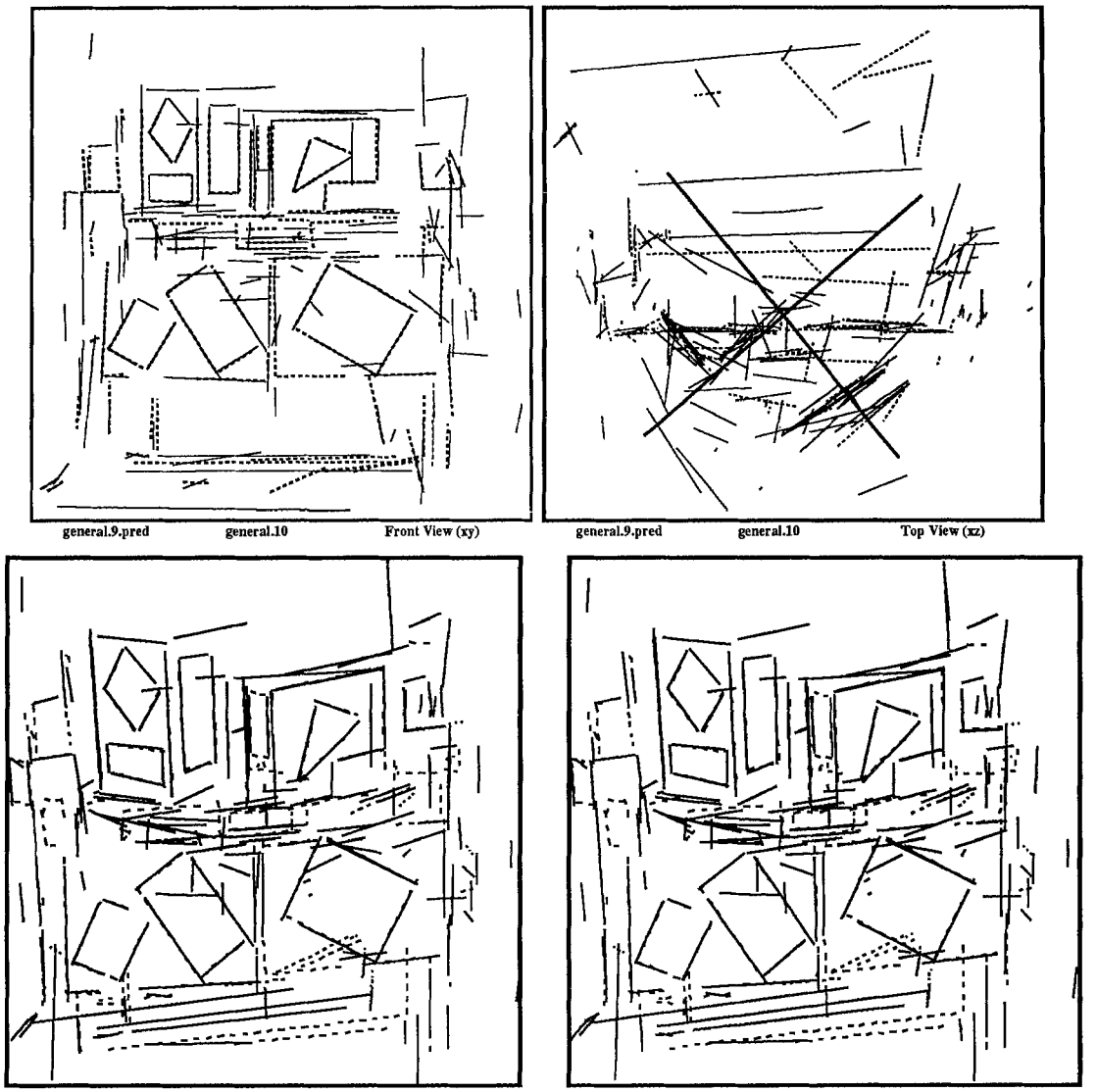

Fig. 12. The superposition of the predicted (in solid lines) and the observed (in dashed lines) segments at time t_{10}.

segment in the first frame is initialized as a token to be tracked. The kinematic parameters are all initialized to zero, but with fairly large uncertainties: the standard deviation is 0.0873 radians/unit-time for each angular velocity component, and 150 millimeters/unit-time for each translational velocity component. The variances on the acceleration components are set to zero (no acceleration). The only difference is that the noise term $\mathbf{n}_{i-1}$ in the state equation (equation (26)) is not zero. We have added at each step a noise with standard deviation 0.5 degrees/unit-time to the ω components and a noise with standard deviation 2 millimeters/unit-time to the $\mathbf{v}$ components.

The position and orientation of those tokens are then predicted for the next instant t_2 and the predicted tokens are compared with those in the new frame. Of course, since no motion is assumed, the predicted position and orientation of each token remains unchanged, but its uncertainty changes and becomes very large. Figure 16 displays a stereogram showing the difference between the predicted tokens (in solid lines) and the observed line segments (in dashed lines) at t_2. The reader can perceive the 3D information by cross-eye fusion. As expected, multiple matches occur for most of tokens, which are treated in the same way as in the previous experiment.

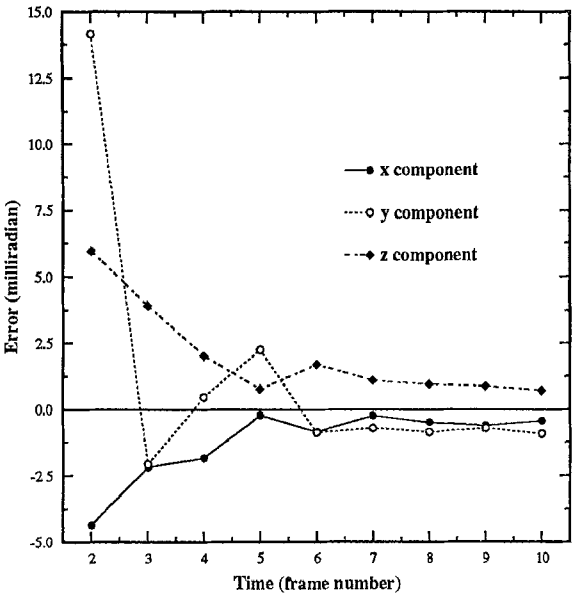

Fig. 13. Error evolution in the angular velocity of the rotating table undergoing general motion.

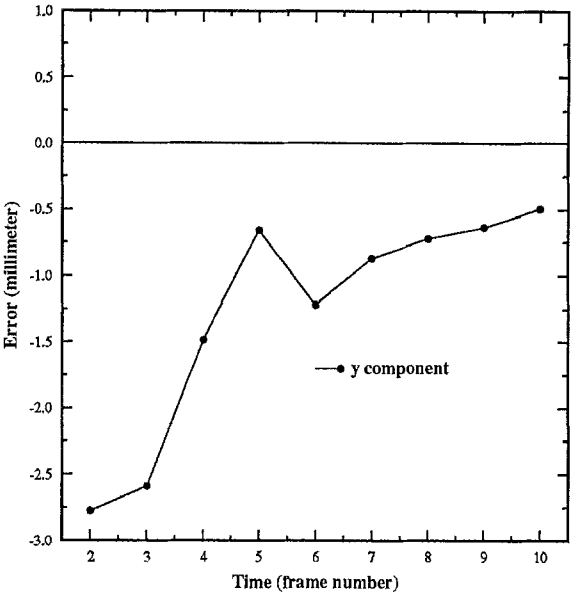

Fig. 14. Error evolution in the translational velocity of the rotating table undergoing general motion.

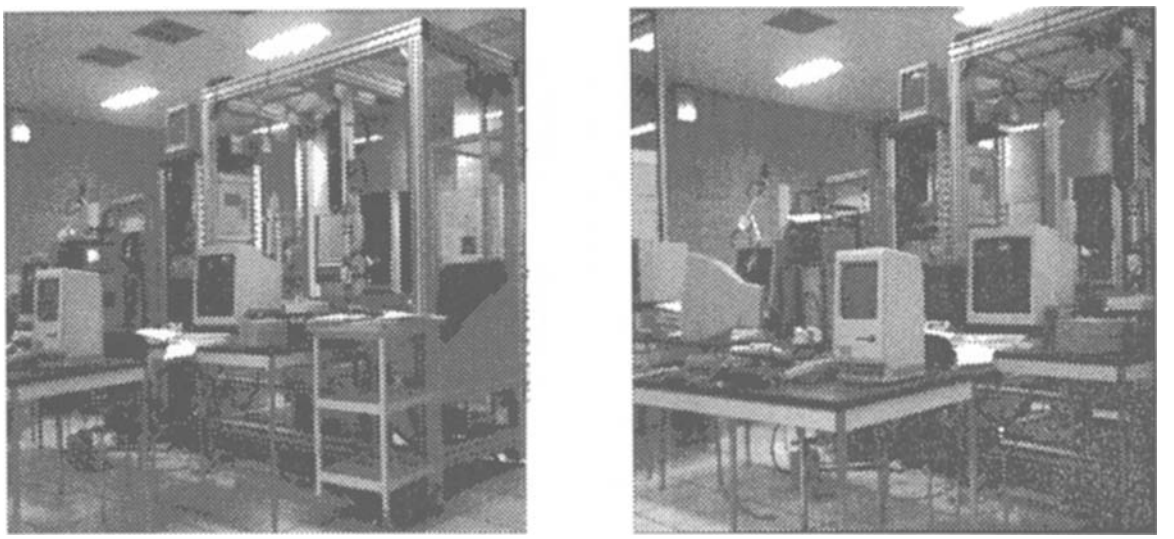

Fig. 15. The images taken by the first camera at t_1 and t_{15}.

Table 1. x and z components of the translations between successive views.

Views	1–2	2–3	3–4	4–5	5–6	6–7	7–8	8–9	9–10	10–11	11–12	12–13	13–14	14–15
x (mm)	30.8	34.8	18.7	24.7	21.0	29.5	33.1	26.2	7.1	30.7	9.8	19.9	21.1	25.4
z (mm)	104.1	85.7	91.1	99.7	107.7	108.7	89.4	95.0	126.8	97.3	104.8	104.3	103.7	93.8

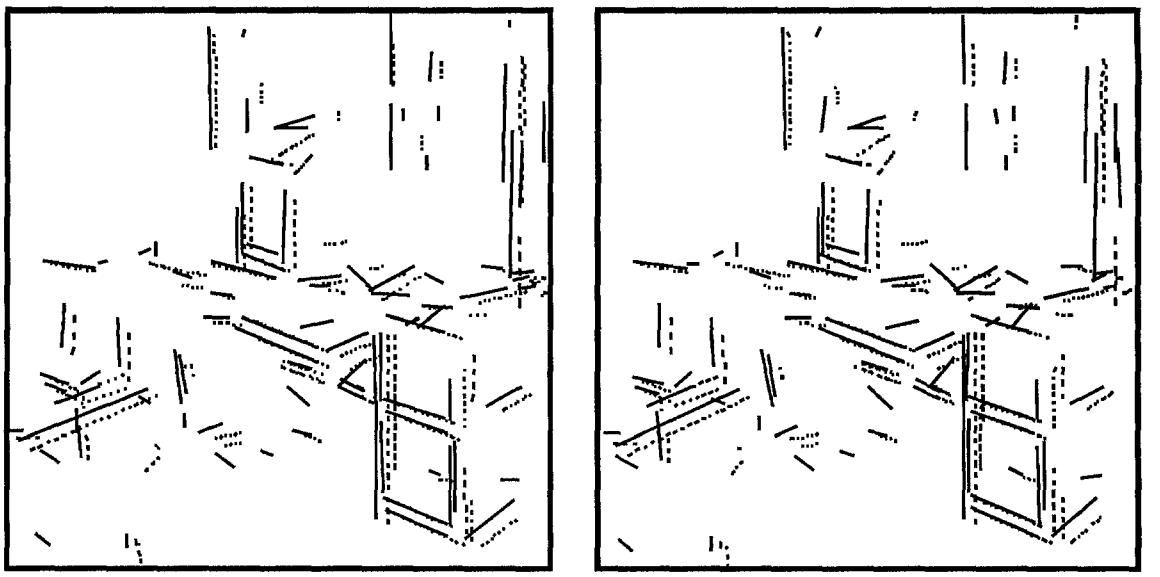

Fig. 16. A stereogram showing the superposition of the predicted (in solid lines) and observed (in dashed lines) segments at t_2.

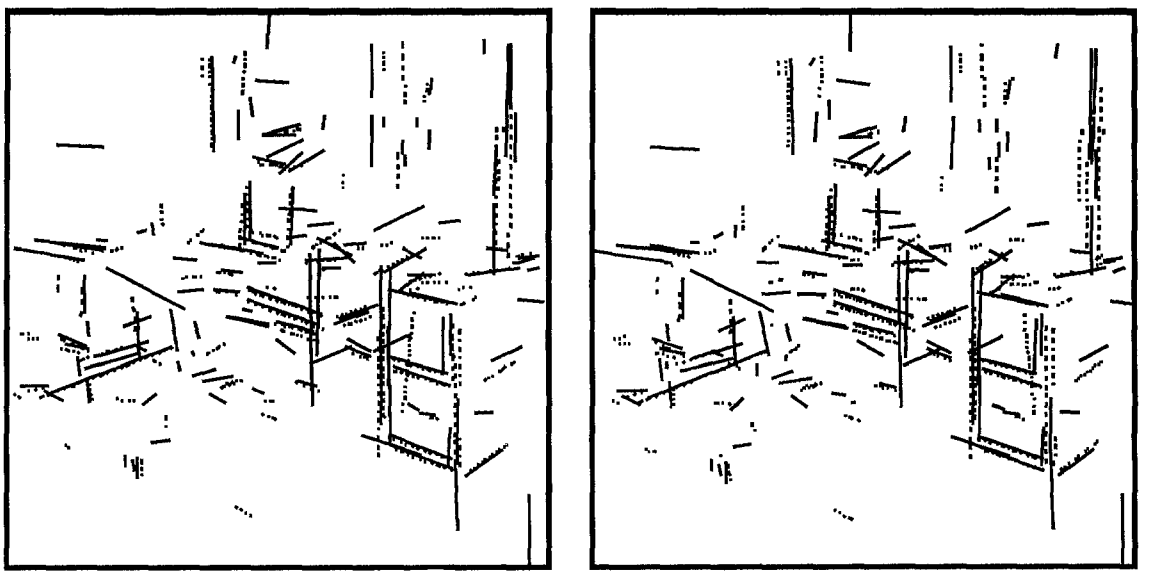

Fig. 17. A stereogram showing the superpositon of the predicted (in solid lines) and observed (in dashed lines) segments at t_3.

Figure 17 displays a stereogram showing the superposition of the predicted (in solid lines) and observed (in dashed lines) segments at t_3. As can be observed from the figure, more active tokens (in solid lines) exist at this moment: some have been activated due to multiple matches at time t_2 and some just entered into the field of view. We observe that most of the tokens already have good kinematics information since their predictions coincide well with their observations. Unfortunately, this is not always the case. For example, the motion between the ninth and tenth views is not coherent with the global motion (see table 1). Figure 18 displays a stereogram showing the superposition of the predicted (in solid lines) and observed (in dashed lines) segments at t_{10}. We can observe a big difference between the prediction and the observation. After several views, such occasional incoherent motion will be compensated for by the algorithm. Figure 19 displays a stereogram showing the superposition of the predicted (in solid lines) and observed (in dashed lines) segments

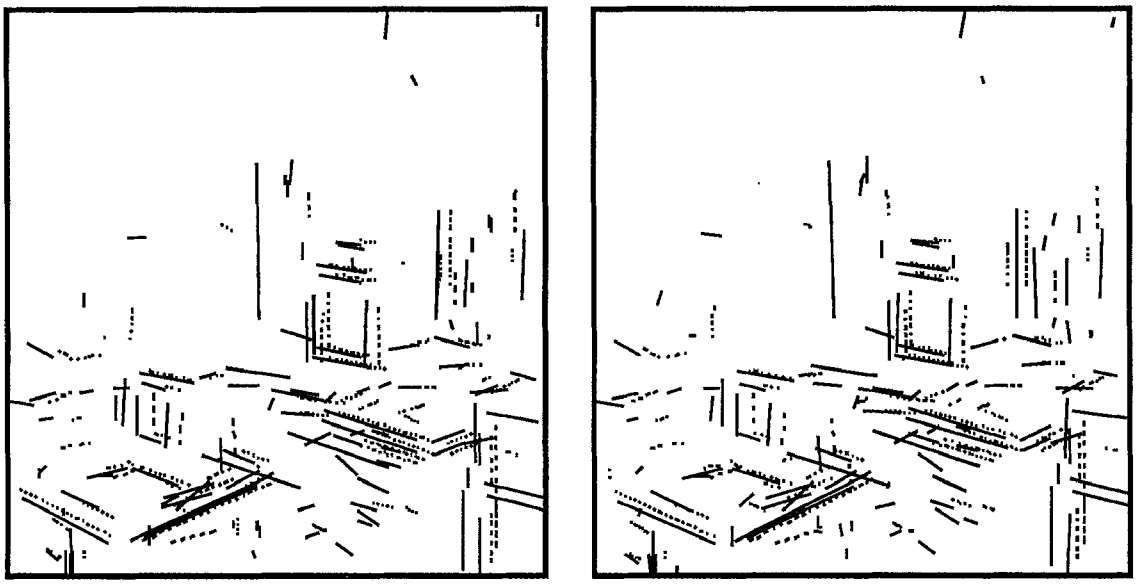

Fig. 18. A stereogram showing the superposition of the predicted (in solid lines) and observed (in dashed lines) segments at t_{10}.

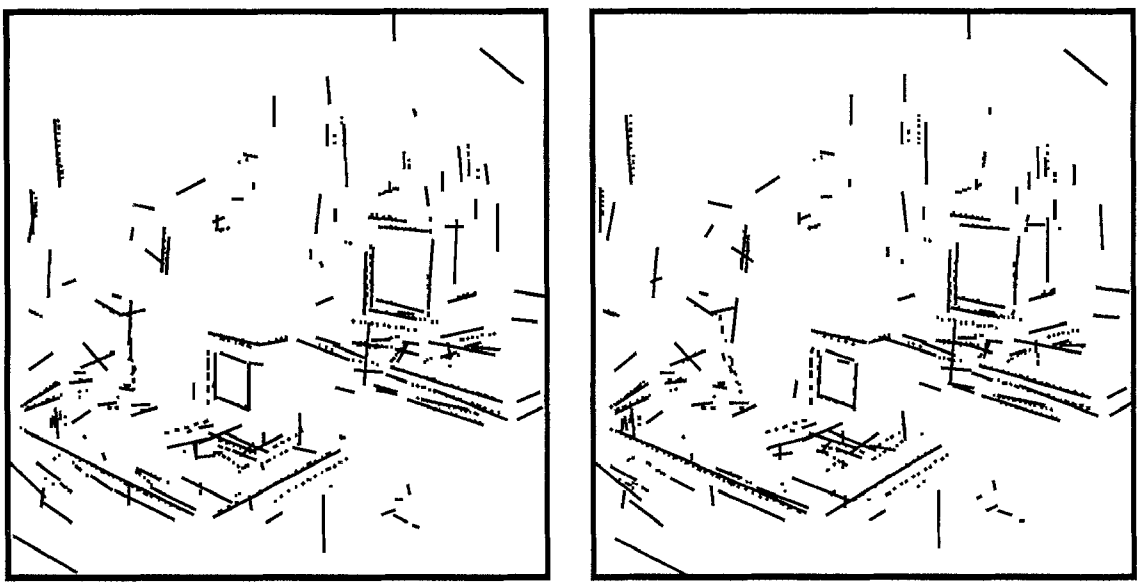

Fig. 19. A stereogram showing the superposition of the predicted (in solid lines) and observed (in dashed lines) segments at t_{15}.

at t_{15}. Quite a good fitting between the prediction and observation can be observed. The final estimate of the angular velocity (in radians/unit-time) is $[-1.1 \times 10^{-5}, 2.6 \times 10^{-3}, -3.2 \times 10^{-4}]^t$. The final estimate of the translational velocity (in millimeters/unit-time) is $[-24.5, -1.25, -97.1]^t$.

The program runs on a SUN 4/60 workstation. The number of active tokens varies between 110 and 155, except in the first view (89 tokens). The average user time for predicting a token is about 6.7 milliseconds and that for matching and updating a token is about 44.6 milliseconds. That is, if we implement the algorithm on a parallel machine with the same performance as SUN 4/60, the time required to process two frames is a little more than 50 milliseconds. The average user time required to group tokens into objects is about 2.5 seconds. In the last view, 90 segments have been identified as belonging to the static environment.

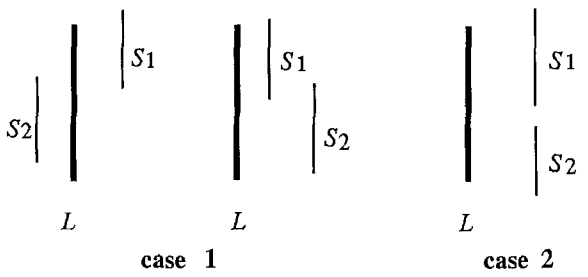

Fig. 20 Two different cases of multiple matches.

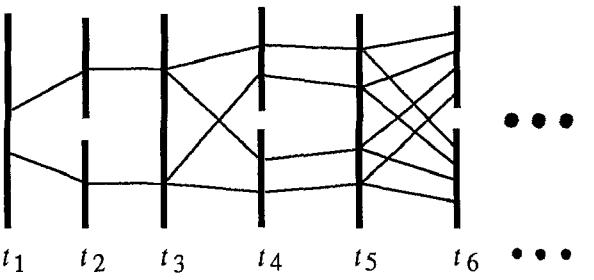

Fig. 21. An example to show the importance of merging.

12 Discussion

In the above discussion about multiple matches we presented the concept of *splitting* of a token. In fact, two cases of multiple matches should be distinguished. Figure 20 shows such cases. The token L is represented by a thick line segment. It has two matches S_1 and S_2 (represented by thin line segments). In the first case, the two matches S_1 and S_2 are not collinear. The splitting technique can be applied to handle the problem. That is, the token L is duplicated, one pursuing the tracking by incorporating S_1, and another pursuing the tracking by incorporating S_2.

In the second case, the two matches S_1 and S_2 are collinear. Of course, the splitting technique is still applicable, but we do not want to apply it (see below). A reasonable interpretation is that S_1 and S_2 are both parts of a single longer segment in space. The segment is broken in two because of the edge detection process, the line segment fitting process or other reasons. Note that not *any* two collinear segments can be interpreted this way. We use the fact that there exists a token, L which links S_1 and S_2 together because it has been matched to both of them. Based on the above consideration, we first *merge* S_1 and S_2 into a single segment S (see [Zhang & Faugeras 1990a] for the merging technique). The token L continues to track by matching S without splitting.

The *merging* concept is very important. It can avoid the abnormal growth of the number of tokens due to splitting. For example, in an extreme case as shown in figure 21, we have 2 tokens at t_2, 4 tokens at t_4, 8 tokens at t_6, ...

13 Conclusion

Several of the ideas developed here are not completely new, and can be found in the literature [Roach & Aggarwal, 1979; Hwang 1989; Crowley et al. 1988]. Consider, for example, the concept of support. In [Hwang 1989], Hwang postulates that the correspondence process in human vision is local and opportunistic. The correspondence of two image features in two consecutive frames should be determined only by the contextual information collected during some short time interval in the past. And the correspondence algorithm should allow multiple competing solutions. As more three-dimensional frames are observed, the correspondence that best fits the observed data should eventually win. Due to occlusion or absence of features or false match in the past, a trajectory may not find any match in the current frame by extension. In his algorithm, he uses a concept called *age* to indicate the number of times that a trajectory (similar to our concept of token) does not find any match in the observed frame. Any trajectory whose age is greater than MAXAGE (a fixed integer) is removed from further consideration.

Our concept of support differs significantly from the age concept in that it takes into account not only the number of times a token has not been present, but also the number of times it has been present in the past and how well the measurements agreed with the predictions. This is nicely summarized in the log-likelihood function defined in section 9.

Jenkin and Tsotsos [1986] proposed an approach different from ours to handle the multiple matching problem. They called it a "wait and see" approach: multiple matches are first hypothesized to be correct and are later disambiguated based on a temporal smoothness constraint by considering all possible *temporal* combinations. Their approach requires storing of observed data during the last instants (at least three frames) in memory. In our approach, only the log-likelihood function is updated and retained.

Our approach to tracking is similar to several methods existing in the literature [Gennery 1982; Broida & Chellappa 1986; 1989] in the sense that they

are also based on estimation theory and that the state parameters are estimated by filtering over time. The main difference is that the other approaches assume that there exists only one moving object in the scene or that all objects are known a priori. Such an assumption is not used in our approach, in which objects are segmented after tracking, making it more flexible.

Our system is based upon three main assumptions. The first one is that the stereo process provides us with features that are fairly stable over time in the sense that they move rigidly in space. Our choice of features has been straight-line segments which work well if the environment is close to polyhedral. It is expected that if many curved objects are present the method will break down unless we choose new features; but this is an open area for research.

The second assumption is the approximation of the angular velocity by a constant vector and the linear velocity by a polynomial of known degree in time (in practice, the degree has been taken equal to 0 or 1). The goodness of this approximation increases with the time sampling frequency which is limited by the rate at which we can perform stereo. Currently our hardware for stereo works at frequencies varying between 1 and 5 Hz depending on the complexity of the scene. We have noticed that the use of noise in the state equation was very useful in that it helped compensate for the systematic modeling errors.

The third assumption is that the Kalman filter and more precisely the extended Kalman filter can be used to predict the positions and orientations of the straight-line segments being tracked. Even though the assumptions that underlie the use of the Kalman filter are stretched a little in our case (but we have justified the reasons for our choice in sections 4 and 6) we have found in our experiments that we obtained excellent results if the first assumption was well verified.

We are currently programming this technique on special-purpose hardware to accommodate the throughput of the stereo process and are trying to extend it to deal with significantly curved objects.

Appendix: Proofs of the Theorems

In this appendix, we prove that when the angular velocity $\omega(t)$ is constant and the linear velocity $\mathbf{v}(t)$ is a polynomial in t we can integrate in closed-form the differential equation (13), that is, we can express the vector $\mathbf{p}_t$ as a function of $\omega(t)$, $\mathbf{v}(t)$.

The following notation (exponential of a matrix) will be used in this appendix:

$$e^M \triangleq \mathbf{I} + \frac{1}{1!} M + \frac{1}{2!} M^2 + \ldots + \frac{1}{n!} M^n + \ldots \tag{49}$$

where M is an $m \times m$ matrix, $\mathbf{I}$ is the $m \times m$ identity matrix, and M^n denotes the multiplication of n matrixes M, that is,

$$M^n \triangleq \overbrace{M \ldots M}^{n}$$

If M is a constant matrix, it can be easily shown that

$$\frac{d}{dt} e^{Mt} = Me^{Mt} = e^{Mt}M \tag{50}$$

since we have

$$\frac{d}{dt} (Mt)^n = \frac{d}{dt} (M^n t^n) = nM^n t^{n-1}$$
$$= nM(Mt)^{n-1} = n(Mt)^{n-1}M$$

Note that in general, if M is not a constant matrix, equation (50) does not hold.

The following theorem known as *Roderigues' formula* [Rodrigues 1840] will be used in the following derivation.

Theorem 3. *Rodrigues' formula:*
Given a three-dimensional vector $\mathbf{r}$. *The following relation holds:*

$$e^{\tilde{\mathbf{r}}} = \mathbf{I}_3 + \frac{\sin\theta}{\theta} \tilde{\mathbf{r}} + \frac{1 - \cos\theta}{\theta^2} \tilde{\mathbf{r}}^2 \tag{51}$$

where $\tilde{\mathbf{r}}$ *has the same definition as in equation (12) and* θ *is the magnitude of* $\mathbf{r}$ *(i.e.,* $\theta = ||\mathbf{r}||$*).*

Proof: From equation (49), $e^{\tilde{\mathbf{r}}}$ can be written as

$$e^{\tilde{\mathbf{r}}} = \mathbf{I}_3 + \frac{1}{1!} \tilde{\mathbf{r}} + \frac{1}{2!} \tilde{\mathbf{r}}^2 + \ldots + \frac{1}{n!} \tilde{\mathbf{r}}^n + \ldots$$

It can be easily verified that

$$\tilde{\mathbf{r}}^{2n-1} = (-1)^{n-1}\theta^{2(n-1)}\tilde{\mathbf{r}} \qquad \text{for } n \geq 1$$
$$\tilde{\mathbf{r}}^{2n} = (-1)^{n-1}\theta^{2(n-1)}\tilde{\mathbf{r}}^2 \qquad \text{for } n \geq 1$$

Therefore we have

$$e^{\tilde{\mathbf{r}}} = \mathbf{I}_3 + \left(\frac{1}{1!} - \frac{\theta^2}{3!} + \ldots + \frac{(-1)^{n-1}\theta^{2(n-1)}}{(2n-)!} + \ldots \right) \tilde{\mathbf{r}}$$

$$+\left(\frac{1}{2!}-\frac{\theta^2}{4!}+\ldots+\frac{(-1)^{n-1}\theta^{2(n-1)}}{(2n)!}+\ldots\right)\tilde{\mathbf{r}}^2$$

Recall that

$$\sin\theta=\frac{\theta}{1!}-\frac{\theta^3}{3!}+\frac{\theta^5}{5!}+\ldots+(-1)^{n-1}\frac{\theta^{2n-1}}{(2n-1)!}+\ldots$$

and

$$\cos\theta=1-\frac{\theta^2}{2!}+\frac{\theta^4}{4!}-\ldots+(-1)^n\frac{\theta^{2n}}{(2n)!}+\ldots$$

we get formula (51). ■

Now we show the following important theorem.

Theorem 4: *Trajectory with constant angular velocity and polynomial linear velocity.*

The trajectory of a point $\mathbf{p}_t$ *given by equation (13) can be given in closed form if the angular velocity is constant and if the translational velocity is a polynomial of degree* n $(n \geq 0)$. ■

Proof: Let the motion be described by $\omega_t = \omega$ and

$$\mathbf{v}_t=\mathbf{v}_0+\mathbf{v}_1(t-t_0)+\mathbf{v}_2\frac{(t-t_0)^2}{2!}+\ldots+\mathbf{v}_n\frac{(t-t_0)^n}{n!}$$

where $\mathbf{v}_0$ is the velocity at time t_0. Define a new variable $\mathbf{y}_t$ so that

$$\mathbf{y}_t=e^{-\tilde{\omega}(t-t_0)}\mathbf{p}_t$$

This yields $\mathbf{y}_{t_0}=\mathbf{p}_0$, the position of the point at t_0. Based on equation (50), we have

$$\dot{\mathbf{y}}_t=e^{-\tilde{\omega}(t-t_0)}(\dot{\mathbf{p}}_t-\tilde{\omega}\mathbf{p}_t)=e^{-\tilde{\omega}(t-t_0)}\mathbf{v}_t$$

Integrating the above equation, we get

$$\mathbf{y}_t=\mathbf{p}_0+\int_{t_0}^{t}e^{-\tilde{\omega}(s-t_0)}\mathbf{v}_s\,ds$$

From the definition of $\mathbf{y}_t$, we thus have

$$\begin{aligned}\mathbf{p}_t&=e^{\tilde{\omega}(t-t_0)}\mathbf{y}_t\\&=e^{\tilde{\omega}(t-t_0)}\mathbf{p}_0+\int_{t_0}^{t}e^{\tilde{\omega}(t-s)}\mathbf{v}_s\,ds\end{aligned}\qquad(52)$$

Using theorem 3, we find that

$$e^{\tilde{\omega}(t-t_0)}=W$$

where W is given by equation (15). Denoting the last term of equation (52) by $\mathbf{r}_t$ and using theorem 3, we have

$$\begin{aligned}\mathbf{r}_t=\int_{t_0}^{t}&\left[\mathbf{I}_3+\frac{\sin[\theta(t-s)]}{\theta}\tilde{\omega}+\frac{1-\cos[\theta(t-s)]}{\theta^2}\tilde{\omega}^2\right]\\&\cdot\left[\mathbf{v}_0+\mathbf{v}_1(s-t_0)+\mathbf{v}_2\frac{(s-t_0)^2}{2!}+\ldots+\mathbf{v}_n\frac{(s-t_0)^n}{n!}\right]ds\end{aligned}$$

Let us define

$$L_k=\int_{t_0}^{t}(s-t_0)^k\sin[\theta(t-s)]\,ds$$

and

$$J_k=\int_{t_0}^{t}(s-t_0)^k\cos[\theta(t-s)]\,ds$$

We can express $\mathbf{r}_t$ in terms of L_k and $J_k(k=0\ldots n)$. If we obtain closed-form expressions for L_k and J_k, this will be true also for $\mathbf{r}_t$. Indeed,

$$\begin{aligned}L_k&=\frac{1}{\theta}\int_{t_0}^{t}(s-t_0)^k\,d(\cos[\theta(t-s)])\\&=\frac{1}{\theta}(s-t_0)^k\cos[\theta(t-s)]\Big|_{s=t_0}^{s=t}\\&\quad-\frac{k}{\theta}\int_{t_0}^{t}(s-t_0)^{k-1}\cos[\theta(t-s)]\,ds\\&=\frac{1}{\theta}(t-t_0)^k-\frac{k}{\theta}J_{k-1}\end{aligned}\qquad(53)$$

$$\begin{aligned}J_k&=-\frac{1}{\theta}\int_{t_0}^{t}(s-t_0)^k\,d(\sin[\theta(t-s)])\\&=-\frac{1}{\theta}(s-t_0)^k\sin[\theta(t-s)]\Big|_{s=t_0}^{s=t}\\&\quad+\frac{k}{\theta}\int_{t_0}^{t}(s-t_0)^{k-1}\sin[\theta(t-s)]\,ds\\&=\frac{k}{\theta}L_{k-1}\end{aligned}\qquad(54)$$

$$\begin{aligned}L_0&=\int_{t_0}^{t}\sin[\theta(t-s)]\,ds\\&=\frac{1}{\theta}\{1-\cos[\theta(t-t_0)]\}\end{aligned}\qquad(55)$$

$$J_0=\int_{t_0}^{t}\cos[\theta(t-s)]\,ds=\frac{1}{\theta}\sin[\theta(t-t_0)]\qquad(56)$$

By iteration, we can obtain closed forms of L_k and J_k for all $k > 0$. Indeed, if we use the following notations:

$$\mathbf{x}_k = \begin{bmatrix} L_k \\ J_k \end{bmatrix}$$

$$A = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

$$\mathbf{b}_k = \begin{bmatrix} (t - t_0)^k/\theta \\ 0 \end{bmatrix}$$

$$\mathbf{a} = \begin{bmatrix} 1 - \cos[\theta(t - t_0)] \\ \sin[\theta(t - t_0)] \end{bmatrix}$$

we have

$$\mathbf{x}_k = \frac{k}{\theta} A\mathbf{x}_{k-1} + \mathbf{b}_k \qquad \text{for } k > 0 \tag{57}$$

$$\mathbf{x}_0 = \frac{1}{\theta}\mathbf{a} \tag{58}$$

After some algebra, we obtain

$$\mathbf{x}_k = \frac{k!}{\theta^{k+1}} A_k\mathbf{a} + B_k \qquad \text{for } k > 0 \tag{59}$$

where

$$A_k = (-1)^{\lceil (k+1)/2 \rceil} \begin{bmatrix} (k+1) \bmod 2 & k \bmod 2 \\ -k \bmod 2 & (k+1) \bmod 2 \end{bmatrix}$$

$$B_k = \begin{bmatrix} \sum_{i=0}^{\lceil k/2 \rceil} \frac{(-1)^i}{\theta^{2i+1}} \frac{d^{2i}}{dt^{2i}} (t - t_0)^k \\ \sum_{i=0}^{\lceil (k-1)/2 \rceil} \frac{(-1)^i}{\theta^{2(i+1)}} \frac{d^{2i+1}}{dt^{2i+1}} (t - t_0)^k \end{bmatrix}$$

Here $\lceil j \rceil$ denotes the largest integer less than j, "mod" denotes the modulo function, and $d^i/dt^i(\cdot)$ denotes the ith derivative with respect to t. This yields a closed-form expression for $\mathbf{r}_t$ and therefore, also for $\mathbf{p}_t$.

Q.E.D. ■

Theorem 1 is a special case of theorem 4 for $\mathbf{v}_t = \mathbf{v}$ (that is, $n = 0$). The reader can easily verify it.

Theorem 2 describes the special case $\mathbf{v}_t = \mathbf{v} + \mathbf{a}(t - t_0)$ (that is, $n = 1$). From equation (59), we obtain

$$L_1 = \frac{1}{\theta}(t - t_0) - \frac{1}{\theta^2}\sin[\theta(t - t_0)]$$

$$= \frac{1}{\theta^2}\{\theta(t - t_0) - \sin[\theta(t - t_0)]\} \tag{60}$$

$$J_1 = \frac{1}{\theta^2}\{1 - \cos[\theta(t - t_0)]\} \tag{61}$$

After some algebra, we get theorem 2.

Notes

1. One can also update the position parameters by modifying a little the state vector in the formulation of section 7. The computation will be more expensive, as the complexity of EKF is $O(n^3)$, where n is the dimension of the state vector.
2. For simplicity, we assume that $\mathbf{x}$ and $\mathbf{y}$ are scalar.

Acknowledgment

The authors would like to thank one of the anonymous reviewers for his detailed comments which improved this paper.

References

Aggarwal, J.K. and Wang, Y.F. 1987. Analysis of a sequence of images using point and line correspondences. *Proc. Intern. Conf. Robotics Autom.* pp. 1275–1280, Raleigh, NC, March 31–April 3.

Ayache, N. 1988. *Construction et Fusion de Représenntations Visuelles 3D—Applications à la Robotique Mobile.* Thèse d'Etat, University of Paris XI, Paris-Orsay.

Ayache, N., and Faugeras, O.D. 1987a. Building, registrating and fusing noisy visual maps. *Proc. 1st Intern. Conf. Comput. Vis.* London, June. pp. 73–82.

Ayache, N., and Faugeras, O.D. 1987b. Maintaining representations of the environment of a mobile robot. *4th Intern. Symp. Robotics Res.* Santa Cruz, CA. MIT Press: Cambridge MA.

Ayache, N., and Faugeras, O.D. 1989. Maintaining representations of the environment of a mobile robot. *IEEE Trans. Robotics Autom.* 5(6):804–819. December; also INRIA report 789.

Ayache, N., and Lustman, F. 1987. Fast and reliable trinocular stereovision. *Proc. 1st Intern. Conf. Comput. Vis.*, London, June, pp. 422–427.

Baker, H., and Binford, T.O. 1981. Depth from edge and intensity based stereo. *Proc. 7th Joint Conf. Artif. Intell.* Vancouver, August, pp. 631–636.

Bar-Shalom, Y., and Fortmann, T.E. 1988. *Tracking and Data Association.* Academic Press: San Diego, CA.

Broida, T.J. and Chellappa, R. 1986. Kinematics and structure of a rigid object from a sequence of noisy images. *Proc. IEEE Workshop on Motion: Representation and Analysis*, Charleston, SC, May, pp. 95–100.

Broida, T.J. and Chellappa, R. 1989. Experiments and uniqueness results on object structure and kinematics from a sequence of monocular images. *Proc. IEEE Workshop on Visual Motion*, Irvine, CA, March, pp. 21–30.

Crowley, J.L., Stelmaszyk, P, and Discours, C. 1988. Measuring image flow by tracking edge-lines. *Proc. 2nd Intern. Conf. Comp. Vis.* Tampa, FL, December, pp. 658–664.

Deriche, R., and Faugeras O. 1990. Tracking line segments. In *Proc. European Conference on Computer Vision*, O. Faugeras, ed. Springer-Verlag: Antibes, Fr, April, pp. 259–268.

Dickmanns, E.D. 1987. 4D-dynamic scene analysis with integral spatio-temporal models. *Proc. ISSR'87*, Santa Cruz, pp. 73–80.

Dickmanns, E.D., and Graefe, V. 1988a. Applications of dynamic monocular vision. *Mach. Vis. Appl.* 1:241–261.

Dickmanns, E.D., and Graefe, V. 1988b. Dynamic monocular machine vision. *Mach. Vis. Appl.* 1:223–240.

Durrant-Whyte, H.F. 1988. *Integration, Coordination, and Control of Multi-sensor*. Kluwer Academic Publishers: Norwell, MA.

Faugeras, O.D. 1990. On the motion of 3-D curves and its relationship to optical flow. *Proc. 1st Europ. Conf. Comp. Vis.* Springer-Verlag, April, pp. 107–117.

Faugeras, O.D. 1991. *Three-Dimensional Computer Vision*. MIT Press, Cambridge, MA. To appear.

Faugeras, O.D., and Maybank, S. 1990. Motion from point matches: Multiplicity of solutions. *Intern. J. Comput. Vis.* 4 (3):225–246; also INRIA Technical Report 1157.

Faugeras, O.D., and Toscani, G. 1986. The calibration problem for stereo. *Proc. IEEE Conf. Comput. Vis. Patt. Recog.*, Miami Beach. FL, June, pp. 15–20.

Faugeras, O.D., Ayache, N., and Zhang, Z. 1988a. A preliminary investigation of the problem of determining ego- and object-motions from stereo. *Proc. 9th Intern. Conf. Patt. Recog.* Rome, pp. 242–246.

Faugeras, O.D., Deriche, R., Ayache, N., Lustman, F., and Giuliano, E. 1988b. Depth and motion analysis: The machine being developed within Esprit Project 940. *Proc. IAPR Workshop on Comput. Vis. (Special Hardware and Industrial Applications)*, October, Tokyo, pp. 35–44.

Faugeras, O.D., Lebras-Mehlman, E., and Boissonnat, J.D. 1990. Representing stereo data with the Delaunay triangulation. *Artif. Intell. J.* 44(1–2); also INRIA Technical Report 788.

Gambotto, J.P. 1989. Tracking points and line segments in image sequences. *Proc. IEEE Workshop on Visual Motion*, Irvine, CA, March 20–22, pp. 38–45.

Gennery, D.B. 1982. Tracking known three-dimensional objects. *Proc. Amer. Assoc. Artif. Intell.*, Carnegie-Mellon University, Pittsburgh.

Gibson, J.J. 1950. *The Perception of the Visual World*. Houghton-Mifflin: Boston, MA.

Gordon, G.L. 1989. On the tracking of featureless objects with occlusion. *Proc. IEEE Workshop of Visual Motion*, Irvine, CA, March 20–22, pp. 13–20.

Grimson, W.E.L. 1981. *From Images to Surfaces*. MIT Press: Cambridge, MA

Grimson, W.E.L. 1985. Computational experiments with a feature based stereo algorithm. *IEEE Trans. Patt. Anal. Mach. Intell.* 7(1): 17–34.

Hildreth, C. 1984. *The Measurement of Visual Motion*. MIT Press: Cambridge, MA.

Horn, K P., and Schunk, B.G. 1981. Determining optical flow. *Artificial Intelligence* 17:185–203.

Huang, T.S., and Tsai, R.Y. 1981. Image sequence analysis: Motion estimation. In T.S. Huang, ed., *Image Sequence Processing and Dynamic Scene Analysis*. Springer-Verlag: New York.

Hwang, V.S.S. 1989. Tracking feature points in time-varying images using an opportunistic selection approach. *Pattern Recognition*, 22(3):247–256.

Jenkin, M., and Tsotsos, J.K. 1986. Applying temporal constraints to the dynamic stereo problem. *Comput. Vis. Graph. Image Process.*, 24:16–32.

Kim, Y.C., and Aggarwal, J.K. 1987. Determining object motion in a sequence of stereo images. *IEEE J. Robot. Autom.* 3(6):599–614.

Kitamura, Y., and Yachida, M. 1990. Three-dimensional data acquisition by trinocular vision. *Advanced Robotics* 4(1):29–42, Robotics Society of Japan.

Koenderink, J.J. 1986. Optic flow. *Vision Research* 26(1):161–180.

Koenderink, J.J., and van Doorn, A.J. 1975. Invariant properties of the motion parallax field due to the movement of rigid bodies relative to an observer. *Optica Acta* 22:717–723.

Koenderink, J.J., and van Doorn, A.J. 1978. *How an Ambulant Observer Can Construct a Model of the Environment from the Geometrical Structure of the Visual Inflow*. Oldenburg: Muenchen.

Liu, Y., and Huang, T.S. 1986. Estimation of rigid body motion using straight line correspondences. *Proc. Workshop on Motion: Representation and Analysis*, Charleston, May, pp. 47–51.

Liu, Y., and Huang, T.S. 1988. A linear algorithm for determining motion and structure from line correspondences. *Comput. Vis. Graph. Image Process.* 44(1):35–57.

Longuet-Higgins, H.C. 1981. A computer algorithm for reconstructing a scene from two projections. *Nature* 293:133–135.

Lowerre, B , and Reddy, R. 1980. The harpy speech understanding system. In W. Lea, ed. *Trends in Speech Recognition*. Prentice-Hall: Englewood Cliffs, NJ.

Marr, D., and Poggio, T. 1975. Cooperative computation of stereo disparity. *Science* 194:283–287.

Marr, D., and Poggio, T. 1979. A computational theory of human stereo vision. *Proc. Roy. Soc.*, B-204:301–328.

Maybeck, P.S. 1979. *Stochastic Models, Estimation and Control*. vol. 1. Academic Press: San Diego, CA.

Maybeck, P.S. 1982. *Stochastic Models, Estimation and Control*. vol. 2. Academic Press: San Diego, CA.

Nagel, H.H. 1983. Displacement vectors derived from second order intensity variations in image sequences. *Comput. Vis. Graph. Image Process.* 21:85–117.

Nagel, H.H. 1986. Image sequences—ten (octal) years—from phenomenology towards a theoretical foundation. *Proc. 8th Intern Conf. Patt. Recog.* Paris, October, pp. 1174–1185.

Nishihara, H.K. 1984. PRISM, a practical real-time imaging stereo matcher. Technical Report A.I. Memo 780, MIT. Cambridge, MA, 31 pages.

Ohta, Y., and T. Kanade 1985. Stereo by intra- and inter-scanline search. *IEEE Trans. Patt. Anal. Mach. Intell.* 7(2):139–154.

Pollard, S.B., Mayhew, J.E.W., and Frisby, J.P. 1985. PMF: A stereo correspondence algorithm using a disparity gradient constraint. *Perception*, 14;449–470.

Preparata, F., and Shamos, M. 1986. *Computational Geometry, An Introduction*. Springer-Verlag: New York.

Roach, J.W., and Aggarwal, J.K. 1979. Computer tracking of objects moving in space. *IEEE Trans. Patt. Anal. Mach. Intell.* 1(2):127–135.

Roberts, K.S. 1988. A new representation for a line. *Proc. Conf. Comput. Vis. Patt. Recog.* Ann Arbor, June 5–9, pp. 635–640.

Rodrigues, O. 1840. Des lois géométriques qui régissent les déplacements d'un système solide dans l'espace, et de la variation des

coordonnées provenant de ces déplacements considérés indépendamment des causes qui peuvent les produire. *Journal de Mathématiques Pures et Appliquées* 5:380–440.

Sethi, S.K., and Jain, R. 1987. Finding trajectories of feature points in a monocular image sequence. *IEEE Trans. Patt. Anal. Mach. Intell.* 9(1):56–73.

Tsai, R.Y., and Huang, T.S. 1981. Estimating 3-D motion parameters of a rigid planar patch, i. *IEEE Trans. Acoustic, Speech Sig. Process.* 29(6):1147–1152.

Tsai, R.Y., and Huang, T.S. 1984. Uniqueness and estimation of three-dimensional motion parameters of rigid objects with curved surface. *IEEE Trans. Patt. Anal. Mach. Intell.* 6(1):13–26.

Ullman, S. 1979. *The Interpretation of Visual Motion*. MIT Press: Cambridge, MA.

Webb, J.A., and Aggarwal, J.K. 1982. Structure from motion of rigid and jointed objects. *Artificial Intelligence*. 19:107–130.

Weng, J., Huang, T.S., and Ahuja, N. 1987. 3-D motion estimation, understanding, and prediction from noisy image sequences. *IEEE Trans. Patt. Anal. Mach. Intell.* 9(3):370–389.

Yachida, M. 1986. 3D data acquisition by multiple views. In O.D. Faugeras and G. Giralt, eds. *Robotics Research: the Third International Symposium*. MIT Press, Cambridge, MA. pp. 11–18.

Yen, B.L. and Huang, T.S. 1983. Determining 3-D motion/structure of a rigid body over 3 frames using straight line correspondences. *Proc. Conf. Comput. Vis. Patt. Recog.*, Washington, DC. June 19–23, pp. 267–272.

Young, G.S., and Chellappa, R. 1988. 3-D motion estimation using a sequence of noisy stereo images. *Proc. Conf. Comput. Vis. Patt. Recog.*, Ann Arbor, pp. 710–716.

Zhang, Z. 1990. *Motion Analysis from a Sequence of Stereo Frames and its Applications*. PhD Thesis, University of Paris-Sud, Orsay, Paris, France, in English.

Zhang, Z., and Faugeras, O.D. 1990a. Building a 3D world model with a mobile robot: 3D line segment representation and integration. *Proc. 10th Intern. Conf. Patt. Recog.*, Atlantic City, June, pp. 38–42.

Zhang, Z., and Faugeras, O.D. 1990b. Tracking and motion estimation in a sequence of stereo frames. In L.C. Aiello, ed. *Proc. 9th Europ. Conf. Artif. Intell*, Stockholm, August, pp. 747–752.

Zhang, Z., and Faugeras, O.D. 1991. Determining motion from 3D line segments: a comparative study. *Image and Vis. Comput.* 9(1): 10–19.

Zhang, Z., and Faugeras, O.D. 1992. Estimation of displacements from two 3D frames obtained from stereo. *IEEE Trans. Patt. Anal. Mach. Intell.*, accepted, to appear. 1992.

Zhang, Z., Faugeras, O.D., and Ayache, N. 1988. Analysis of a sequence of stereo scenes containing multiple moving objects using rigidity constraints. *Proc. 2nd Intern. Conf. Comput. Vis.*, Tampa, FL, December, pp. 177–186.

Zhuang, X., and Haralick, R.M. 1985. Two view motion analysis. *Proc. Conf. Comput. Vis. Patt. Recog.*, San Francisco, California, June, pp. 686–690.