INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Token Tracking in a Cluttered Scene

Zhengyou Zhang

N° 2072

Octobre 1993

Z. Zhang, "Token Tracking in a Cluttered Scene",
Image and Vision Computing, Vol.12, No.2,
pages 110-120, 1994.

PROGRAMME 4

Robotique,
image
et vision

Rapport
de recherche

1993

ISSN 0249-6399

Token Tracking in a Cluttered Scene

Zhengyou Zhang

Programme 4 — Robotique, image et vision
Projet Robotvis

Rapport de recherche n˚ 2072 — Octobre 1993 — 24 pages

Unité de recherche INRIA Sophia-Antipolis
2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex (France)
Téléphone : (33) 93 65 77 77 – Télécopie : (33) 93 65 77 65

Abstract: The statistical data association technique is an important approach to analyze long sequences of images in Computer Vision. Although it has extensively been studied in other domains such as in radar imagery, it was introduced only recently in Computer Vision, and is already recognized as an efficient approach to solving correspondence and motion problems. This paper has two purposes. The first is to present a general formulation of token tracking. The parameterization of tokens is not addressed. This might be useful to those who are not familiar with statistical tracking techniques. The second is to introduce some strategies for tracking with emphasis on practical importance. They include beam search for resolving multiple matches, support of existence for discarding false matches, and locking on reliable tokens and maximizing local rigidity for handling combinatorial explosion. We have implemented those strategies in a 3D line segment tracking algorithm and found them very useful.

Key-words: Token Tracking, Matching, Cluttered Scenes, Search Strategies

(Résumé : tsvp)

This paper will appear in the international journal of ***Image and Vision Computing***, 1993. A short version appeared in ***Proc. British Machine Vision Conference BMVC93***, pages 207–216, University of Surrey, Guildford, UK, 21-23 September 1993

Suivi de cible dans une scène encombrée

Résumé : La technique statistique de la fusion de données est une approche importante en vision par ordinateur pour analyser une séquence longue d'images. Bien qu'elle ait été largement étudiée dans les autres domaines comme l'imagerie radar, ce n'est que récemment qu'elle a été adoptée par la communauté de vision par ordinateur, et est déjà considérée comme une approche efficace pour résoudre la mise en correspondance et l'analyse du mouvement. Cet article a deux objectifs :

- présenter une formulation générale du suivi de cible; la représentation de primitives n'est pas traitée; ceci peut être utile pour ceux qui ne connaissent pas très bien les techniques statistiques de suivi,
- introduire quelques stratégies pour le suivi en insistant sur l'importance pratique; les stratégies sont : recherche par faisceau pour lever des ambiguïtés d'appariements, support de l'existence pour écarter de mauvais appariements, gestion de priorités des cibles et utilisation de contraintes de rigidité locale pour éviter l'explosion combinatoire.

Nous avons implanté ces stratégies dans le cadre du suivi de segments de droite 3D, et elles ont été trouvées très utiles.

Mots-clé : Suivi de cible, Mise en correspondance, Scènes encombrées, Stratégies, Recherche

Contents

List of Figures

1 Introduction

Statistical data association techniques have been extensively studied in radar imagery for target tracking[1,2]. Only recently they were introduced in Computer Vision. Early work on motion analysis in Computer Vision was mainly on the computation of motion for two frames obtained from two quite different positions[3-5]. One dominant difficulty is the establishment of feature correspondences between frames. Many techniques have been proposed which are mainly based on subgraph isomorphism, relational structure matching and tree searching[6-11]. A number of constraints or heuristics, especially the rigidity assumption, have been incorporated. The correspondence problem is still found to be very difficult. Sooner, researchers realized that the problem would become much easier if long sequences of images taken at short time interval are used. Indeed, as the time interval is small and object velocity is constrained by physical laws, the interframe displacements of objects are bounded, i.e., the correspondence of a token at the next instant must be in its neighborhood. Furthermore, objects usually move smoothly[12-14], thus the motion coherence can be used to predict the occurrence of tokens in the future, which considerably reduces the search space. The statistical data association techniques for target tracking, originally developed for radar imagery, fit well in this framework, and are already recognized as an efficient approach to solving correspondence and motion problems[15-17]. The reader is referred to[18] for a recent review of statistical data association techniques.

However, most of these techniques were originally developed for tracking a few and known targets, although recently progresses have been made to deal with a large number of targets[19]. The theoretical base under these techniques is directly applicable to tracking problems in computer vision. A number of particularities, though, are required to be taken care:

- large number of tokens, usually several hundreds. Furthermore, several tokens are close to each other.
- appearance. A previously unseen object may partially or totally come into view.
- disappearance. A moving object in the current field of view may move partially or totally out of it in the next frames.
- occlusion. A moving object may be partially or totally occluded by the background or by other objects.
- absence. Some tokens which should be present are not due to the failure of the feature extraction (or reconstruction) process.
- coherence of tokens. In radar imagery, a target represents an object, e.g., an aircraft, and it usually moves independently from the others. In computer vision, however, tokens originate from several independently moving objects. Thus tokens from a single object undergo a similar (same, if the object is rigid) 3D motion.

The interested reader is referred to[20,21] for the above topics. This paper is a continuation of our previous work and we concentrate on a couple of strategies we recently developed for tracking tokens in a cluttered scene. After presentation of a formulation for tracking "general" tokens, we describe in this paper some strategies for tracking with emphasis on practical importance. They include beam search for resolving multiple matches, support of existence for discarding false matches, and locking on reliable tokens and maximizing local rigidity for handling combinatorial explosion. We have implemented those strategies in a 3D line segment tracking algorithm and found them very useful.

2 Notations and Terminology

We are interested in tracking geometric primitives including points, lines and curves. A group of geometric primitives such as vertex and attached edges is also of interest. We shall call them *tokens.* A token at time t_i is characterized by its position, orientation and kinematic parameters, which are captured in a vector called the *state vector* $\mathbf{x}_i$. An imaging system observes the token which is represented by a vector called the *measurement* (or *observation*) *vector* $\mathbf{z}_i$. We call the observation a *scene token.*

The (right) *subscript* is used to denote the time instant, as in $\mathbf{x}_i$ and $\mathbf{z}_i$. At each instant, there are many tokens and scene tokens which will be distinguished to each other by a *left subscript.* For example, ${}_j\mathbf{z}_i$ is the jth scene token observed at time i. One or both subscripts will be omitted if this does not result in any ambiguity. The caret ˆ denotes the estimation or prediction. For example, $\hat{\mathbf{x}}_{k|k-1}$ denotes the prediction of the state at time k given measurements up to time $k-1$. P denotes the covariance matrix of a state vector and Λ denotes that of a measurement vector.

3 Problem Formulation

The dynamics of a token is assumed to be described by a difference equation

$$\mathbf{x}_{k+1} = \mathbf{f}_k(\mathbf{x}_k) + \mathbf{w}_k \ , \tag{1}$$

where $\mathbf{f}_k(\cdot)$ is a vector function describing the transition of the state vector from t_k to t_{k+1} (the so-called *state transition function*), and $\mathbf{w}_k$ is the random disturbance of the dynamic system. In practice, the state transition function is determined by the underlying token kinematics assumed. Two commonly used kinematic models are:

a) Polynomial model: State variables evolve polynomially in time. In general, constant velocity or constant acceleration model is used[15,16].

b) General motion model: A token is assumed to undergo a motion with polynomial angular velocity and polynomial translational velocity[20,22]. In practice, constant angular velocity and constant translational velocity or acceleration model is sufficient.

In fact, the polynomial model is a special case of the general motion model where the angular velocity is zero. One advantage of the polynomial model is that the transition function $\mathbf{f}_k(\cdot)$ is linear while we generally cannot write down a linear function using the latter model. However, the latter can more reasonably approximate a real motion than the former.

The statistical property of the system noise term $\mathbf{w}_k$ cannot in general be known exactly. We model $\mathbf{w}_k$ as an independent Gaussian noise sequence with zero mean and known covariance, i.e.,

$$E[\mathbf{w}_k] = 0 \ , \quad \text{and} \quad E[\mathbf{w}_k \mathbf{w}_l^T] = Q_k \delta_{kl} \tag{2}$$

for all k and l, where δ_{kl} is the Kronecker delta, which is 1 for $k = l$ and 0 otherwise. Q_k is usually determined on the basis of the designer's experience and physical understanding about the dynamic system. It is used, on the one hand, to model disturbances due to, for example, vibration of objects during motion, and on the other hand, to partially take into account the error in modeling. The model we use is only an approximation to the real motion which is usually very complex.

The measurement equation describes the relation between measurements (observations) and state variables of the dynamic system, which can usually be expressed as

$$\mathbf{z}_k = \mathbf{h}_k(\mathbf{x}_k) + \mathbf{n}_k \ , \tag{3}$$

where $\mathbf{h}_k(\cdot)$ is a vector function called the *observation function* and $\mathbf{n}_k$ represents the random noise contained in the measurements. Measurements are obtained through some signal processing algorithm such as edge detection and 3D reconstruction in a stereo system. The statistical property of $\mathbf{n}_k$ is either provided by the signal processing algorithm if uncertainty is modeled or determined on the basis of the designer's experience and physical understanding of the signal processing algorithm. We model $\mathbf{n}_k$ as an independent

Gaussian noise sequence with zero mean and known covariance, i.e.,

$$E[\mathbf{n}_k] = 0 \ , \quad \text{and} \quad E[\mathbf{n}_k \mathbf{n}_l^T] = R_k \delta_{kl} \tag{4}$$

for all k and l.

In general, the noise in the state equation and that in the measurement equation are determined independently. We thus assume that there is no correlation between them, that is, $E[\mathbf{w}_k \mathbf{n}_l^T] = 0$ for all k and l.

Given a sequence of measurements $\{\mathbf{z}_k \mid k = 1..n\}$ of a token, we are ready to use the Kalman filter if $\mathbf{f}_k(\cdot)$ and $\mathbf{h}_k(\cdot)$ are linear, or the extended Kalman filter otherwise, to estimate the state variable $\mathbf{x}_k$ of the token. The reader is referred to[23−25] for the details of the Kalman filter and the extended Kalman filter.

4 Main Steps in Tracking

We shall sketch out in this section the tracking process. By tracking, we mean establishing at each instant a correspondence between the tokens being tracked and the scene tokens observed. As time goes on, some tokens move out of and some others come into the field of view. Thus we must also deal with the disappearance and appearance problems. The tracking problem becomes more difficult, because some tokens may be occluded by others (the so-called *occlusion problem*) or may not be detected due to temporary failure of the signal processing algorithm (which we refer as the *absence problem*). We shall address these issues in this and next sections.

4.1 Prediction-Matching-Update Loop

The tracking is performed in a prediction-matching-update loop. At time t ($t_{k-1} \leq t < t_k$), i.e., before data at t_k are available, we predict its occurrence at t_k for each token being tracked. When data at t_k are available, we try to find for each token a scene token as its match in the neighborhood of its predicted position. When a match is found, the token (state) parameters are updated using either the Kalman filter or the extended Kalman filter (EKF). In the following, both the state transition and measurement observation functions are assumed nonlinear, and the EKF will be used. The discussions, however, are directly applicable to the linear case.

The prediction is done in two stages. First, the state and its error covariance are propagated to t_k according to Eq. (1), that is

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{f}_{k-1}(\hat{\mathbf{x}}_{k-1}) \ , \tag{5}$$

$$P_{k|k-1} = F_{k-1}(\hat{\mathbf{x}}_{k-1}) P_{k-1} F_{k-1}^T(\hat{\mathbf{x}}_{k-1}) + Q_{k-1} \ , \tag{6}$$

where $F_{k-1}(\hat{\mathbf{x}}_{k-1})$ is the partial derivative matrix of $\mathbf{f}_k(\mathbf{x})$ with respect to $\mathbf{x}$, i.e.,

$$F_{k-1}(\hat{\mathbf{x}}_{k-1}) \triangleq \frac{\partial \mathbf{f}_{k-1}(\mathbf{x})}{\partial \mathbf{x}} \Big|_{\mathbf{x}=\hat{\mathbf{x}}_{k-1}} \ . \tag{7}$$

As one can observe, we use the first order approximation to compute the prediction of the state error covariance if $\mathbf{f}_{k-1}(\cdot)$ is nonlinear. Second, the predicted position and its covariance matrix of the token are computed. We have according to Eq. (3):

$$\hat{\mathbf{z}}_k = \mathbf{h}_k(\hat{\mathbf{x}}_{k|k-1}) \tag{8}$$

with an uncertainty of

$$\Lambda_k = H_k(\hat{\mathbf{x}}_{k|k-1}) P_{k|k-1} H_k^T(\hat{\mathbf{x}}_{k|k-1}) \ , \tag{9}$$

where

$$H_k(\hat{\mathbf{x}}_{k|k-1}) \triangleq \frac{\partial \mathbf{h}_k(\mathbf{x})}{\partial \mathbf{x}} \Big|_{\mathbf{x}=\hat{\mathbf{x}}_{k|k-1}} \ . \tag{10}$$

Here we again use the first order approximation if $\mathbf{h}_k(\cdot)$ is nonlinear.

Due to noise from multiple sources, it is very unlikely that a scene token observed at t_k has exactly $\hat{\mathbf{z}}_k$. Given n observed scene tokens at t_k

$\{{}_j\mathbf{z}_k \mid j = 1, \ldots, n\}$ with covariance matrices $\{{}_jR_k \mid j = 1, \ldots, n\}$, we use the Mahalanobis distance to decide which scene token matches the token having the predicted measurement vector $\hat{\mathbf{z}}_k$ with covariance matrix Λ_k.

The (squared) Mahalanobis distance between the prediction and the ith scene token is defined as

$$ {}_id_k^M \triangleq {}_i\mathbf{r}_k^T \Lambda_{{}_i\mathbf{r}_k}^{-1} {}_i\mathbf{r}_k \ , \tag{11}$$

where ${}_i\mathbf{r}_k = {}_i\mathbf{z}_k - \hat{\mathbf{z}}_k$ and $\Lambda_{{}_i\mathbf{r}_k} = {}_iR_k + \Lambda_k$. We usually call ${}_i\mathbf{r}_k$ the *measurement residual.* The variable ${}_id_k^M$ is a scalar random variate following a $\boldsymbol{\chi}^2$ distribution with q degrees of freedom, where q is the dimension of the measurement vector. By looking up the $\boldsymbol{\chi}^2$ distribution table, we can choose an appropriate threshold ϵ by setting $\Pr(\boldsymbol{\chi}_p^2 < \epsilon) = \alpha$, where α is typically equal to 95%. If ${}_id_k^M < \epsilon$, then the ith scene token is considered as a potential match of the token.

A naive matching algorithm yields a linear complexity in the number of scene tokens to match one token being tracked, i.e., $O(n)$. However, the matching process may be slow, especially when there is a large number of scene tokens. This is because the computation of the Mahalanobis distance involves a matrix inversion and is relatively expensive. Many techniques exist to speed up the matching process. One of them is the bucketing technique, which allow us to access directly a subset of scene tokens which are in the neighborhood of the prediction. See[22] for details. Another technique is proposed by Orr et al.[26], which uses the inequality

$$ \mathbf{r}^T \Lambda_{\mathbf{r}}^{-1} \mathbf{r} \geq \frac{\mathbf{r}^T \mathbf{r}}{\mathrm{trace}(\Lambda_{\mathbf{r}})} \ . \tag{12}$$

Thus we can first compute the simplified distance $d' = \mathbf{r}^T\mathbf{r}/\mathrm{trace}(\Lambda_{\mathbf{r}})$, which is computationally much simpler than Eq. (11). If $d' \geq \epsilon$, so will be ${}_id_k^M$, then the computation of Eq. (11) is not necessary. This avoids the necessity of performing a matrix inverse for every test.

Once a match is found, the (extended) Kalman filter is used to update the token parameters. Let ${}_i\mathbf{z}_k$ be the match of the token, the Kalman gain is first computed:

$$K_k = P_{k|k-1} H_k^T(\hat{\mathbf{x}}_{k|k-1})(\Lambda_k + {}_iR_k)^{-1} \ , \tag{13}$$

where H_k and Λ_k are given in (10) and (9). The state and its covariance matrix are eventually updated with the measurement as

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_{k|k-1} + K_k({}_i\mathbf{z}_k - \hat{\mathbf{z}}_k) \ , \tag{14}$$

$$P_k = [\mathbf{I} - K_k H_k(\hat{\mathbf{x}}_{k|k-1})] P_{k|k-1} \ . \tag{15}$$

4.2 Initialization

At time t_1, each scene token is used to initialize a token. As described earlier, the state of a token is composed of its position, orientation and kinematic parameters. The position and orientation parameters of a scene token are assigned to those of its corresponding token. The initialization of the kinematic parameters depends upon the a priori information. If such information is not available, it is reasonable to initialize them to zero, because we are considering a dense sequence and that the interframe motion is small. However, in the state covariance matrix, we should set the diagonal elements corresponding to the kinematic parameters to a fairly big number and the off-diagonal ones to zero, in order to reflect the fact that we know nothing about the kinematics of the token.

4.3 Appearance

Because some new tokens enter the field of view, their corresponding scene tokens in the current frame cannot be matched to any token being tracked. In this case, each such scene token is used to initialize a new token as described in the previous subsection, which starts the same process as the others.

5 Beam Search and Support of Existence

5.1 Different Cases in Matching and Beam Search Strategy

In using the criterion of the Mahalanobis distance, three cases occur in matching a token:

(i) Unique match: only one scene token is identified as a match of the token.

(ii) No match: no scene token is identified as a match of the token.

(iii) Multiple matches: several scene tokens are identified as plausible matches of the token.

If there is only one match, then there is a high probability that the scene token is the observation of the token being tracked. Thus we just update the token's state by incorporating the scene token.

"No-match" may occur due to a number of reasons. Firstly, the token being tracked disappears, i.e., it is out of the current field of view. This case can be easily verified by projecting the token onto one of the camera planes. Such a token may be retained or discarded depending upon whether we want to know it will reenter the field of view or not. Secondly, the token being tracked is resulted from previous false matches. Such a token has no reason to survive and should be discarded from further consideration. However, how to determine whether this is the case? Thirdly, the token being tracked is occluded by other tokens. Such a token should be retained for further consideration, because it corresponds to a real token in space and that it will be observed later. One may verify whether this is the case using geometric knowledge. However, the test may be computationally expensive. Finally, the token being tracked does exist, but the feature extraction process fails in reconstructing it. This is usually the case in Computer Vision. Although it is purely algorithmic, we can treat the token in the same manner as in the previous case where the problem is physical.

"Multiple-matches" occurs especially when a token is very uncertain (for example, during the first instants after initialization) or when several scene tokens are near to each other. One (maybe the most common) strategy is *best-first search*, that is, to choose the nearest scene token as in[15,16] and to discard the other possibilities. This method is efficient but not robust. It may lead to unpredictable results, because the closest scene token is not always the correct match. To increase the robustness, one may resort to backtrack. This, however, results in a loss of efficiency because we must save all previous data in the memory. Another possible strategy is to replace all scene tokens satisfy the criterion of the Mahalanobis distance by a virtual one with a modified probability distribution. This is the idea of the JPDAF method proposed by Bar-Shalom and Fortmann[1]. However, this method introduces a bias in the state estimate because it merges several physically distinct scene tokens as a single one to update the token's state.

An optimal assignment of which scene token to the token being tracked can be expected to be achieved through a global optimization by taking all tokens and their plausible matches into account. The computation, however, will be very expensive. The difficulty can be overcome if decision-making is deferred until more observations are collected. One approach is to first save several, say three, forthcoming frames, and then disambiguate multiple matches based on a temporal smoothness constraint by considering all possible combinations. This is the method used in[13]. However, it is not efficient enough, because it requires to store in the memory all data observed during several instants and to test all possible combinations.

A more efficient approach is to exploit the *beam-search strategy.* That is, instead of choosing the nearest scene token, several (2, in our implementation), if any, nearest ones are used. This approach is similar to the *track-splitting filter* in the literature[1]. Different from the JPDAF method, we split the token being tracked into several, as many as the scene tokens found in the search space. Each split token updates its state with one of the scene tokens chosen. We leave the forthcoming observations to decide which match is correct. The token resulted from the correct match will be confirmed by forthcoming scene tokens, while those resulted from incorrect matches will in general not. Thus the multiple-matches problem is handled gracefully. However, the algorithm is potentially exponential. Take an extreme example. We are given a sequence of observations of two tokens which are close to each other. Each token may be matched to either one, and the number of tokens being tracked at t_n would be 2^n. Some strategy needs to be developed to discard the false tokens.

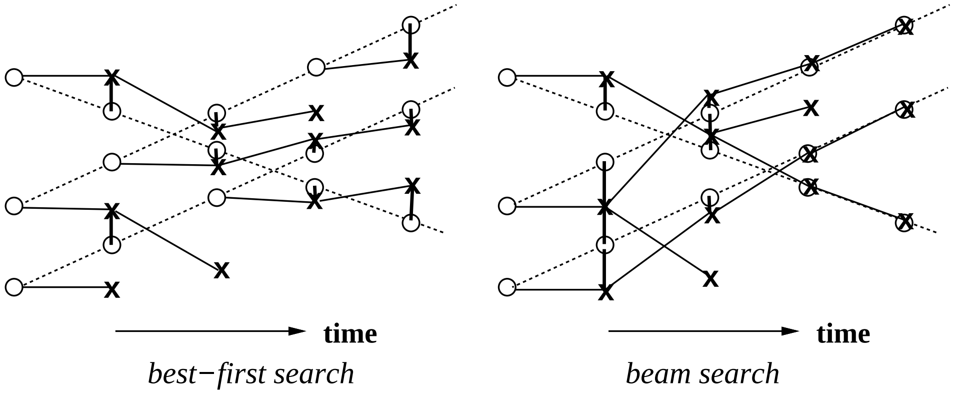


Fig. 1. A pedagogical example to show the strength of the beam search over the best-first search

In Fig. 1 we provide a pedagogical example to show the strength of the beam search over the best-first search. Originally there are three tokens whose trajectories are shown in dashed lines. The real position of a token is indicated by a small circle, and its prediction, by the symbol **x**. The association between the prediction and the observation is indicated by thick solid lines. The evolution of the tracking is shown by thin solid lines. Clearly, the beam-search approach yields much better performance than the best-first approach.

The multiple-hypothesis filter[1], originally developed by Reid[27], provides a consistent way to deal with multiple matches, token initialization and termination. However, the algorithm is much more complicated to implement and requires a number of parameters, e.g., the probabilities of detection and termination of a token and the probability of appearance of a new token. Furthermore, it has also exponential complexity. A practical implementation of this approach then exploits many heuristics, too.

5.2 Support of Existence

As described in the previous section, our idea of matching is to keep open the possibility of accepting several or no matches for any given token. However, such strategy may lead to a computational explosion. To avoid this we must discard tokens resulted from false matches. We compute for each token a number that we call its *support of existence* which measures the adequateness of the token with the measurements. We have already introduced this measure in[21], but for completeness we still include it here.

We denote the sequence of measurements corresponding to the token being tracked up to time t_k as $Z^k \triangleq \{\mathbf{z}_1, \ldots, \mathbf{z}_k\}$ in which $\mathbf{z}_i$ is the scene token observed at time t_i. Denote the event that Z^k yields a correct token, i.e., that its components $\mathbf{z}_i$ were produced by the same token 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):

$$L^k(e) = p(Z^k|e) = p[\mathbf{z}_1, \ldots, \mathbf{z}_k|e] \ . \tag{16}$$

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

$$L^k(e) = p[Z^{k-1}, \mathbf{z}_k|e] = p[\mathbf{z}_k|Z^{k-1}, e]\ p[Z^{k-1}|e] = \prod_{i=1}^{k} p[\mathbf{z}_i|Z^{i-1}, e] \ , \tag{17}$$

where Z^0 represents the prior information.

As before, we denote the measurement residual as $\mathbf{r}$, i.e., $\mathbf{r}_i = \mathbf{z}_i - \hat{\mathbf{z}}_i$. Then $p(\mathbf{r}_i) = N[\mathbf{r}_i; \mathbf{0}, \Lambda_{\mathbf{r}_i}]$ with $\Lambda_{\mathbf{r}_i} = \Lambda_i + R_i$. We use $N[\mathbf{x}; \bar{\mathbf{x}}, \Lambda]$ to denote the Gaussian density function of the random variable $\mathbf{x}$ with mean $\bar{\mathbf{x}}$ and covariance Λ. We now make the admittedly strong assumption that the $\mathbf{r}_i$'s are Gaussian and uncorrelated. We thus write:

$$p[\mathbf{z}_i | Z^{i-1}, e] = N[\mathbf{r}_i; \mathbf{0}, \Lambda_{\mathbf{r}_i}] \ . \tag{18}$$

It follows under the previous assumption that:

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

Note that $\mathbf{r}_i^T \Lambda_{\mathbf{r}_i}^{-1} \mathbf{r}_i = d_i^M$ (see Eq. (11)). The modified log-likelihood function, corresponding to the exponent of $L^k(e)$, is defined as

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

and can be computed recursively as follows:

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

The last term has a χ^2 distribution with q degrees of freedom. Since the $\mathbf{r}_i$'s are assumed to be independent, l_k has a χ^2 distribution with kq degrees of freedom.

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

$$l_k \leq \kappa \ , \tag{19}$$

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

In practice, the test (19) cannot be used for a long sequence because the likelihood function is dominated by old measurements and responds very slowly to recent ones. In order to limit the "memory" of the 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 $q(1+c)/(1-c)$ degrees of freedom, mean $q/(1-c)$, and variance $2q/(1-c^2)$. See[1] for the proof. In our implementation, $c = 0.75$, thus a bad token may last for 4 frames before being discarded.

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. In this case, it means that:

$$d_k^M \geq \epsilon \ .$$

Thus, if at time t_k no match is 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, i.e., if the token does not find any good match in the scene too often, then l_k will go beyond the threshold κ, and the token will be discarded, as expected. 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. In our implementation we set $d_k^M = \alpha\epsilon$ where $\alpha = 1.2$ in case of no match. α must be larger than 1 because no match is found within the threshold ϵ. The bigger the value of α is, the faster a token will be discarded if it cannot find any match at several consecutive time instants.

5.3 Discarding Redundant Tokens

In the beam-search approach, a token can be split, each being updated using a scene token satisfying the Mahalanobis distance. On the other hand, a scene token can be used to update several tokens being tracked. This occurs, for example, when a token splits into two (e.g., two fractions of a line segment are observed) and then both new tokens are updated with identical subsequent scene tokens. This implies that the state estimates of two or more tokens tokens may be similar, and it is likely that they represent the same token. We can thus just retain one token and discard the redundant ones.

6 Trying to Resolve Ambiguity as Early as Possible

Use of the support of existence does prevent the algorithm from a computational explosion. However, it is not efficient enough because we need to process a token resulted from previous false matches during four or more frames before it is discarded. It is of much benefit if we can resolve match ambiguities as early as possible. This section describes two strategies which reduce the match ambiguity and thus reduce the number of tokens to be processed.

6.1 Locking on a Reliable Token

Besides potentially computational explosion, one major drawback of beam-search approach is due to the fact that a scene token can be shared by several different tokens being tracked. Thus it is possible that this approach generates tokens which are not mutually exclusive, nor consistent with each other. The former was already discussed in the previous section (discarding redundant tokens). The latter is sometimes a desired feature, because if we have not enough information, it is wise to leave the forthcoming observations to resolve the ambiguity.

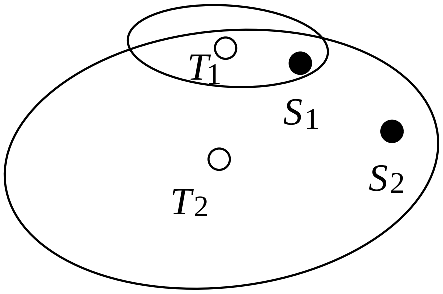


Fig. 2. A scene token can be locked by a secure token. ○: tokens; ●: scene tokens

However, in the situation as shown in Fig. 2, we can exploit a strategy, which we call the "locking-on-a-token", to obtain a better performance. Here two tokens share one of the measurement (S_1), and one token (T_1) has much less uncertainty than the other one (T_2). When a measurement (scene token) is validated by a secure token (T_1 in this case), whose state parameters are precise enough, the pairing is almost unambiguous. This measurement is said to be locked on by the token, and all other tokens search for their correspondences as if this measurement did not exist. In the situation as shown in Fig. 2, the scene token S_1 is locked on by the token T_1, and then the scene token S_2 is uniquely paired to the token T_2.

There are at least three ways to implement this strategy:

1. Comparing the uncertainty measures. The trace of the covariance matrix roughly measures the magnitude of the uncertainty. If $\mathrm{trace}(\mathrm{Cov}(T_1)) \ll \mathrm{trace}(\mathrm{Cov}(T_2))$ (e.g., $\mathrm{trace}(\mathrm{Cov}(T_1)) < \frac{1}{3}\mathrm{trace}(\mathrm{Cov}(T_2))$), then T_1 can lock on the shared scene token.
2. Counting the number of appearances. If during the past N (say, 5) frames, the number of appearance of T_1 (denoted by N_1) is much bigger than that

of T_2 (denoted by N_2), e.g., $N_1 \geq 4$ and $N_2 \leq 2$, then T_1 can lock on the shared scene token.

3. Comparing the support of existence l_k. A secure token implies that it has a high coincidence with the measurements, that is, it should have a low value of l_k (it has a high support for the existence). If the l_k of the token T_1 is less than $\lambda\kappa$, where κ is a threshold as defined in Eq. (19), and $\lambda < 1$ (we set $\lambda = 1/3$), then T_1 can lock on the shared scene token.

The third method has been implemented because the value of l_k is readily available.

6.2 Maximizing the Rigidity

Rigidity assumption has been used in most matching algorithms, especially in short-sequence motion analysis. Psychological study shows that, among many possible interpretations of any change between two successive frames, the human visual system only accepts a few, often only one, which are consistent with the rigidity assumption[3]. In long-sequence motion analysis like the problem studied in this paper, rigidity assumption is not exploited because the motion continuity or coherence is usually strong enough to resolve matching ambiguities. Here, we combine the rigidity and motion continuity to reduce the ambiguities.

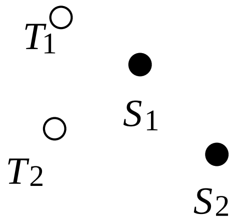


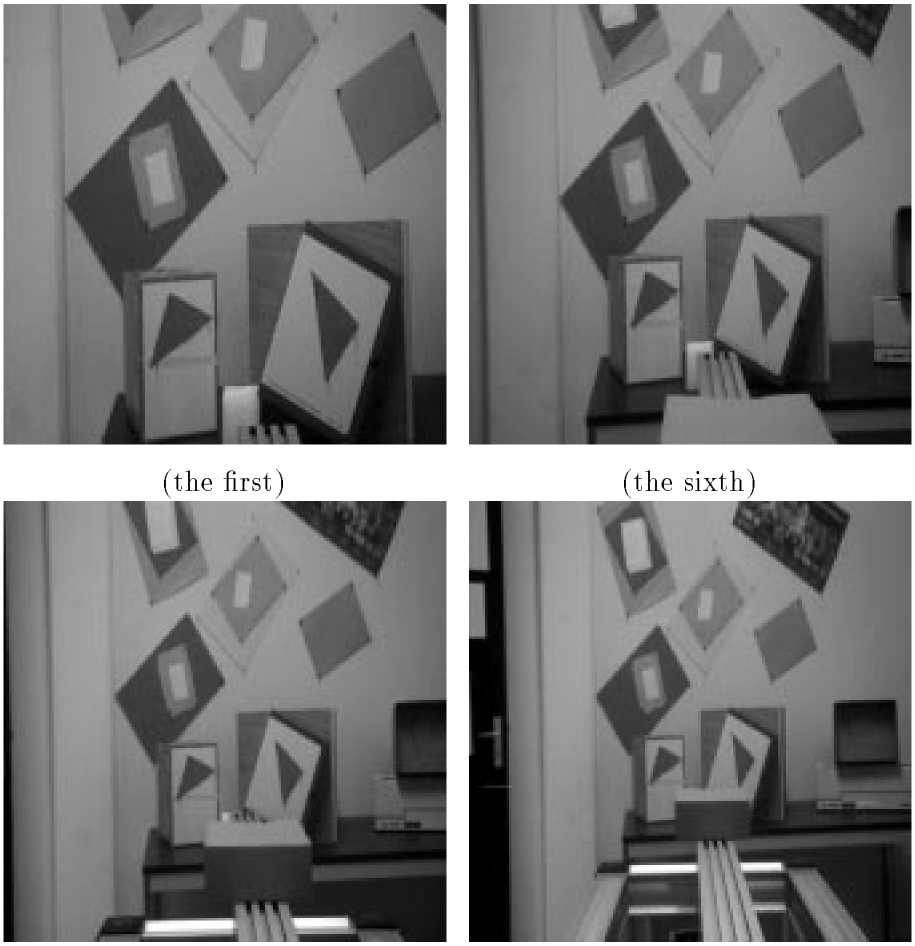
Fig. 3. Combining rigidity and motion continuity to disambiguate matches. ○: tokens; ●: scene tokens

Given a situation as shown in Fig. 3, where two tokens (T_1 and T_2) share the same measurements (S_1 and S_2). If we split tokens, we will obtain four tokens. If the relationship between S_1 and S_2 is not rigid compared with that between T_1 and T_2, then they originate from two different objects (or the object is deformed), and splitting is the only way we can do. However, if they satisfy the rigidity constraints, we can resolve the ambiguity using the motion continuity. The displacement of a rigid object between two successive frames in a sequence with high sample frequency cannot be large due to physical law. A reasonable constraint, for example, is that the rotation angle between two successive frames must be less than some threshold, say 60 degrees. To explain how to exploit this constraint, we refer to Fig. 3 and consider the two-dimensional case. If we assign S_1 to T_1 and S_2 to T_2, the rotation angle is about 45 degrees. On the other hand, if we assign S_1 to T_2 and S_2 to T_1, the rotation angle will be about 135 degrees, which is of course not reasonable. We thus resolve the ambiguity.

The reader is referred to[11,28] for a complete set of rigidity constraints for 3D line segments. As the data we have are always corrupted with noise, the equalities hardly ever hold true. We have formulated the rigidity constraints by explicitly taking into account the uncertainty of measurements. The reader is referred to[10,29,30] for other formalisms of rigidity constraints.

7 Experimental Results

We have incorporated the strategies described above into a tracking algorithm previously developed[20,21]. The algorithm tracks 3D line segments in a sequence of 3D frames reconstructed by a trinocular stereo system. It computes at the same time the 3D kinematic parameters for *each* line segment, and can segment the scene into objects by grouping line segments based on motion similarity.



(the first)

(the sixth)

(the eleventh)

(the sixteenth)

Fig. 4. Sample images of the stereo sequence studied

Table 1. The numbers of scene tokens and active tokens in each frame

frame number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
scene tokens	37	42	42	44	43	43	48	50	51	58	66	73	102	101	98	103
tokens (`modified`)	37	54	61	51	51	50	52	60	61	69	79	86	115	134	146	147
tokens (`original`)	37	68	78	63	61	56	59	70	74	80	98	119	147	179	190	192

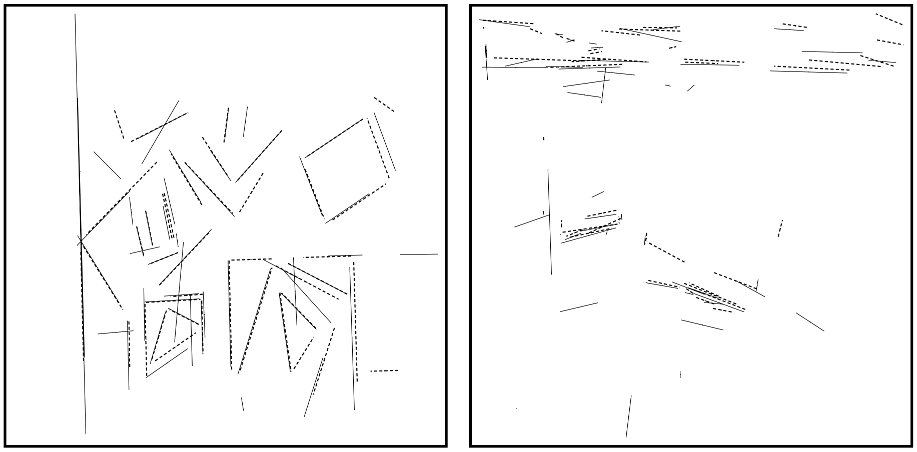
We have tested the modified algorithm on the sequences described in[20,21], and better results have been obtained. In this paper, we provide the results on a different sequence, consisting of 16 triplets of images. The 1st, 6th, 11th and 16th images taken by the first camera of the stereo rig are shown in Fig. 4. The sequence was acquired by manually moving the stereo rig away from a wall on which we have put several posters to increase the number of line segments. The interframe displacement was supposed a pure translation of 10 centimeters. It is in fact almost true except for the thirteenth frame, as can be seen later. This sequence is interesting in that more and more tokens are visible when time goes on, i.e., the appearance is remarkable. (Several line segments are not observable in the 3D frames due to the *absence problem* described in the introduction section.) If we process the sequence in the reverse direction, more and more tokens would disappear. However, the appearance problem is more difficult to tackle than the disappearance in tracking. The number of line segments reconstructed by the stereo system in each frame is shown in the first row of Table 1.

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 inter-frame 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 (or 5 degrees/unit-time), and that for each translational velocity component is 150 millimeters/unit-time.

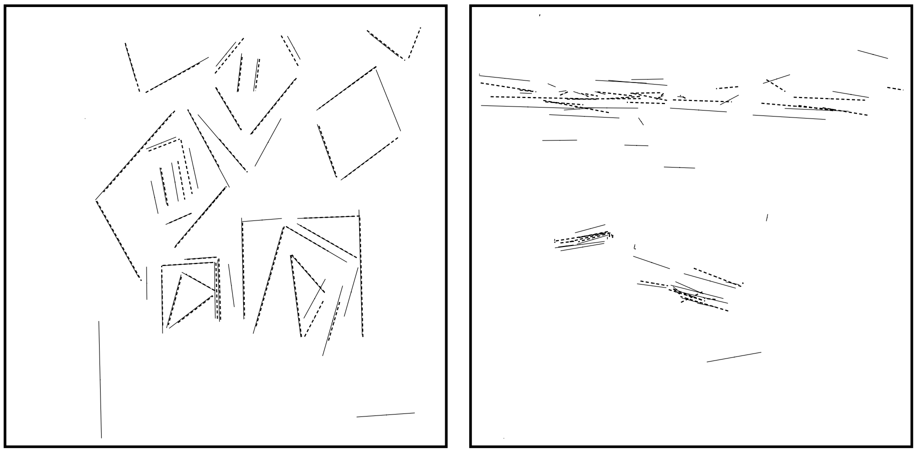
Those tokens are then predicted for the next instant t_2 and the predicted tokens are compared with those in the new frame. Of course, since we assume that there is no motion, the predicted position and orientation of each token remain unchanged, but their uncertainty changes and becomes very large. 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 scene token is not always the correct match. We retain the two best matches if a token has multiple matches. Furthermore, the strategies described in this paper are exploited to reduce the matching ambiguities. The token updates its kinematic parameters using its best match. A new token is initialized by combining the original token and its second best match which is used to estimate its kinematic parameters. It pursues the tracking in the same manner as the others. Usually the tokens originated from false matching in the preceding instants are losing their support for existence as more frames are processed, and are eventually deactivated. The number of active tokens after processing each frame is shown in the second row of Table 1. The number does not become overwhelming, even though there is a significant increase in the number of scene tokens.

In Fig. 5, we show the superposition of the predicted (in solid lines) and the observed (in dashed lines) segments at time t_3. (*Note:* In figures 5 through 8, the left picture is a perspective view of a 3D frame, that is, its perspective projection onto the first image plane; the right one is the top view, that is, its orthographic projection onto the ground plane.) In the top view, the segments on the top correspond to the projections of the posters on the wall; those in the middle correspond to the boxes on the table. 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 the tokens originated from good



(Perspective view) (Top view)

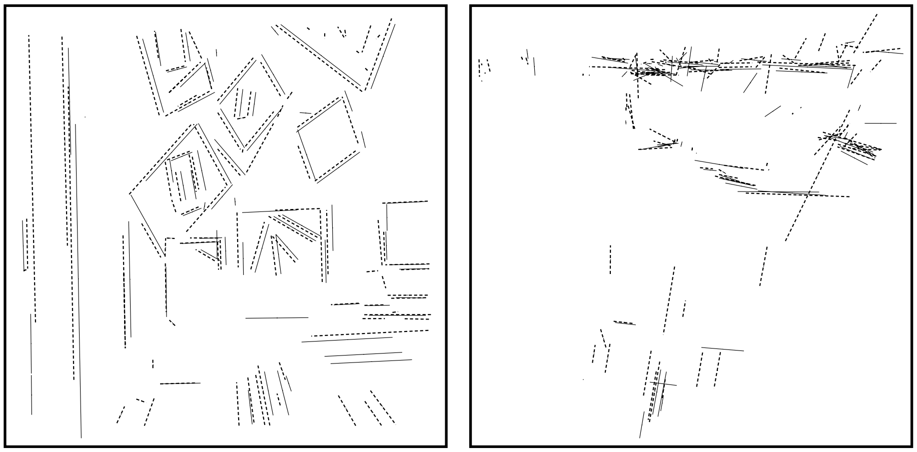
Fig. 5. The superposition of the predicted (in solid lines) and the observed (in dashed lines) segments at time t_3



(Perspective view) (Top view)

Fig. 6. The superposition of the predicted (in solid lines) and the observed (in dashed lines) segments at time t_5

matching coincide well with the scene tokens. After having processed the fourth frame, a number of false tokens disappear, as shown in Fig. 6, where the predictions for t_5 are overlayed on the observations at t_5.

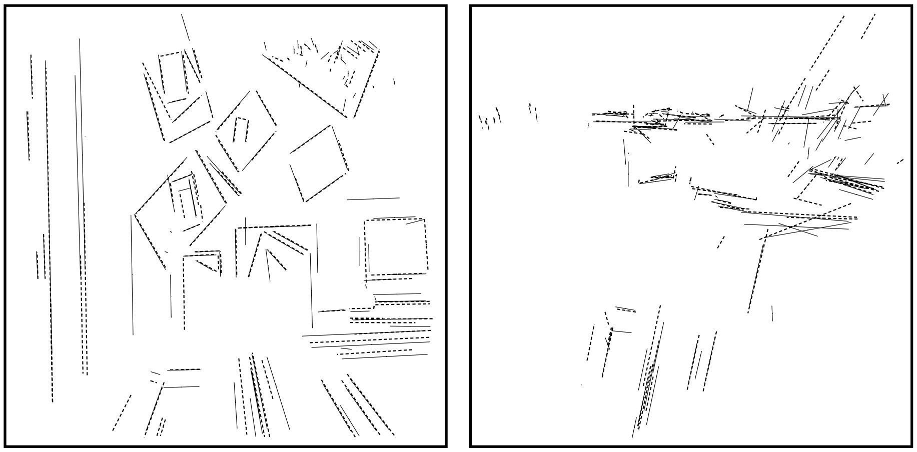


(Perspective view) (Top view)

Fig. 7. The superposition of the predicted (in solid lines) and the observed (in dashed lines) segments at time t_{13}

As said earlier, the thirteenth frame was taken, not intentionally, in a shifted position. Figure 7 shows the superposition of the predicted (in solid lines) and the observed (in dashed lines) segments at time t_{13}. Compared with the results shown in Fig. 5 and Fig. 6, we can observe a relatively big difference between the prediction and the observation. After several frames, such occasional incoherent motion will be compensated for by the algorithm. Figure 8 shows the superposition of the predicted (in solid lines) and observed (in dashed lines) segments at t_{16}. Quite a good fitting between the prediction and observation is now again observed.

As described in [20,21], we can group the individual tokens into objects based on the motion coherence. Here there is only one object. The final estimate of the interframe rotation is 1.1 milliradians, or 0.063 degrees. The



(Perspective view) (Top view)

Fig. 8. The superposition of the predicted (in solid lines) and the observed (in dashed lines) segments at time t_{16}

final estimate of the interframe translation is 99.52 millimeters. Recall that the supposed displacement is a pure translation of 100 millimeters.

To compare the performance of the original and the modified algorithm, we rerun the algorithm without using the strategies described in the section "Trying to resolve ambiguity as early as possible", i.e., *locking-on-a-token* and *maximizing the rigidity*. The third row of Table 1 shows clearly that the number of tokens being tracked at each instant with the original algorithm is larger than with the modified one. Indeed, the number of tokens is 30% larger in average. The difference becomes even bigger as more frames are processed. This indicates the necessity of exploiting the strategies described. Because it tracks less, but more reliable, tokens, the modified algorithm is also computationally more efficient. For example, if we run both algorithms on a SUN SS10 workstation (about 5.4 MFLOPS), the CPU time for processing frame 15 is shown in Table 2. The execution time with the modified algorithm is about 30% less.

Table 2. Comparison of computational costs of the original and modified algorithms (CPU time in seconds for Frame 15)

Algorithm	prediction time	matching-update time	total
modified	0.11	0.25	0.36
original	0.16	0.31	0.47

8 Conclusion

In this paper we have presented our recent work on token tracking in a cluttered scene in the statistical data association framework. A general formulation has been described. The main steps have been summarized. We have focused in this paper on several strategies including beam search for resolving multiple matches, support of existence for discarding false matches, locking tokens and maximizing local rigidity for handling combinatorial explosion. We have implemented these strategies in a 3D line segment tracking algorithm and found them to be very useful. Indeed, the number of tokens has been reduced by 30% by exploiting these strategies, and the modified algorithm is computationally more efficient.

References

[1] Y. Bar-Shalom and T. Fortmann, *Tracking and Data Association.* Academic, New York, 1988.

[2] S. S. Blackman, *Multiple-Target Tracking with Radar Application.* Artech House, Norwood, MA, 1986.

[3] S. Ullman, *The Interpretation of Visual Motion.* MIT Press, Cambridge, MA, 1979.

[4] R. Tsai and T. Huang, "Estimating 3-D motion parameters of a rigid planar patch, i," *IEEE Trans. ASSP*, vol. 29, pp. 1147–1152, December 1981.

[5] H. Longuet-Higgins, "A computer algorithm for reconstructing a scene from two projections," *Nature*, vol. 293, pp. 133–135, 1981.

[6] R. Haralick and L. Shapiro, "The consistent labeling problem: Part i," *IEEE Trans. PAMI*, vol. 1, pp. 129–139, April 1979.

[7] R. Hummel and S. Zucker, "On the foundation of relaxation labeling process," *IEEE Trans. PAMI*, vol. 5, pp. 267–286, May 1983.

[8] O. Faugeras and M. Berthod, "Improving consistency and reducing ambiguity in stochatic labeling: An optimization approach," *IEEE Trans. PAMI*, vol. 3, pp. 412–423, April 1981.

[9] B. Radig, "Image sequence analysis using relational structures," *Pattern Recog.*, vol. 17, no. 1, pp. 161–167, 1984.

[10] H. Chen and T. Huang, "Maximal matching of 3-D points for multiple-object motion estimation," *Pattern Recog.*, vol. 21, no. 2, pp. 75–90, 1988.

[11] Z. Zhang, O. Faugeras, and N. Ayache, "Analysis of a sequence of stereo scenes containing multiple moving objects using rigidity constraints," in *Proc. Second Int'l Conf. Comput. Vision*, (Tampa, FL), pp. 177–186, December 1988. Also as a chapter in R. Kasturi and R.C. Jain (eds), *Computer Vision: Principles*, IEEE computer society press, 1991.

[12] A. van Doorn and J. Koenderink, "Spatiotemporal integration in the detection of coherent motion," *Vision Research*, vol. 24, no. 1, pp. 47–53, 1984.

[13] M. Jenkin and J. Tsotsos, "Applying temporal constraints to the dynamic stereo problem," *Comput. Vision, Graphics Image Process.*, vol. 24, pp. 16–32, 1986.

[14] S. Sethi and R. Jain, "Finding trajectories of feature points in a monocular image sequence," *IEEE Trans. PAMI*, vol. 9, no. 1, pp. 56–73, 1987.

[15] J. Crowley, P. Stelmaszyk, and C. Discours, "Measuring image flow by tracking edge-lines," in *Proc. Second Int'l Conf. Comput. Vision*, (Tampa, FL), pp. 658–664, IEEE, December 1988.

[16] R. Deriche and O. Faugeras, "Tracking line segments," in *Proc. First European Conf. Comput. Vision*, (O. Faugeras, ed.), (Antibes, France), pp. 259–268, Springer, Berlin, Heidelberg, April 1990.

[17] Z. Zhang and O. Faugeras, "Tracking and motion estimation in a sequence of stereo frames," in *Proc. 9th European Conf. Artif. Intell.*, (L. Aiello, ed.), (Stockholm, Sweden), pp. 747–752, August 1990.

[18] I. Cox, "A review of statistical data association techniques for motion correspondence," *Int'l J. Comput. Vision*, vol. 10, no. 1, pp. 53–66, 1993.

[19] J. Uhlmann, "Algorithms for multiple-target tracking," *American Scientist*, vol. 80, pp. 128–141, March-April 1992.

[20] Z. Zhang and O. Faugeras, "Tracking and grouping 3D line segments," in *Proc. Third Int'l Conf. Comput. Vision*, (Osaka, Japan), pp. 577–580, IEEE, December 1990.

[21] Z. Zhang and O. Faugeras, "Three-dimensional motion computation and object segmentation in a long sequence of stereo frames," *Int'l J. Comput. Vision*, vol. 7, pp. 211–241, March 1992.

[22] Z. Zhang, *Motion Analysis from a Sequence of Stereo Frames and its Applications.* PhD thesis, University of Paris XI, Orsay, Paris, France, 1990. in English.

[23] A. Jazwinsky, *Stochastic Processes and Filtering Theory.* Academic, New York, 1970.

[24] P. Maybeck, *Stochastic Models, Estimation and Control.* Vol. 1, Academic, New York, 1979.

[25] P. Maybeck, *Stochastic Models, Estimation and Control.* Vol. 2, Academic, New York, 1982.

[26] M. Orr, J. Hallam, and R. Fisher, "Fusion through interpretation," in *Proc. Second European Conf. Comput. Vision*, (Santa Margherita Ligure, Italy), pp. 801–805, May 1992.

[27] D. Reid, "An algorithm for tracking multiple targets," *IEEE Trans. AC*, vol. 24, pp. 843–854, Dec. 1979.

[28] Z. Zhang and O. Faugeras, "Estimation of displacements from two 3D frames obtained from stereo," *IEEE Trans. PAMI*, vol. 14, pp. 1141–1156, December 1992.

[29] S. Pollard, J. Porrill, J. Mayhew, and J. Frisby, "Matching geometrical descriptions in three-space," *Image and Vision Computing*, vol. 5, pp. 73–78, may 1987.

[30] D. Murray and D. Cook, "Using the orientation of fragmentary 3D edge segments for polyhedral object recognition," *Int'l J. Comput. Vision*, no. 2, pp. 153–169, 1988.

Unité de recherche INRIA Lorraine, Technôpole de Nancy-Brabois, Campus scientifique,
615 rue de Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unité de recherche INRIA Rennes, IRISA, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unité de recherche INRIA Rhône-Alpes, 46 avenue Félix Viallet, 38031 GRENOBLE Cedex 1
Unité de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105,
78153 LE CHESNAY Cedex
Unité de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS
Cedex

Éditeur

INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex

(France)

ISSN 0249-6399