Pergamon

Pattern Recognition, Vol. 31, No. 3, pp. 333–344, 1998

0031-3203/98$17.00 + .00

PII: S0031-3203(97)00044-7

VEHICLE-TYPE MOTION ESTIMATION BY THE FUSION OF IMAGE POINT AND LINE FEATURES

SOO-CHANG PEI* and LIN-GWO LIOU

Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan, R.O.C.

(*Received* 14 *November* 1994; *received for publication* 7 *April* 1997)

Abstract—Three-dimensional (3D) motion estimation is a very important topic in machine vision. However, reliability of the estimated 3D motion seems to be the most challenging problem, especially to the linear algorithms developed for solving a general 3D motion problem (six degrees of freedom). In real applications such as the traffic surveillance and auto-vehicle systems, the observed 3D motion has only three degrees of freedom because of the ground plane constraint (GPC). In this paper, a new iterative method is proposed for solving the above problem. Our method has several advantages: (1) It can handle both the point and line features as its input image data. (2) It is very suitable for parallel processing. (3) Its cost function is so well-conditioned that the final 3D motion estimation is robust and insensitive to noise, which is proved by experiments. (4) It can handle the case of missing data to a certain degree. The above benefits make our method suitable for a real application. Experiments including simulated and real-world images show satisfactory results. © 1997 Pattern Recognition Society. Published by Elsevier Science Ltd.

Image points Image lines Traffic surveillance system Ground plane constraint

1. INTRODUCTION

The problem of estimating 3D motion/structure parameters from an image sequence has been the focus of a significant amount of research during the past years. In several industrial applications, the ability to perform detection and estimation of motion has become a basic task of the robotic vision system. The difficulty of motion analysis may be broken into two parts: (1) measurement—to extract image features and measure their 2D motions on the image plane. (2) 3D motion estimation—from the above 2D measurements to estimate 3D motion and infer the object structure. The second part is the main concern of this paper.

To solve the above 3D motion estimation problem, the so-called correspondence approaches are often utilized. They analyze discrete motion measurement of the same physical features (points, lines, ... , etc.) over time. Besides, they often assume that the 3D motion of the target is rigid. By the image features and strategies they used, we briefly describe these techniques.

Roach and Aggarwal[1] proposed an algorithm of 3D motion estimation using point correspondences. It is based on the distance invariance of the points of a 3D rigid object. This method is nonlinear and therefore initial guesses or global search have to be adopted. Besides, several linear algorithms were proposed. Examples include Longuet-Higgins,[2] Tsai and Huang,[3] Zhuang,[4] Weng and Huang,[5] Philip,[6] and Spetsakis and Aloimonos.[7] The main step of these linear algorithms is to solve an intermediate matrix linearly from point correspondences. Then the motion parameters will be determined by decomposing this intermediate matrix. These linear methods need at least eight point correspondences at two views and they are sensitive to noise. Several methods about point tracking are listed in references (8–11).

Compared with the point features, the line features have better performance in the reliability of measurement. Measurement of a line feature can easily achieve the sub-pixel accuracy. However, a line feature reveals less 3D motion clues than that from a point feature. So it often needs more sets of line correspondences in more views (than points) to solve a 3D motion problem. The algorithm proposed by Yen and Huang[12] is based on projecting image lines on a unit sphere; the 3D rotations are estimated iteratively from line correspondences over three frames. Mitichi, Seida and Aggarwal[13] used angle invariance of any two lines of a 3D rigid object; this algorithm used rigidity constraint to reconstruct the 3D lines and then computes the 3D motion parameters. Lately, linear algorithms for lines are proposed. Their basic derivations are very similar to that for points. Examples include Liu and Huang,[14–15] Spetsakis and Aloimonos,[16] Weng, Huang and Ahuja.[17] These linear algorithms need at least 13-line correspondences at three views and they are very sensitive to noise. Some methods about tracking a line feature are listed in references (18–21).

In industrial applications, reliability of an algorithm should be highly considered. To achieve high robustness, three possible strategies were often utilized.

- To use long image sequence and local smoothness 3D motion constraint. Examples include Broida

*Author to whom correspondence should be addressed.

and Chellappa,[22] Weng, Huang and Ahuja,[23] Tseng and Sood,[24] Shariat and Price,[25] and Broida and Chellappa,[26] and Hu and Ahuja.[27] Most of them used point correspondences only.

- To use as many available image features as possible. For example, Wang, Karandikar and Aggarwal[28] and Liu, Huang and Faugeras[29] considered the use of both image point and line features as their input.
- To use additional physical constraints in some special applications. Nonlinearity of the motion analysis will be greatly simplified, and the number of unknowns will be reduced. Examples include the ground plane constraint (GPC) in the traffic surveillance system or the auto-navigation system (see Fig. 1).[30,31]

Inspired by the above ideas, a new iterative algorithm is proposed in this paper. It considers a special vehicle-type motion problem. A closed-form solution is also derived as an initial estimate to our algorithm. This initial solution is precisely accurate when no errors exist.

Our algorithm has several important properties: (1) Both the image points and lines can be accepted as the input of the algorithm. (2) It minimizes a 2D cost function to find the optimum solution. (3) The definition of the cost function is especially taken care such that the cost function is well-shaped (or well-conditioned). (4) It is very suitable for parallel processing. (5) Our algorithm can be easily extended to a long image sequence and it can handle missing data to a certain degree.

To complete the algorithm, several important issues are discussed: (1) Weighting problem. (2) The smallest number of required image features. (3) The case when no rotation exists. Experiments, including both the simulated and real-world image tests, prove the correctness and robustness of this algorithm.

To stress the difference between our iterative method and the traditional linear one [especially to the Spetsakis's method[33]], both of them are compared in the following directions:

- *Technique.* Basically, the linear method is indirect. No matter how the motion type is constrained, 27 intermediate variables should be first solved anyway. It means that we should use more redundant image features than that required (and even larger for the robustness of solution). Finally, the 27 intermediate variables are still be sent to a nonlinear optimization process for further refinement. On the contrary, our method is a direct one. We do not have to solve the intermediate variables; therefore, the redundant image features are not so badly required.
- *Accuracy.* Of no doubt, solution to the intermediate variables will be very error-sensitive for ignoring the nonlinear constraints among them. Besides, finding the best-fit motion parameters to minimize the residual errors of the 27 intermediate variables is not a very good idea. Constructing a good optimization criterion needs to consider the noise model of image features and the balance of weighting. The linear method never mentioned about that. On the contrary, in the vehicle-type motion problem, our method does consider all of the above factors.
- *Practical use.* In the vehicle-type 3D motion problem, our method only needs to search a two-variable cost function (about rotation). The linear method may have a large advantage in the theoretical analysis of general 3D-motion problem, but it may be not the most appropriate method for the vehicle-type 3D motion estimation.

On the whole, we can say: "The linear method is designed for speed improvement and theoretical analysis, but ours is designed for accuracy improvement and real application ".

The remainder of this paper is organized as follows: Section 2, describe several basic transform relationships between the camera coordinate system and the global coordinate system. Section 3, problem formulation. Section 4, description of the main algorithm. Section 5, discussions about several related problems of the algorithm. Section 6, experiments including simulated and real-world image. Section 7, final conclusion.

2. COORDINATE TRANSFORM

Without loss of generality, let us consider a traffic surveillance system shown in Fig. 1. There are two coordinate systems: ground plane coordinate system (GPCS) and camera coordinate system (CCS). The origin O of GPCS is just lying on this plane. $\{\hat{\mathbf{x}}, \hat{\mathbf{y}}, \hat{\mathbf{z}}\}$ forms an orthonormal basis of GPCS, and the first two orthonormal vectors ($\hat{\mathbf{x}}$ and $\hat{\mathbf{y}}$) span the whole ground plane. The third unit vector $\hat{\mathbf{z}}$ is then the normal vector of the ground plane. For convenience, the GPCS is considered as the global coordinate system in this paper. The camera is placed at a suitable position such that it can see the objects moving on the ground plane. The origin F of CCS is positioned at $\mathbf{P}_c \equiv (P_{cX}, P_{cY}, P_{cZ})^{\mathrm{T}}$ (relative to GPCS). Its three orthonormal vectors, $\{\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3\}$, specify the pose of the camera. $\mathbf{u}_3$ is the camera's main axis; focal length $\overline{Fo}$ is set to 1; $\mathbf{u}_1$ and $\mathbf{u}_2$ are the two directions of the image axes.

A point in 3D space is separately denoted by P and Q in GPCS and CCS. We can easily derive the coordinate transform relationship between GPCS and CCS as follows:

$$\mathbf{P} = \mathbf{R}_c\mathbf{Q} + \mathbf{P}_c \text{ or } \mathbf{Q} = \mathbf{R}_c^T(\mathbf{P} - \mathbf{P}_c), \tag{1}$$

where the matrix $\mathbf{R}_c$ is an orthonormal matrix and defined as

$$\mathbf{R} \equiv [\mathbf{u}_1|\mathbf{u}_2|\mathbf{u}_3]. \tag{2}$$

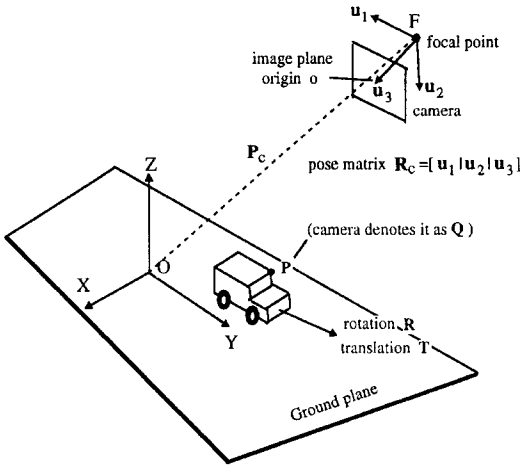

Fig. 1. Configuration of a traffic surveillance system.

Assume that the vehicle-type motion is specified by a rotation matrix **R** and a translation vector **T** like this

$$\mathbf{P}' = \mathbf{RP} + \mathbf{T}, \tag{3}$$

where

$$R = \begin{bmatrix} \cos\omega & -\sin\omega & 0 \\ \sin\omega & \cos\omega & 0 \\ 0 & 0 & 1 \end{bmatrix}; \quad \mathbf{T} = \begin{bmatrix} T_X \\ T_Y \\ 0 \end{bmatrix}. \tag{4}$$

The same 3D motion observed by the camera (denoted by "*") can be represented by

$$\mathbf{Q}' = \mathbf{R}^*\mathbf{Q} + \mathbf{T}^*. \tag{5}$$

$\mathbf{R}^*$ and $\mathbf{T}^*$ also have the following special forms:

$$\mathbf{R}^* = \mathbf{R}_c^{\mathrm{T}}\mathbf{R}\mathbf{R}_c; \quad \mathbf{T}^* = \alpha\mathbf{m}_x + \beta\mathbf{m}_y, \tag{6}$$

where $\mathbf{m}_x \equiv \mathbf{R}_c^{\mathrm{T}}\hat{\mathbf{x}}$, $\mathbf{m}_y \equiv \mathbf{R}_c^{\mathrm{T}}\hat{\mathbf{y}}$, $\mathbf{m}_z \equiv \mathbf{R}_c^{\mathrm{T}}\hat{\mathbf{z}}$, and

$$\alpha \equiv P_{cX}(-1+\cos\omega) - P_{cY}\sin\omega + T_X;$$

$$\beta \equiv P_{cX}(\sin\omega) + P_{cY}(-1+\cos\omega) + T_Y. \tag{7}$$

For a simpler form in derivation, $\mathbf{T}^*$ can be rewritten as [from equation (6)]

$$\mathbf{T}^* = [\mathbf{m}_x|\mathbf{m}_y]\begin{bmatrix}\alpha \\ \beta\end{bmatrix} \equiv \mathbf{Ma}, \tag{8}$$

where **M** is a 3×2 matrix and **a** is a 2×1 vector.

The above equations define the transform relationships between GPCS and CCS. The three motion parameters ω, T_X, and T_Y defined in GPCS are directly related to the moving velocity of the 3D object along the ground plane (road). In fact, with a minor modification, the above equations are also suitable for the auto-navgation system.

3. PROBLEM FORMULATION

Let us consider the monocular traffic surveillance system shown in Fig. 1. We assume that the image point and line features corresponding to a 3D moving vehicle can be continuously tracked at three image frames ($t = -1$, $t = 0$, $t = +1$). Figure 2 shows the definitions for motion parameters: ω_- and $\mathbf{T}_-$ for time instants 0 and -1; ω_+ and $\mathbf{T}_+$ for time instants 0 and $+1$. The pose $\mathbf{R}_c$ and position $\mathbf{P}_c$ of the camera has been *known* as prior information. The camera obeys the rule of perspective projection.

Now the question is: "*How to determine the unknown 3D motion parameters* $\{\omega_-, \omega_+, \mathbf{T}_-, \mathbf{T}_+\}$ *from the input image data?*"

Before leaving this section, we briefly explain why we need three frames to solve this problem. To the image point features, two frames are just enough for the 3D motion estimation; however, it takes at least three for the image line features to do the same thing. In order to fuse the contributions from both image points and lines, three image frames are grouped together and considered as a basic processing unit in our method.

4. MAIN ALGORITHM

To present our method in an organized form, we divide its derivations into the following six subsections (4.1–4.6):

4.1. Definition of a general cost function

The purpose of our algorithm is to find the optimum 3D motion parmeters which can minimize the value of a given cost function J. Assume that we have tracked N_p image points and N_l image lines in three frames. Therefore, the cost function J must be divided into two parts: one for the point features, and one for the line features. It can be defined as

$$J \equiv \sum_{i=1}^{N_p} J_{p,i} + \sum_{j=1}^{N_l} J_{l,j}, \tag{9}$$

where the subscripts p and l separately represent "point" and "line" features. If three is no available image line feature (or points) in the input data, we may neglect all of the cost terms $J_{l,j}$s (or $J_{p,i}$s). Therefore, the definition of cost function in equation (9) is very flexible.

4.2. Cost function for image points

Here we will define the cost function $J_{p,i}$ for $i = 1$ to N_p. To simplify the notations, the subscript i is temporarily neglected in the following derivation. Let us consider Fig. 3. A moving-feature point Q in 3D space is separately denoted by $\mathbf{Q}_-$, $\mathbf{Q}_0$, and $\mathbf{Q}_+$ at the three interested time instants $t = -1$, $t = 0$, and $t = +1$. For the purpose of proper weighting (explained later), their projected image points are represented by three

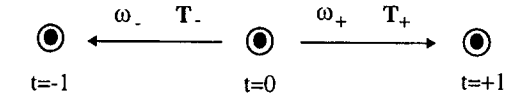

Fig. 2. Definition of motion parameters at three time instants.

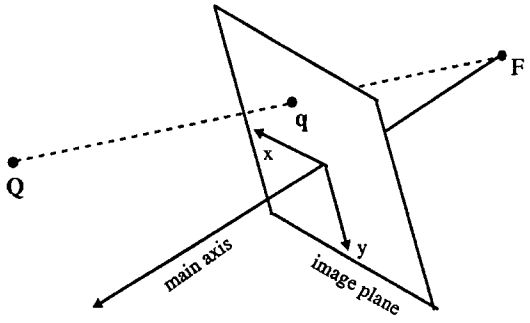

Fig. 3. Perspective projection for a point feature.

unit vectors $\mathbf{q}_-$, $\mathbf{q}_0$, and $\mathbf{q}_+$ (from $[x, y, 1]^T$ to $[x, y, 1]/\sqrt{x^2+y^2+1}$).

Under the perspective projection, it is easy to observe that $\mathbf{Q}_k$ ($k = -, 0, +$) and its projection $\mathbf{q}_k$ are of the same direction. So the cost function J_p can be defined as

$$J_p \equiv w_p\{\|\mathbf{q}_- \times \mathbf{Q}_-\|^2 + \|\mathbf{q}_0 \times \mathbf{Q}_0\|^2 + \|\mathbf{q}_+ \times \mathbf{Q}_+\|^2\}, \tag{10}$$

where w_p (subscript "p" denotes "point") is a weighting factor. The operator "$\times$" denotes the outer product of vectors.

According to equations (5), (6) and (8), we have

$$\mathbf{Q}_- = \mathbf{R}_-^*\mathbf{Q}_0 + \mathbf{T}_-^* \; ; \quad \mathbf{Q}_+ = \mathbf{R}_+^*\mathbf{Q}_0 + \mathbf{T}_+^*, \tag{11}$$

where $\mathbf{T}_k^* = \mathbf{M}\mathbf{a}_k$ ($k = -, +$).

Therefore, J_p is in fact a function of ω_-, ω_+, $\mathbf{a}_-$, $\mathbf{a}_+$, and $\mathbf{Q}_0$. To minimize J_p, we substitute equation (11) into equation (10) and set $\partial J_p/\partial \mathbf{Q}_0$ to zero. An optimum $\mathbf{Q}_0$ can be solved in terms of a given set of motion parameters

$$\mathbf{Q}_0 = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}, \tag{12}$$

where

$$A \equiv \sqrt{w_p}\begin{bmatrix}\mathbf{E}_-\mathbf{R}_-^* \\ \mathbf{E}_0 \\ \mathbf{E}_+\mathbf{R}_+^*\end{bmatrix}; \quad \mathbf{b} \equiv -\sqrt{w_p}\begin{bmatrix}\mathbf{E}_-\mathbf{M}\mathbf{a}_- \\ \mathbf{0} \\ \mathbf{E}_+\mathbf{M}\mathbf{a}_+\end{bmatrix}. \tag{13}$$

The matrix E_k ($k = -, 0, +$) is defined as

$$\mathbf{E}_k \equiv \begin{bmatrix} 0 & -q_{k,z} & q_{k,y} \\ q_{k,z} & 0 & -q_{k,x} \\ -q_{k,y} & q_{k,x} & 0 \end{bmatrix}. \tag{14}$$

When substituting the best $\mathbf{Q}_0$ in equation (12) into the cost function J_p, we have

$$J_p = \mathbf{b}^T(\mathbf{I} - \mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T)\mathbf{b} \equiv \mathbf{b}^T\mathbf{D}\mathbf{b}. \tag{15}$$

Notice that the matrix $\mathbf{D}$ only depends on the unknown rotation parameters, ω_- and ω_+.

For a simpler mathematical form, the vector $\mathbf{b}$ can be rearranged as

$$\mathbf{b} = -\sqrt{w_p}\begin{bmatrix}\mathbf{E}_-\mathbf{M} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{E}_+\mathbf{M}\end{bmatrix}\begin{bmatrix}\mathbf{a}_- \\ \mathbf{a}_+\end{bmatrix} \equiv \mathbf{H}\mathbf{a}. \tag{16}$$

Here $\mathbf{H}$ is a 9×4 matrix, and it depends only on the input image data; $\mathbf{a}$ is a 4×1 vector.

Substituting equation (16) into equation (15), we have

$$J_p = \mathbf{a}^T(\mathbf{H}^T\mathbf{D}\mathbf{H})\mathbf{a} \equiv \mathbf{a}^T\mathbf{C}_p\mathbf{a}. \tag{17}$$

Two things are obvious: (1) the 4×4 symmetric matrix C_p is a function of only two unknowns, ω_- and ω_+; (2) the cost function J_p is a quadratic form of $\mathbf{a}$.

Each set of the corresponding image points can provide a cost function like equation (17). Adding these cost functions together, we have the final form

$$\sum_{i=1}^{N_p} J_{p,i} = \mathbf{a}^T\left[\sum_{i=1}^{N_p} C_{p,i}(\omega_-, \omega_+)\right]\mathbf{a} \equiv \mathbf{a}^T\mathbf{B}_p\mathbf{a}. \tag{18}$$

4.3. *Cost function for image lines*

Here we will define the cost function $J_{l,j}$ for $j = 1$ to N_l. The subscript j is also neglected for simplifying the notations. Let us consider Fig. 4. A moving feature line L in 3D space is separately denoted by L_-, L_0, and L_+ at the three interested time instants $t = -1$, $t = 0$, and $t = +1$. Their projected image lines are separately denoted by l_ks ($k = -, 0, +$). The corresponding 3D points on the feature lines are denoted by $\mathbf{Q}_k$s. Each image line l_k can be represented by a unit vector ε_k illustrated in Fig. 4. ε_k is just defined as the normal vector of the plane π which passes through the focal point F and the image line l_k. Obviously, ε_k is orthogonal to every point on the 3D feature line L_k.

To define the cost function J_l, we appropriately choose two *virtual* points, say $\mathbf{q}_0^{(a)}$ and $\mathbf{q}_0^{(b)}$, on an image line l_0. For the purpose of proper weighting, the norms of $\mathbf{q}_0^{(a)}$ and $\mathbf{q}_0^{(b)}$ are also re-scaled to 1 just as in Section 4.2. The points on the line L_0 which are corresponding to the two virtual points on l_0 are denoted by $\mathbf{Q}_0^{(a)}$ and $\mathbf{Q}_0^{(b)}$. Their corresponding points at other time instants ($t = -1, 0, +1$) are then denoted by $\mathbf{Q}_k^{(a)}$ and $\mathbf{Q}_k^{(b)}$, where $k = -, 0, +$.

There is no need for the two virtual points being physically extracted on the image line because a line can be re-constructed by any two points on it. However, it will be better to choose the two detected edge points of a tracked line segment as the virtual points. It is because the detected edge points on an image line

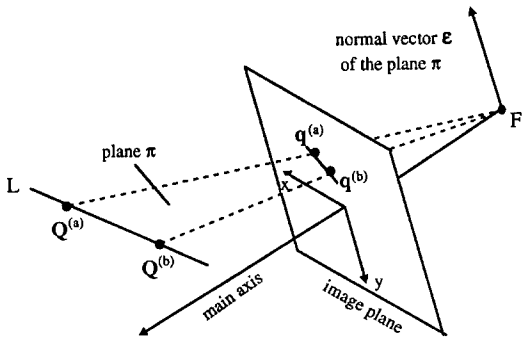

Fig. 4. Perspective projection for a line feature.

are usually projected by real 3D points on the object, and these 3D points are of approximately the same visual distance to the camera.

Take the first virtual point $\mathbf{q}_0^{(a)}$ for example, we may define a function $J_l^{(a)}$ like this

$$J_l^{(a)} = w_l\{\|\mathbf{Q}_-^{(a)}\cdot\varepsilon_-\|^2 + \|\mathbf{Q}_0^{(a)}\times\mathbf{q}_0^{(a)}\|^2 + \|\mathbf{Q}_+^{(a)}\cdot\varepsilon_+\|^2\}, \tag{19}$$

where w_l is a weighting factor for the line's cost function. The operator "$\cdot$" denotes the inner product of vectors.

Following the similar derivations described in Section 4.2, we may set $\partial J_l^{(a)}/\partial\mathbf{Q}_0^{(a)}$ to zero to minimize the cost function $J_l^{(a)}$. An optimum $\mathbf{Q}_0^{(a)}$ can be determined in terms of a given set of motion parameters.

$$\mathbf{Q}_0^{(a)} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}, \tag{20}$$

where

$$\mathbf{A} \equiv \sqrt{w_l}\begin{bmatrix}\varepsilon_-^T\mathbf{R}_-^* \\ \mathbf{E}_0 \\ \varepsilon_+^T\mathbf{R}_+^*\end{bmatrix}; \quad \mathbf{b} \equiv -\sqrt{w_l}\begin{bmatrix}\varepsilon_-^T\mathbf{M}\mathbf{a}_- \\ \mathbf{0} \\ \varepsilon_+^T\mathbf{M}\mathbf{a}_+\end{bmatrix}, \tag{21}$$

and the matrix $\mathbf{E}_0$ is the same as that defined in equation (14) except that its $\mathbf{q}_0$ is replaced by $\mathbf{q}_0^{(a)}$.

Substituting the optimum $\mathbf{Q}_0^{(a)}$ into equation (19), we have

$$J_l^{(a)} = \mathbf{b}^T(\mathbf{I} - \mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T)\mathbf{b} \equiv \mathbf{b}^T D^{(a)}\mathbf{b}. \tag{22}$$

Similar to equation (16), the vector $\mathbf{b}$ defined here can be rearranged as

$$\mathbf{b} = -\sqrt{w_l}\begin{bmatrix}\varepsilon_-^T\mathbf{M} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \varepsilon_+^T\mathbf{M}\end{bmatrix}\begin{bmatrix}\mathbf{a}_- \\ \mathbf{a}_+\end{bmatrix} \equiv \mathbf{H}\mathbf{a}. \tag{23}$$

Here $\mathbf{H}$ is a 5×4 matrix, and it depends only on the input image lines; $\mathbf{a}$ is a 4×1 vector.

Substituting equation (23) into equation (22), we have

$$J_l^{(a)} = \mathbf{a}^T(\mathbf{H}^T\mathbf{D}^{(a)}\mathbf{H})\mathbf{a} \equiv \mathbf{a}^T\mathbf{C}_l^{(a)}\mathbf{a}. \tag{24}$$

The 4×4 symmetric matrix $\mathbf{C}_l^{(a)}$ is also a function of only two unknowns, ω_- and ω_+; (2) the cost function $J_l^{(a)}$ is also a quadratic form of $\mathbf{a}$, just like J_p.

As to the other point $\mathbf{Q}_0^{(b)}$, we have the same derivation as that from equation (19) to equation (24). So we will have another 4×4 matrix $C_l^{(b)}$. Notice that the virtual point $\mathbf{q}_0^{(a)}$ (or $\mathbf{q}_0^{(b)}$) only appear in $D^{(a)}$ and $D^{(b)}$.

Each set of the three corresponding image lines can provide two cost functions $J_l^{(a)}$ and $J_l^{(b)}$. Adding these cost functions together, we have the final form

$$\sum_{j=1}^{N_l} J_{l,j} = \mathbf{a}^T\left[\sum_{j=1}^{N_l}\mathbf{C}_{l,j}^{(a)} + \mathbf{C}_{l,j}^{(b)}\right]\mathbf{a} \equiv \mathbf{a}^T\mathbf{B}_l\mathbf{a}. \tag{25}$$

4.4 *Fusion of image point and line features*

Substituting equations (18) and (25) into equation (9), we have

$$J = \mathbf{a}^T(\mathbf{B}_p + \mathbf{B}_l)\mathbf{a} \equiv \|\mathbf{a}\|^2(\bar{\mathbf{a}}^T\mathbf{B}\bar{\mathbf{a}}). \tag{26}$$

Here the 4×4 matrix $\mathbf{B}$ is a function of two unknowns ω_- and ω_+; $\bar{\mathbf{a}}$ is the unit vector of $\mathbf{a}$. Equation (26) combines both the contributions from image points and lines. It is the reason why we call "fusion" in the title of this paper.

Because a monocular camera system cannot recover the absolute value of translation (subject to a unknown factor), the magnitude of $\mathbf{a}$ [refer to equations (7) and (8)] is also undetermined. So we may temporarily assume that $\|\mathbf{a}\| = 1$. Then the cost function J can be further minimized if $\bar{\mathbf{a}}$ is the eigenvector corresponding to the smallest eigenvalue $\lambda_{\min}$ of $\mathbf{B}$. Therefore, minimizing J is equivalent to finding the best parameters (ω_-, ω_+) which can minimize the smallest eigenvalue of $\mathbf{B}(\omega_-, \omega_+)$. A lot of searching method can be applied to solve this minimization problem. For example, the Nelder–Meade simplex method used in a procedure named "FMINS" in the computer software "MATLAB" is adopted in this paper. Experiments show that the cost function J is well-conditioned and the minimization can always converge to the true solution.

When the rotation parameters ω_- and ω_+ are estimated, the unit vector $\bar{\mathbf{a}}$ is then determined. Because the true vector $\mathbf{a}$ is in fact a unknown scale-multiple h of $\bar{\mathbf{a}}$, the translation $\mathbf{T}_-$ and $\mathbf{T}_+$ can be written in term of the unknown variable h like this [according to the definitions in equations (7) and (8)]

$$\begin{bmatrix}T_{-,X} \\ T_{-,Y}\end{bmatrix} = \begin{bmatrix}-P_{cX}(-1+\cos\omega_-) + P_{cY}\sin\omega_- \\ -P_{cX}(\sin\omega_-) \quad -P_{cY}(-1+\cos\omega_-)\end{bmatrix} + h\begin{bmatrix}\bar{a}_1 \\ \bar{a}_2\end{bmatrix}, \tag{27}$$

$$\begin{bmatrix}T_{+,X} \\ T_{+,Y}\end{bmatrix} = \begin{bmatrix}-P_{cX}(-1+\cos\omega_+) + P_{cY}\sin\omega_+ \\ -P_{cX}(\sin\omega_+) \quad -P_{cY}(-1+\cos\omega_+)\end{bmatrix} + h\begin{bmatrix}\bar{a}_3 \\ \bar{a}_4\end{bmatrix}. \tag{28}$$

Finally, from equations (12) and (20), the 3D locations of the feature points $\mathbf{Q}_0$s and lines L_0s can be solved to the unknown factor h. If there is no additional constraint available, the unknown factor h seems to be an inherent determinacy for such a monocular camera system.

However, a traffic surveillance system which can not determine the true value of translation ($\mathbf{T}_-$ and $\mathbf{T}_+$) is usually worthless for real application. Therefore, in Section 4.5, a new constraint is proposed for estimating the final unknown factor h.

4.5. *The positional constraint for a moving vehicle*

To a traffic surveillance system, its interested objects are usually vehicles moving on the road. Generally speaking, we can observe several common constraints (see Fig. 1): (1) The width of the road constrains the varying range of $X-$component of an object point (in GPCS), say $[X_{\min}, X_{\max}]$. (2) The height of the moving vehicles on the road is usually limited, say $[0, Z_{\max}]$. These new constraints help us to determine the final unknown h.

From the determined motion parameters ω_-, ω_+, and $\bar{\mathbf{a}}$, we may substitute them into equations (4) and (6) and to determine $\mathbf{R}_i^*$, where $i = -, +$. If the object point $\mathbf{Q}_0$ is the feature point described in Section 4.2, $\mathbf{R}_i^*$ and $\bar{\mathbf{a}}$ can be substituted into equatons (12)–(14) to determine $\mathbf{Q}_0$ to the scale factor h. If the object point $\mathbf{Q}_0$ is the virtual point defined in Section 4.3, $\mathbf{R}_i^*$ and $\bar{\mathbf{a}}$ can be similarly substituted into equations (20) and (21) to determine $\mathbf{Q}_0$ to the scale factor h. Both of them have the following form: $\mathbf{Q}_0 = h\bar{\mathbf{Q}}_0$. For improving the reading, the explicit form of $\bar{\mathbf{Q}}_0$ is neglected here.

Because the positional constraints for a moving vehicle is expressed by GPCS, we have to transform $\mathbf{Q}_0$ to $\mathbf{P}_0$ by using the first equation of equation (1). Therefore,

$$\mathbf{P}_0 = h\mathbf{R}_c\bar{\mathbf{Q}}_0 + \mathbf{P}_c. \tag{29}$$

Every chosen points $\mathbf{P}_0$ (feature points or virtual points) must satisfy the two constraints (inequalities) given in the first paragraph of this subsection. However, it is still impossible to solve h from these inequalities. So we have to give another constraint Z_{ave} such that

$$\frac{1}{N_p}\sum_{i=1}^{N_p} P_{0,Z} = Z_{\text{ave}} = \text{constant}. \tag{30}$$

From equations (29) and (30), we approximately estimate the final unknown factor h.

4.6. *Initial solution for the optimization process*

In such a constrained SFM problem formulated in Section 2, it is possible that a closed-form solution may exists. Similar to the linear algorithm proposed by Weng, Huang and Ahuja,[5] we can derive an initial solution for our iterative algorithm. First, let us consider the motion defined in equation (5)

$$\mathbf{Q}' = \mathbf{R}^*\mathbf{Q} + \mathbf{T}^*. \tag{31}$$

Then, we have

$$\mathbf{Q}'\cdot[\mathbf{T}^*\times(\mathbf{R}^*\mathbf{Q})] = 0 \quad \text{or} \quad \mathbf{q}'\cdot[\mathbf{T}^*\times(\mathbf{R}^*\mathbf{q})] = 0, \tag{32}$$

where $\mathbf{q}$ and $\mathbf{q}'$ are separately the image points of $\mathbf{Q}$ and $\mathbf{Q}'$.

Substituting equation (6) into equation (32), we obtain the following homogeneous equation

$$\begin{aligned}&\alpha\cos\omega[\mathbf{q}'\cdot(\mathbf{m}_x\times\mathbf{q}) + (\mathbf{m}_z\cdot\mathbf{q})\quad(\mathbf{m}_y\cdot\mathbf{q}')] + \quad\beta\cos\omega\\ &[\mathbf{q}'\cdot(\mathbf{m}_y\times\mathbf{q})\quad -(\mathbf{m}_z\cdot\mathbf{q})\quad(\mathbf{m}_x\cdot\mathbf{q}')] + \alpha[-(\mathbf{m}_z\cdot\mathbf{q})\\ &(\mathbf{m}_y\cdot\mathbf{q}')] + \quad\beta[(\mathbf{m}_z\cdot\mathbf{q})\quad(\mathbf{m}_x\cdot\mathbf{q}')] + \alpha\sin\omega[(\mathbf{m}_x\cdot\mathbf{q})\\ &(\mathbf{m}_z\cdot\mathbf{q}')] + \beta\sin\omega\ [(\mathbf{m}_y\cdot\mathbf{q})\ (\mathbf{m}_z\cdot\mathbf{q}')] = 0.\end{aligned} \tag{33}$$

Equation (33) can be considered as a linear equation of six unknowns by defining six new variables like this

$$\begin{aligned}\mathbf{e} \equiv [e_1, e_2, e_3, e_4, e_5, e_6]^{\mathrm{T}} \equiv [&\alpha\cos\omega, \beta\cos\omega, \alpha, \beta,\\ &\alpha\sin\omega, \beta\sin\omega]^{\mathrm{T}}.\end{aligned} \tag{34}$$

If there are at least five given sets of point correspondences, the vector $\mathbf{e}$ can be easily solved to a scale factor (say c) by a least-squares approach. This unknown scale factor c can be determined by using the relationship among the components of $\mathbf{e}$. For example, we have $e_1^2 + e_5^2 = e_3^2$, $e_2^2 + e_6^2 = e_4^2$, and $(e_1/e_2) = (e_3/e_4) = (e_5/e_6)$. Finally, we will obtain a closed-form solution. It can be adopted as an initial solution.

In fact, according to the behavior of the cost function J in our experiences, it is usually good enough to adopt $\omega_- = \omega_+ = \mathbf{0}$ as the initial solution. It is because the rotation of a moving vehicle is seldom large.

5. DISCUSSION

In this section, we will discuss three related topics. They are described in Sections 5.1–5.3.

5.1. *The balance of weighting*

To a minimization process, the cost function J should be well-shaped or well-conditioned. If not, the iterative searching may be trapped by a local minima or causes a very error-sensitive motion estimation. These problems are often caused by an improper definition of the cost function. Now we will check whether the cost function J is well defined or not.

Considering the definitions of $J_{p,i}$ and $J_{l,j}$ in equations (10) and (19), it is easy to find that their basic elements are defined as an operation (inner or outer vector product) between an object point and an unit vector. So the cost values of the basic elements are proportional to the ray distance of the interested 3D object points. As stated in Section 4.5, there is no significant difference among the ray distances of

object points. It makes sure that each image feature approximately has the same level of contribution to the cost function. Therefore, the cost function is still quite well-conditioned even if all of the weighting factors are set to one.

5.2. *To handle missing data*

Our method still works even if some of the input data are missing. Basically, we set the quantities related to the missing image features to zero, and they will not contribute to the cost function J any more. Now the question is: "*How much missing data can our method handle?*" To answer this question, let us consider equations (13)–(15) again (for points). The cost function defined in equation (15) is the residual error of the system $\mathbf{AQ}_0 = \mathbf{b}$ when we substitute the optimum solution of $\mathbf{Q}_0$ [defined in equation (12)] into the linear system. Notice that only an over-determined linear system has a residual error. In other

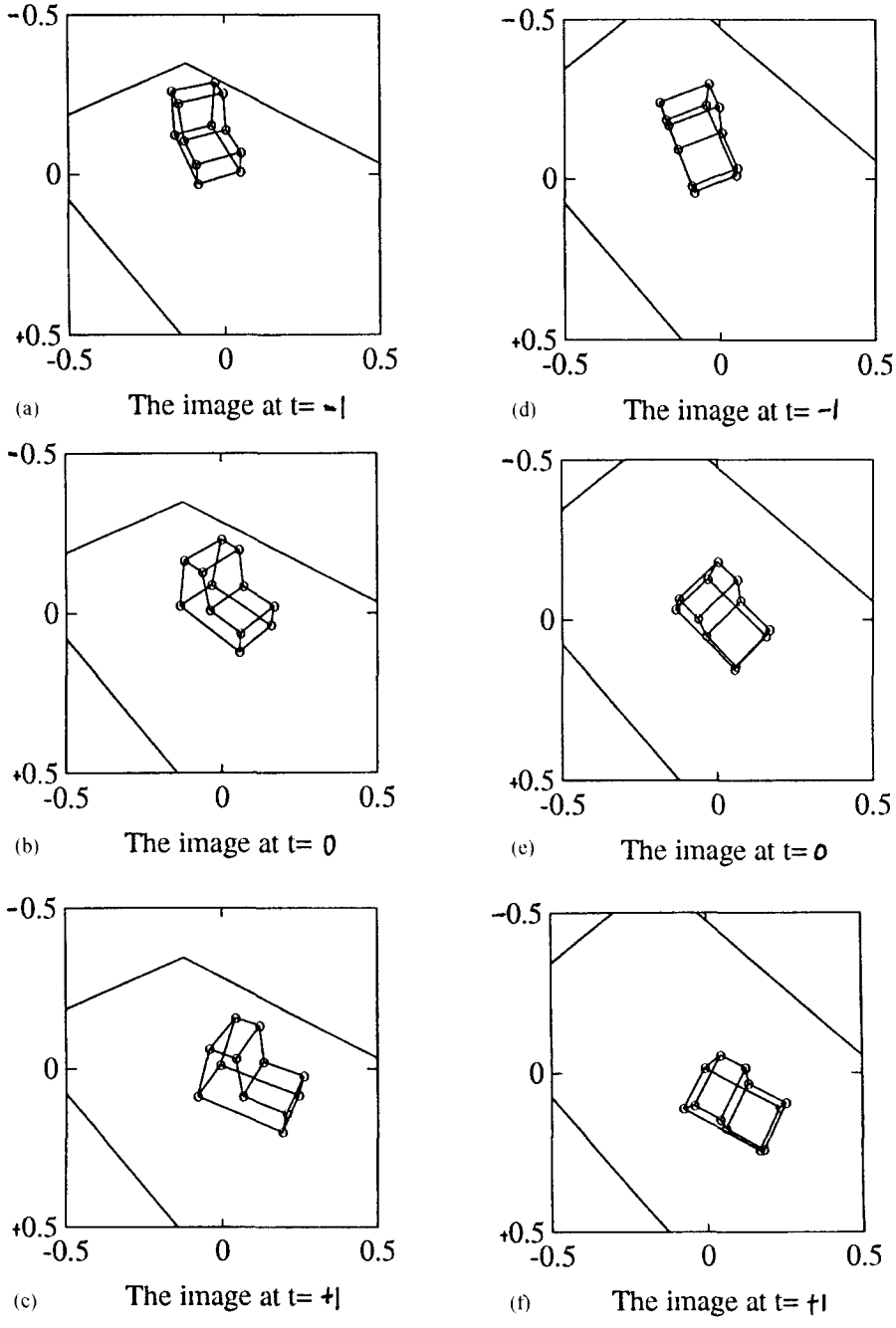

Fig. 5. Simulated image sequence—a moving vehicle. (a)–(c) seen by a camera whose focal point F is located at $(r, \theta, \phi) = (15, 45^\circ, 45^\circ)$. (a) $t = -1$; (b) $t = 0$; (c) $t = +1$. Figures (d)–(f) are similarly defined as that in Figs (a)–(c). But the focal point of the camera is now located at $(r, \theta, \phi) = (15, 15^\circ, 45^\circ)$.

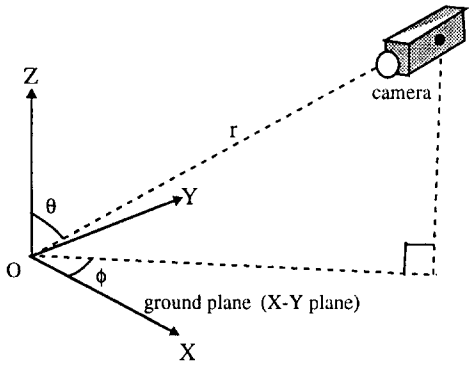

Fig. 6. The pose of the camera. The position of its focal point F (denoted by a vector $\mathbf{P}_c$) is specified by three parametrs: $\mathbf{r}$, θ, and ϕ.

words, the number of the rows of $\mathbf{A}$ should be larger than three and $\dim(\mathbf{A}^T\mathbf{A})$ must be equal to three. If there are too many missing image data such that the above condition is not satisfied, the cost function defined in equation (15) is always zero and makes no contribution to the total cost J any more. Similar discussion is suitable for the line features. So we have the following conclusion:

"*An image point should appear at least twice during the time interval included in the defined cost function J. Besides, an image line feature should appear at least three times in the same time interval*".

5.3. Simplification when there is no rotation

In daily experiences, the rotation of a moving vehicle on the road is usually small. So we may directly set ω_- and ω_+ to zero and substitute them into equation (26) to determine the matrix $\mathbf{B}$. The vector $\mathbf{a}$ is just the eigenvector corresponding to the smallest eigenvalue of $\mathbf{B}$. From equation (26) to equation (30), the motion estimation problem is completely solved without using any iterative search.

6. EXPERIMENTAL RESULTS

There are three goals for our experiments: (1) To prove the correctness of our algorithm. (2) To test the robustness of our algorithm under different conditions. (3) Our algorithm can work well when considering a real-world image sequence. For a better control of the experiments, a simulated image sequence is adopted for the first two goals.

6.1. Experiments for simulated image

Figures 5(a)–(c) shows a moving vehicle on the road. Three frames captured at $t = -1$, $t = 0$, and $t = +1$ are considered. Figures 5(d)–(f) shows the same moving vehicle but seen by the camera at different pose. To define the pose of the camera properly, let us see the configuration shown in Fig. 6. The focal

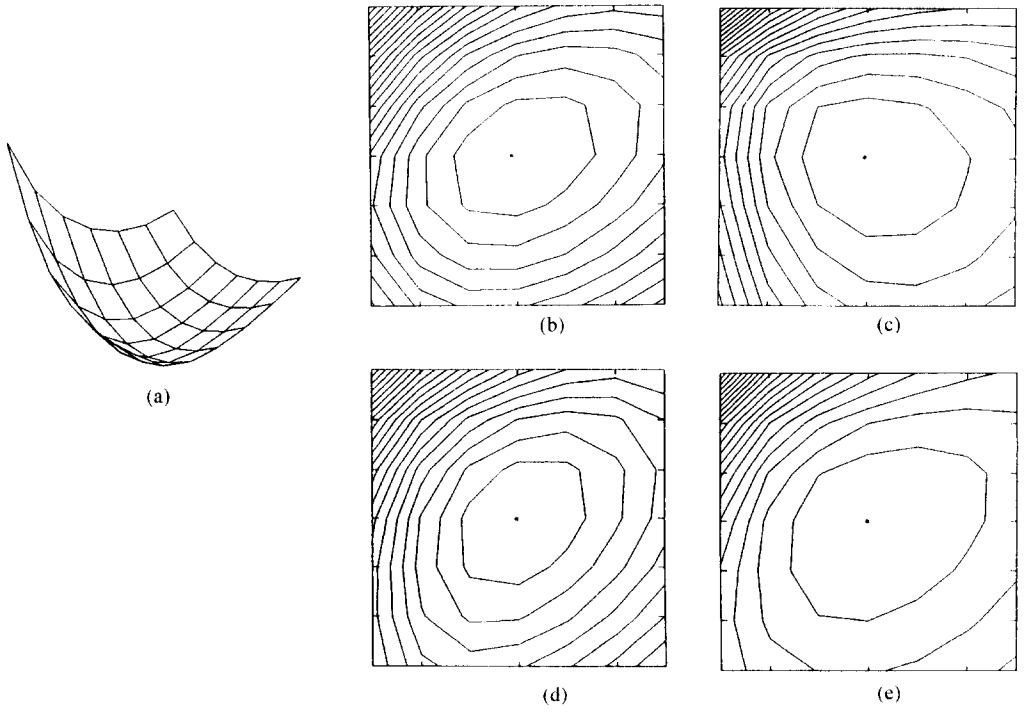

Fig. 7. Changing shapes of the cost function. Four cost functions are shown (a) The cost function near the true solution (has shifted to the center). $(r, \theta, \phi) = (15, 15°, 45°)$. (b) Level contour: $(r, \theta, \phi) = (15, 15°, 45°)$. (c) $(r, \theta, \phi) = (15, 45°, 45°)$. (d) $(r, \theta, \phi) = (10, 45°, 0°)$. (e) $(r, \theta, \phi) = (40, 45°, 0°)$.

center F (position vector $\mathbf{P}_c$) of the camera can be represented by

$$\mathbf{P}_c = [\, r\sin\theta\cos\phi,\ r\sin\theta\sin\phi,\ r\cos\theta]. \quad (35)$$

The three orthonormal vectors, $\mathbf{u}_i$ for $i = 1$ to 3, defining the pose of the camera are now set to

$$\mathbf{u}_3 = \frac{-\mathbf{P}_c}{\|\mathbf{P}_c\|};\quad \mathbf{u}_1 = \frac{\mathbf{u}_3 \times \hat{\mathbf{x}}}{\|\mathbf{u}_3 \times \hat{\mathbf{x}}\|};\quad \mathbf{u}_2 = \mathbf{u}_3 \times \mathbf{u}_1. \quad (36)$$

Figure 5(a)–(c) considers the case when $(r, \theta, \phi) = (15, 45^\circ, 45^\circ)$. Figure 5(d)–(f) considers the case when $(r, \theta, \phi) = (15, 15^\circ, 45^\circ)$. The motion parameters of the moving vehicle shown in Fig. 5 are, $\omega_- = -0.4$, $\omega_+ = 0.3$, $T_{-,X} = -0.5$, $T_{-,Y} = -2.0$, $T_{+,X} = 0.4$, and $T_{+,Y} = 1.8$. On the moving vehicle, its vertices and the lines linking these vertices are used as the input image features ($N_p = 12$ and $N_l = 18$). To compare the true solution and the estimated soluton, we define two vectors

$$\mathbf{c} \equiv [\, \omega_-, \omega_+\,]^T;\quad \mathbf{d} \equiv [\, T_{-,X}, T_{-,Y},\ T_{+,X}, T_{+,Y}]^T. \quad (37)$$

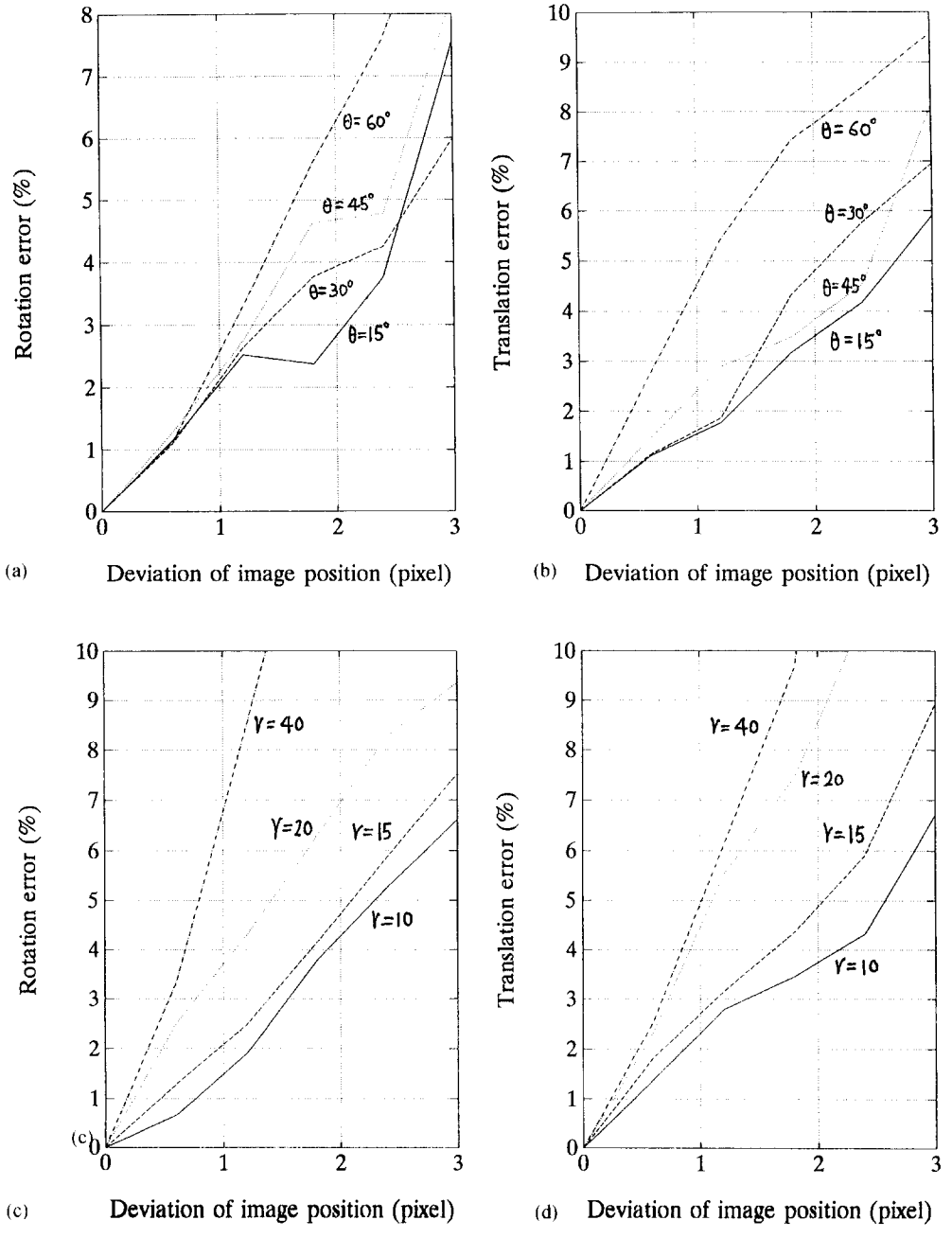

Fig. 8. Error performance of the algorithm under different poses of the camera. (a) and (b) The rotation and translation errors vs. noise when varying the angle θ. (c) and (d) The rotation and translation errors vs. noise when varying the distance r.

Therefore, the percentage errors of the estimated rotation and translation are separately defined as

$$\text{rotation error} = \frac{\|\hat{\mathbf{c}} - \mathbf{c}\|}{\|\mathbf{c}\|} \times 100\%;$$
$$\text{translation error} = \frac{\|\hat{\mathbf{d}} - \mathbf{d}\|}{\|\mathbf{d}\|}. \quad (38)$$

The final estimation errors for Figs 5(a)–(c) and Figs 5(d)–(f) are separately: (1) Rotation errors: $2.14 \times 10^{-8}\%$ and $3.45 \times 10^{-9}\%$. (2) Translation errors: $5.82 \times 10^{-7}\%$ and $1.05 \times 10^{-8}\%$. Here we assume that the true Z_{ave} has been given. It proves the correctness of the algorithm.

Figure 7(a) shows an example of the 2D cost function J. It is obtained by considering the same moving vehicle shown in Fig. 5. The domain of this cost function is $(\omega_-, \omega_+) = [-0.4 \pm 0.2, 0.3 \pm 0.2]$. From different poses of the camera, we can observe the shape change of the cost function. Figures 7(b)–(e) are four cost functions (level contours) obtained by different positions of the camera. Figures 7(b) and (c) consider the shape change induced by different θ (15° and 45°). Figures 7(d) and (e) consider the shape change induced by different distances r (10 and 40). We find that the cost functions corresponding to larger θ and larger distance r will be more ill-conditioned, which may influence the robustness of our algorithm.

Figure 8(a)–(d) shows the error performance under different poses of the camera. The noise-perturbed image-point positions are simulated by adding a 2D random vector (δ_1, δ_2) to each vertices of the moving vehicle shown in Fig. 5. Here δ_1 and δ_2 are random variables of normal distribution $N(0, \delta)$. δ is the standard deviation. If we assume that the total image plane (1×1) is uniformly divided into 512×512 grids, every pixel on it occupies an area of $(1/512) \times (1/512)$ square unit. Therefore, the deviation $\delta = \sigma \times (1/512)$ is considered as the "noise level of σ pixels". Every point

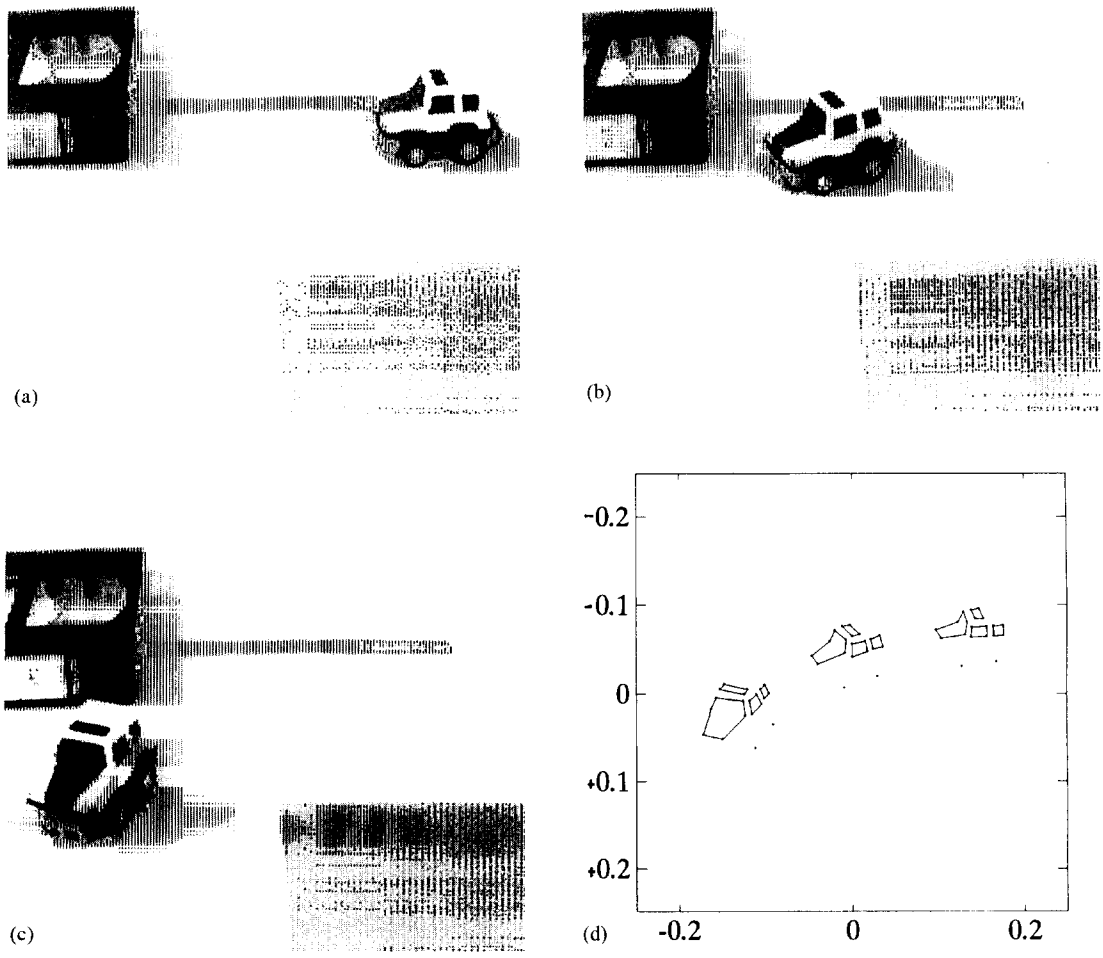

Fig. 9. Real-world image test; (a) the frame at $t = -1$; (b) the frame at $t = 0$; (c) the frame at $t = +1$; (d) the extracted image points and line features ($N_p = 20$, $N_l = 18$) at three time instants.

on the curves shown in Fig. 8 is the average of 40 tests. Figures (a)–(b) consider the cases when $(r, \theta, \phi) = (15, \theta, 45^\circ)$. Figures 8(c)–(d) consider the cases when $(r, \theta, \phi) = (r, 45^\circ, 0^\circ)$. It is easy to find that the cases corresponding to larger θ and larger distance r are more error-sensitive. The influence from the changing distance r seems stronger than that from the changing tilt angle θ.

6.2. *Experiments for real-world image*

Figures 9(a)–(c) show the three frames of the testing real-world image sequence. The toy car moves from upper right to lower left. The size of the toy car is about 4.7 cm × 3.2 cm × 3.5 cm. To extract its image features (points or lines), the original image is first transformed into an "edge" image by edge detection. Both the point and line features are manually picked and traced during the three frames. To obtain a line feature, we collect all of its corresponding edge points and then determine the best-fitting line by linear regression. Finally, we can trace continuously 20 points and 18 lines at three time instants. It is shown in Fig. 9(d).

Following the definitions of coordinate transform described in Section 2, the pose of the camera is specified by $\{\mathbf{R}_c, \mathbf{P}_c\}$. After applying a simple camera calibration process (not shown here), we have the following parameters: aspect ratio = 1.036, focal length = 19.051 cm, viewing angle = 28°, and

$$\mathbf{R}_c = \begin{bmatrix} 0.0336 & 0.5066 & -0.8615 \\ 0.9989 & 0.0099 & 0.0448 \\ 0.0312 & -0.8621 & -0.5057 \end{bmatrix};$$

$$\mathbf{P}_c = 19.051 \times \begin{bmatrix} 2.2386 \\ -0.0559 \\ 1.3303 \end{bmatrix} \text{cm}.$$

The true motion parameters of the moving toy car are directly measured as follows: $[\omega_-, \omega_+] = (-0.3840, 0.8029)$ (rad), $[T_{-,X}, T_{-,Y}] = (-1.74, 7.87)$ (cm), $[YT_{+,X}, T_{+,Y}] = (6.23, -6.13)$ (cm). The average height Z_{ave} defined in equation (30) is measured as 2.56 (cm).

Finally, the estimated results are: $[\omega_-, \omega_+] = (-0.3519, 0.8306)$ (rad), error = 4.7640%; $[T_{-,X}, T_{-,Y}] = (-1.65, 7.25)$ (cm), error = 7.7729%; $[T_{+,X}, T_{+,Y}] = (5.69, -5.58)$ (cm), error = 8.8189%. The estimation errors may be mainly due to quantization errors, calibration errors, and position errors. But they still seem acceptable. The result could be better if a more precise camera calibration can be obtained.

7. CONCLUSIONS

In this paper, a new iterative method is proposed for solving the problem of vehicle-type motion estimation. This method has several advantages:

- It can handle both the point and line features as its input image data. The contributions from these image features can be fused together. All we have to do is to minimize the smallest eigenvalue of a 4×4 symmetry and nonnegative definitive matrix **B**. The dimension of searching space is very small (= 2).
- The definition of the cost function is very suitable for parallel processing.
- Each component cost function in J has approximately the same numerical contribution to the final cost value. It is just fine to set all of the weighting factors w's defined in J to 1. Therefore, there is no serious weighting problem in our method.
- Its cost function J is so well-conditioned that the final 3D motion estimation is robust and insensitive to noise, which is proved by experiments.
- It can handle the case of missing point/line data to certain degree. Besides, a line feature has an inherent ability to handle partial occlusion.

Both the simulated and real-world images are tested in this paper. Simulated experiments are mainly used to test (1) the correctness of the proposed algorithm, (2) the error performance of the proposed algorithm. The results obtained from real-world image show that our method can work well in real application.

REFERENCES

1. J. W. Roach and J. K. Aggarwal, Determine the movement of object from a sequence of image, *IEEE Trans. Pattern Anal. Machine Intell.* **PAMI-2(6),** 554–562 (1980).
2. H. C. Longuet-Higgins, A computer program for reconstructing a scene from two projections, *Nature* **293,** 133–135 (1981).
3. R. Y. Tsai and Thomas S. Huang, Uniqueness and estimation of 3D motion parameters of rigid objects with curved surfaces, *IEEE Trans. Pattern Anal. Machine Intell.* **PAMI-6(1),** pp. 13–27, 1984.
4. Xinhua Zhuang, A simplification to linear two-view motion algorithms, *Comput. Vision Graphics Image Process.* **46,** 175–178 (1989).
5. Juyang Weng, Thomas S. Huang and Narendra Ahuja, Motion and structure from two perspective views: algorithms, error analysis, and error estimation, *IEEE Trans. Pattern Anal. Machine Intell.* **11(5),** 451–476 (1989).
6. Johan Philip, Estimation of 3D motion of rigid objects from noisy observation, *IEEE Trans. Pattern Anal. Machine Intell.* **13(1),** 61–66 (1991).
7. Minas E. Spetsakis and Yiannis Aloimonos, Optimal visual motion estimation: a note, *IEEE Trans. Pattern Anal. and Machine Intell.* **14(9),** 959–964 (1992).
8. H. H. Chen and T. S. Huang, Maximal matching of 3D points for multiple-object motion estimation, *Pattern Recognition* **21(2),** 75–90 (1988).
9. Vincent S. S. Hwang, Tracking feature points in time-varying images using an opportunistic selection approach, *Pattern Recognition* **22(3),** 247–256 (1989).
10. Paul M. Griffin and Sherri L. Messimer, Feature point tracking in time-varying images, *Pattern Recognition Lett.* **11,** 843–848, (1990).
11. K. Rangarajan and M. Shah, Establishing motion correspondences, *CVGIP: Image Understanding* **54(1),** 56–73 (1991).

12. B. L. Yen and T. S. Huang, "Determining 3D motion/structure of a rigid body using straight line correspondences, *Image Sequence Processing and Dynamic Scene Analysis*, Springer, New York/Berlin, (1983).
13. A. Mitichi, S. Seida and J. K. Aggarwal, Line-based computation of structure and motion using angular invariance, *Proc. Workshop on Motion: Representation and Analysis, IEEE Computer Society*, pp. 175–180, 7–9 May, Charleston, SC (1986).
14. Yuncai Liu and Thomas S. Huang, A linear algorithm for motion estimation using straight line correspondences, *Comput. Vision Graphics Image Process.* **44,** 35–57 (1988).
15. Yuncai Liu and Thomas S. Huang, Three-dimensional motion determination from real scene images using straight line correspondences, *Pattern Recognition* **25(6),** 617–639 (1992).
16. Minas E. Spetsakis and Yiannis Aloimonos, Structure from motion using line correspondences, *Int. J. Comput. Vision* **4,** 171–183 (1990).
17. Juyang Weng, Thomas S. Huang and Narendra Ahuja, Motion and structure from line correspondences: closed-form solution, uniqueness, and optimization, *IEEE Trans. on Pattern Anal. Machine Intell.* **14(3),** 318–335 (1992).
18. G. Medioni and R. Nevatia, Matching image using linear features, *IEEE Trans. Pattern Anal. Machine Intell.* **PAMI-6(6),** 675–685 (1984).
19. J. B. Burns, A. R. Hanson and E. M. Riseman, Extracting straight lines, *IEEE Trans. Pattern Anal. Machine Intell.*, **PAMI-8(4),** 425–445 (1986).
20. F. Bergholm, Edge focusing, *IEEE Trans. Pattern Anal. Machine Intell.* **PAMI-9(6),** 726–741 (1987).
21. Rachid Deriche and Oliver Faugeras, Tracking line segments, *Image Vision Comput.* **8(4),** 261–270 (1990).
22. T. Broida and R. Chellappa, Estimation of object motion parameters from noisy images, *IEEE Trans. Pattern Anal. Machine Intell.* **PAMI-8(1),** 90–99 (1986).
23. Juyang Weng, Thomas S. Huang, and Narendra Ahuja, 3D motion estimation, understanding, and prediction from noisy image sequences, *IEEE Trans. Pattern Anal. Machine Intell.* **PAMI-9(3),** 370–389 (1987).
24. Gwo Jyh Tseng and Arun K. Sood, Analysis of long image sequence for structure and motion estimation, *IEEE Trans. System Man and Cybernet.* **19(6),** 1511–1526 (1989).
25. Hormoz Shariat and Keith E. Price, Motion estimation with more than two frames, *IEEE Trans. Pattern Anal. and Machine Intell.* **12(5),** 417–434 (1990).
26. Ted J. Broida and Rama Chellappa, Estimating the kinematics and structure of a rigid object from a sequence of monocular images, *IEEE Trans. Pattern Anal. Machine Intell.* **13(6),** 497–512 (1991).
27. Xiaoping Hu and Narendra Ahuja, Motion and structure estimation using long sequence motion models, *Image Vision Comput.* **11(9),** 549–569 (1993).
28. Y. F. Wang, Nitin Karandikar and J. K. Aggarwal, Analysis of video image sequences using point and line correspondences, *Pattern Recognition* **24(11),** 1065–1084 (1991).
29. Yuncai Liu, Thomas S. Huang, and O. D. Faugeras, Determination of camera location from 2D to 3D line and point correspondences, *IEEE Trans. Pattern Anal. Machine Intell.* **12(1),** 28–37 (1990).
30. Mun K. Leung and T. S. Huang, Estimating 3D vehicle motion in an outdoor scene using stereo image sequences, *Int. J. Imaging Systems Technol.* **4,** 80–97 (1992).
31. Yuncai Liu and Thomas S. Huang, Vehicle-type motion estimation from multi-frame images, *IEEE Trans. on Pattern Anal. Machine Intell.* **15(8),** 802–808 (1993).
32. M. Spetsakis and Y. Aloimonos, A unified thoery of structure from motion, *Proc. DARPA Image Understanding Workshop*, pp. 271–283. Pittsburgh, PA (1990).
33. M. Spetsakis, A linear algorithm for point and line based structure from motion, *CVGIP: Image Understanding* **56,** 230–241 (1992).

About the Author—LIN-GWO LIOU was born in Taiwan. He received his B.S. degree from the National Chiao Tung University (N.C.T.U.) in Taiwan in 1989 and Ph.D. degree from the National Taiwan University in 1995, all in Electrical Engineering. Right now he is in military service His research interests include motion image analysis, methods for 3D object reconstruction, pattern recognition in image application.

About the Author—SOO-CHANG PEI was born in Soo-Auo, Taiwan in 1949. He received his B.S.E.E. from National Taiwan University in 1970 and M.S.E.E. and Ph.D. from the University of California, Santa Barbara in 1972 and 1975, respectively. He was an engineering officer in the Chinese Navy Shipyard from 1970 to 1971. From 1971 to 1975, he was a research assistant at the University of California, Santa Barbara. He was Professor and Chairman in the EE Department of Tatung Institute of Technology from 1981 to 1983. Presently, he is the Professor and Chairman of EE Department at National Taiwan University. His research interests include digital signal processing, image processing, optical information processing, and laser holography. Dr Pei is a member of the IEEE, Eta Keppa Nu and the Optical Society of America.