Three-Dimensional Trajectory Estimation from Image Position and Velocity

STEVEN D. BLOSTEIN, Senior Member, IEEE
LIN ZHAO
ROBERT M. CHANN, Member, IEEE
Queen's University
Canada

A recursive algorithm for estimating the three-dimensional trajectory and structure of a moving rigid object in an image sequence has been previously developed by Broida, Chandrashekhar, and Chellappa [1]. Since then, steady advances have occurred in the calculation of optical flow. This work improves 3-D motion trajectory and structure estimation by incorporating optical flow into the estimation framework introduced in [1]. The new solution combines optical flow and feature point measurements and determines their statistical relationship. The feasibility of a hybrid feature point/optical flow algorithm, demonstrated through detailed simulation on synthetic and real image sequences, significantly lowers bias and mean squared error in trajectory estimation over the feature-based approach [1].

Manuscript received August 12, 1998; revised November 1, 1999 and April 11, 2000; released for publication June 9, 2000.

IEEE Log No. T-AES/36/4/11357.

Refereeing of this contribution was handled by X. R. Li.

This work was supported by a grant from the Information Technology Research Centre of Ontario and by the Natural Sciences and Engineering Research Council of Canada Grant OGP0041731.

Authors' current addresses: S. D. Blostein and L. Zhao, Department of Electrical and Computer Engineering, Queen's University, Kingston, Ontario K7L 3N6, Canada; R. M. Chann, ISG Technologies Inc., Mississauga, Ontario, L4V 1S7, Canada.

I. INTRODUCTION

Recovering the 3-D motion and structure of a rigid object from image sequences is a fundamental problem in applications such as target recognition, reconnaissance, vision-based servoing, and computer graphics. The main approaches to 3-D trajectory estimation have usually exploited either point features [2] or optical flow [3] independently.

In feature-based approaches, 3-D motion and structure are estimated by observing the 2-D positions of a set of relatively sparse set of image features (corners, lines, regions, etc.) over two or more time-sequential images. In contrast, optical-flow-based techniques estimate instantaneous 3-D motion from dense image plane velocity data.

Feature-based estimation can be subdivided into recursive [1, 4, 5] and batch [6, 7] formulations. This paper focuses on recursive estimation, which is better suited for on-line applications. In [1], a nonlinear dynamical system is formulated, and linearized using an iterated extended Kalman filter (IEKF). Wu, et al. [8] adopt a similar approach, except assume a priori knowledge about object structure, i.e., the configuration of the feature points within an object-centered system, resulting in a simplified measurement function compared with [1]. The formulation in [4] is also simpler than in [1], except it assumed that the center of rotation is always visible, which appears difficult in practice to achieve. Silvén and Repo [9] have recently developed an integrated monocular visual tracking system, with an emphasis on real-time operation. Finally, Chandrashekhar and Chellappa [10] use known navigational landmarks in their state estimation formulation, and interleave motion estimation and feature correspondence. Salient features of these recursive algorithms are compared in Table I.

In contrast, optical-flow-based motion and structure estimation requires computation of an optical flow field $\{\mathbf{u}(x,y) = [u(x,y)\ v(x,y)]^T\}$. The optical flow at coordinates (x,y) on the image plane, in terms of 3-D translational velocity $[T_x\ T_y\ T_z]^T$, the depth Z of 3-D point $[X\ Y\ Z]^T$, and rotational velocity $[\omega_x\ \omega_y\ \omega_z]^T$ can be expressed as [11]

$$u(x,y) = \frac{T_x}{Z} - x\frac{T_z}{Z} - xy\omega_x + (1+x^2)\omega_y - y\omega_z \quad (1)$$

$$v(x,y) = \frac{T_y}{Z} - y\frac{T_z}{Z} - (1+y^2)\omega_x + xy\omega_y + x\omega_z. \quad (2)$$

A variety of methods exist for computing optical flow [12]. Once optical flow is computed, 3-D motion and structure estimation can be estimated using methods such as proposed in [3], which involve splitting the optical flow equation algebraically into three separate components.

It is also possible to estimate 3-D motion without explicitly computing optical flow nor establishing feature correspondence prior to recovery of 3-D motion and structure [13]. However, such

TABLE I
Summary of Recursive 3-D Motion Estimation Techniques

Name	# Frames	# Features	Type	Test Data
Broida et al.	100	4	point	synthetic and real
Chandrashekhar & Chellappa	16	11	point	real
Iu & Wohn	100	3	point	synthetic
Silvén & Repo	1000	1–6	point	real
Wu et al.	6	14	point	real

direct methods appear to be at an early stage of development.

The rest of the paper is organized as follows. As in [1], we assume an environment where a stationary video camera is used to observe a single rigid moving object over time. Our goal is to recover the 3-D trajectory and structure of this object from an extended image sequence captured by the sensor array in perspective projection. Section II introduces the imaging, object, and motion models and briefly reviews the recursive formulation proposed in [1], and Section III describes the new approach that combines optical flow and position measurements. Experimental results on synthetic and real scenes are then described in Section IV.

II. SYSTEM MODEL

A. Imaging Model

The image acquisition process is modeled using perspective projection, in which a 3-D point $\mathbf{P} = (X,Y,Z)$ projects onto the 2-D image point $\mathbf{p} = (x,y)$ via the transformation

$$\mathbf{p} = \begin{bmatrix} x \\ y \end{bmatrix} = \frac{f}{Z}\begin{bmatrix} X \\ Y \end{bmatrix} + \begin{bmatrix} n_x \\ n_y \end{bmatrix} \tag{3}$$

where n_x and n_y represent additive noise terms that account for random errors mainly due to spatial quantization. Without loss of generality, we either assume a unity focal length (i.e., $f = 1$), or, where appropriate, use *focal lengths* as our unit of distance.

B. Object Model

To observe a moving object, whose structure is unknown, through a stationary camera, we define a camera-centered coordinate system (CCCS) with z-axis aligned with the optical axis. An object-centered coordinate system (OCCS) is also needed to specify the object structure to be estimated. Let O_o be the origin of the OCCS which is not directly observable. In the CCCS, let the object rotation center be located at

$$\mathbf{s}_\mathbf{R}(t) = [X_R(t) \quad Y_R(t) \quad Z_R(t)]^T \tag{4}$$

and the position vector of the ith feature point in the OCCS be located at

$$\mathbf{s}_{\mathrm{oi}} = [X_{\mathrm{oi}} \quad Y_{\mathrm{oi}} \quad Z_{\mathrm{oi}}]^T. \tag{5}$$

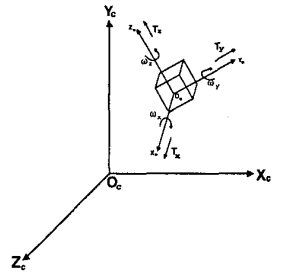


Fig. 1. Camera- and object-centered coordinate systems.

Using (4) and (5), the position vector of the ith feature point in the CCCS is

$$\mathbf{s}_i(t) = \mathbf{s}_\mathbf{R}(t) + R(t)\mathbf{s}_{\mathrm{oi}} \tag{6}$$

where $R(t)$ is the 3-D rotation matrix that aligns the OCCS with the CCCS. Since object rigidity is assumed, $\mathbf{s}_{\mathrm{oi}}$ is not dependent on time (see Fig. 1).

C. Motion Model

Motion of a rigid body is completely characterized by translational and rotational components. We assume constant translational velocity $\mathbf{T} = [T_x \; T_y \; T_z]^T$. A point $\mathbf{x}(t)$ on the translating object with initial location $\mathbf{x}(t_0)$ at time $t = t_0$ will have position

$$\mathbf{x}(t) = \mathbf{x}(t_0) + (t - t_0)\mathbf{T} \tag{7}$$

at any time t.

The unit quaternion, $\mathbf{q}(t)$ is used to represent the orientation of a rotating object. By assuming constant rotational velocity, $\omega = [\omega_x \; \omega_y \; \omega_z]^T$, $\mathbf{q}(t)$ can be propagated in time using the matrix exponential, e.g.,

$$\mathbf{q}(t) = \exp[\Omega(t - t_0)]\mathbf{q}(t_0) \tag{8}$$

where

$$\Omega = \frac{1}{2}\begin{bmatrix} 0 & \omega_z & -\omega_y & \omega_x \\ -\omega_z & 0 & \omega_x & \omega_y \\ \omega_y & -\omega_x & 0 & \omega_z \\ -\omega_x & -\omega_y & -\omega_z & 0 \end{bmatrix}. \tag{9}$$

In addition, the rotation matrix $R(t)$ in (6), can be expressed in terms of the unit quaternion [1].

D. State-Space Model

Our plant equation consists of a rigid object translating and rotating continuously in time, and

measurements are made at equally spaced discrete time intervals. The vector differential equation that describes the continuous plant is

$$\frac{d\mathbf{x}(t)}{dt} = \mathbf{a}(\mathbf{x}(t),t) + G(t)\mathbf{w}(t) \tag{10}$$

where $\mathbf{x}(t)$ is the state of the system, $\mathbf{w}(t)$ represents the process noise and $G(t)$ maps $\mathbf{w}(t)$ onto the state space. Observations, $\mathbf{y}_k$, are described by the discrete measurement equation

$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k) + \mathbf{v}_k \tag{11}$$

where $\mathbf{h}(\cdot)$ is a nonlinear measurement function, due to perspective transformation (1), and $\mathbf{v}_k$ models measurement noise arising mainly from spatial quantization. The problem is to find an estimate, $\hat{\mathbf{x}}_k$, of $\mathbf{x}_k$, based on $\mathbf{y}_k$. Since both $\mathbf{a}(\mathbf{x}(t),t)$ and $\mathbf{h}(\mathbf{x}_k)$ are nonlinear, various types of extended Kalman filters can be used [1, 14] and linearization of $\mathbf{a}(\mathbf{x}(t),t)$ and $\mathbf{h}(\mathbf{x}_k)$ are required, e.g., we form matrices

$$A(\hat{\mathbf{x}}(t),t) = \left.\frac{d\mathbf{a}(\mathbf{x}(t),t)}{d\mathbf{x}(t)}\right|_{\mathbf{x}(t)=\hat{\mathbf{x}}(t)} \quad \text{and}$$

$$H(\hat{\mathbf{x}}_k) = \left.\frac{d\mathbf{h}(\mathbf{x}_k)}{d\mathbf{x}_k}\right|_{\mathbf{x}_k=\hat{\mathbf{x}}_k} \tag{12}$$

where $\hat{\mathbf{x}}(t)$ and $\hat{\mathbf{x}}_k$ denote points about which $\mathbf{a}(\mathbf{x}(t),t)$ and $\mathbf{h}(\mathbf{x}_k)$ are linearized, respectively.

Following [1], we choose our state vector to be estimated as

$$\mathbf{x}(t) = \begin{bmatrix} X_R(t)/Z_R(t) \\ Y_R(t)/Z_R(t) \\ T_x/Z_R(t) \\ T_y/Z_R(t) \\ T_z/Z_R(t) \\ q_1 \\ q_2 \\ q_3 \\ q_4 \\ \omega_x \\ \omega_y \\ \omega_z \\ X_{o1}/Z_R(t) \\ Y_{o1}/Z_R(t) \\ Z_{o1}/Z_R(t) \\ \vdots \\ X_{oM}/Z_R(t) \\ Y_{oM}/Z_R(t) \\ Z_{oM}/Z_R(t) \end{bmatrix} = \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \\ s_7 \\ s_8 \\ s_9 \\ s_{10} \\ s_{11} \\ s_{12} \\ s_{13} \\ s_{14} \\ s_{15} \\ \vdots \\ s_{10+3M} \\ s_{11+3M} \\ s_{12+3M} \end{bmatrix}$$

$$= \begin{bmatrix} x\text{-position} \\ y\text{-position} \\ x\text{-translational velocity} \\ y\text{-translational velocity} \\ z\text{-translational velocity} \\ \text{3-D rotational position} \\ \text{3-D rotational position} \\ \text{3-D rotational position} \\ \text{3-D rotational position} \\ x\text{-axis rotational velocity} \\ y\text{-axis rotational velocity} \\ z\text{-axis rotational velocity} \\ \left.\begin{matrix} \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \end{matrix}\right\} \text{structure states} \end{bmatrix} \tag{13}$$

The normalization factor $Z_R(t)$ is needed due to the unknown scale factor inherent in the perspective transformation [1]. In practice, it is reasonable to assume that crude estimates of object distance, $Z_R(t)$, are available through other sensors. Using the perspective projection, the measurement vector corresponding to (11) is given by [1]

$$\mathbf{y}_{\mathrm{BCC}}(t_k) = \begin{bmatrix} x_1 \\ y_1 \\ \vdots \\ x_M \\ y_M \end{bmatrix} = \begin{bmatrix} X_1/Z_1 \\ Y_1/Z_1 \\ \vdots \\ X_M/Z_M \\ Y_M/Z_M \end{bmatrix} \tag{14}$$

where $\mathbf{p}_i = (x_i, y_i)$ is the projection of ith 3-D feature point $\mathbf{P}_i = (X_i, Y_i, Z_i)$ onto the image plane. This measurement vector was used by Broida, Chandrashekhar, and Chellappa [1] which we refer to as BCC. The above measurement function is nonlinear in the state variables. It can be shown [1, 14] that the image plane coordinates of the ith feature point, in terms of the state variables is given by

$$(x_i, y_i) = \left(\frac{s_1 + a_i}{1 + c_i}, \frac{s_2 + b_i}{1 + c_i}\right) \tag{15}$$

where

$$\begin{aligned} a_i = {} & (s_6^2 - s_7^2 - s_8^2 + s_9^2)s_{10+3i} \\ & + 2(s_6 s_7 - s_8 s_9)s_{11+3i} \\ & + 2(s_6 s_8 + s_7 s_9)s_{12+3i} \end{aligned} \tag{16}$$

$$b_i = 2(s_6s_7 + s_8s_9)s_{10+3i} + (-s_6^2 + s_7^2 - s_8^2 + s_9^2)s_{11+3i} + 2(s_7s_8 - s_6s_9)s_{12+3i} \quad (17)$$

$$c_i = 2(s_6s_8 - s_7s_9)s_{10+3i} + 2(s_7s_8 + s_6s_9)s_{11+3i} + (-s_6^2 - s_7^2 + s_8^2 + s_9^2)s_{12+3i}. \quad (18)$$

III. NEW APPROACH

A. General Optical Flow Formulation

Suppose that, in addition to the image plane coordinates of feature points, we also have available estimates of image plane velocity (optical flow) at each feature location. With this additional information, we improve the accuracy of state estimates as follows. Differentiating image coordinates (x_i, y_i) with respect to time, we obtain optical flow components

$$u_i = \frac{dx_i}{dt} = \frac{d}{dt}\left(\frac{X_i}{Z_i}\right) = \frac{\dot{X}_i}{Z_i} - \frac{X_i\dot{Z}_i}{Z_i^2} \quad (19)$$

$$v_i = \frac{dy_i}{dt} = \frac{d}{dt}\left(\frac{Y_i}{Z_i}\right) = \frac{\dot{Y}_i}{Z_i} - \frac{Y_i\dot{Z}_i}{Z_i^2} \quad (20)$$

where, without loss of generality, the focal length is assumed to be unity.

In order to incorporate optical flow into the recursive filtering framework proposed by Broida, et al., (19) and (20) must be expressed in terms of state variables. Rearranging (19) and (20),

$$u_i = \frac{\dot{X}_i}{Z_R}\frac{Z_R}{Z_i} - \frac{X_i}{Z_R}\frac{Z_R}{Z_i}\frac{\dot{Z}_i}{Z_R}\frac{Z_R}{Z_i} \quad (21)$$

$$v_i = \frac{\dot{Y}_i}{Z_R}\frac{Z_R}{Z_i} - \frac{Y_i}{Z_R}\frac{Z_R}{Z_i}\frac{\dot{Z}_i}{Z_R}\frac{Z_R}{Z_i}. \quad (22)$$

Clearly, we need to express the velocity of the ith feature point with respect to the camera, i.e., $\dot{\mathbf{s}}_i(t) = [\dot{X}_i\ \dot{Y}_i\ \dot{Z}_i]^T$ in terms of the system states. Differentiating (6) with respect to time, we obain

$$\frac{d}{dt}\mathbf{s}_i(t) = \frac{d}{dt}\mathbf{s}_\mathbf{R}(t) + \frac{dR(t)}{dt}\mathbf{s}_{\text{oi}} \quad (23)$$

where

$$\frac{d}{dt}R(t) = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} R(t). \quad (24)$$

Also, from (6),

$$\frac{Z_R}{Z_i} = \frac{Z_R}{Z_R + \mathbf{R}_3\mathbf{s}_{\text{oi}}} = \frac{1}{1 + \mathbf{R}_3\dfrac{\mathbf{s}_{\text{oi}}}{Z_R}}. \quad (25)$$

Using the above equations, (19) and (20) can be expressed in terms of the state variables defined in (13)

$$u_i = \frac{s_3 + c_is_{11} - b_is_{12}}{1 + c_i} - \frac{(s_5 + b_is_{10} - a_is_{11})(s_1 + a_i)}{(1 + c_i)^2} \quad (26)$$

$$v_i = \frac{s_4 - c_is_{10} + a_is_{12}}{1 + c_i} - \frac{(s_5 + b_is_{10} - a_is_{11})(s_2 + b_i)}{(1 + c_i)^2} \quad (27)$$

where a_i, b_i, and c_i are defined in (16), (17), and (18), respectively, and where i denotes one of M feature points. The new measurement vector $\mathbf{y}_k$ doubles in size over (14) as there are now two extra observations at each feature point location, which is given as

$$\mathbf{y}_(t_k) = \begin{bmatrix} x_1 \\ y_1 \\ \vdots \\ x_M \\ y_M \\ u_1 \\ v_1 \\ \vdots \\ u_M \\ v_M \end{bmatrix}. \quad (28)$$

The individual elements of the Jacobian matrix H_k are obtained by differentiating (15), (26), and (27) with respect to each state variable, i.e.,

$$H(\hat{\mathbf{x}}_k) = \left.\frac{d\mathbf{h}(\mathbf{x}_k)}{d\mathbf{x}_k}\right|_{\mathbf{x}_k=\hat{\mathbf{x}}_k} = \begin{bmatrix} \dfrac{\partial x_1}{\partial s_1} & \cdots & \dfrac{\partial x_1}{\partial s_{12+3M}} \\ \dfrac{\partial y_1}{\partial s_1} & \cdots & \dfrac{\partial y_1}{\partial s_{12+3M}} \\ \vdots & \cdots & \vdots \\ \dfrac{\partial u_M}{\partial s_1} & \cdots & \dfrac{\partial u_M}{\partial s_{12+3M}} \\ \dfrac{\partial v_M}{\partial s_1} & \cdots & \dfrac{\partial v_M}{\partial s_{12+3M}} \end{bmatrix}. \quad (29)$$

The expression for $H(\hat{\mathbf{x}}_k)$ is quite lengthy, and can be found in [14].

B. Measurement of Position and Velocity From Image Sequences

In the following, we relate optical flow $\mathbf{u} = [u\ v]^T$, image-plane position $\mathbf{x} = [x\ y]^T$ and observed gray values $g(\mathbf{x},t)$ obtained from a digital camera at time t. When denoting a specific time is unimportant, the variable t is omitted to simplify notation.

As in [15], we assume that $g(\mathbf{x},t_{k-1})$ is observed at time in t_{k-1}, and $g(\mathbf{x},t_k)$ is observed at time t_k. A

local neighborhood around position $\mathbf{x}$ is assumed to have been displaced between t_{k-1} and t_k by an optical flow vector $\mathbf{u}$. To determine $\mathbf{u}$, we minimize $\sum[g(\mathbf{x},t_k) - g(\mathbf{x} - \mathbf{u}\Delta t, t_{k-1})]^2$, where $\Delta t = t_k - t_{k-1}$ is constant for all k and is assumed small. Without loss of generality, let $\Delta t = 1$. The minimization produces two coupled nonlinear equations in terms of the two unknown components of $\mathbf{u}$. In the special case of a *gray value corner*, an elegant closed-form solution can be found [15]. Unfortunately, the solution in [15] requires special gray-value *corners* where the principal curvatures of $g(\mathbf{x},t)$ are aligned with the CCCS, thereby making the second-order spatial derivative g_{xy} zero. In the following, we generalize the feature point in [15] to what we term a *gray value extremum point* and find an alternative closed-form solution.

Assume that the origin $\mathbf{x}_0 = [x_0 \ y_0]^T$ of the coordinate system is at the center of the local neighborhood. The observed gray-level surface can be approximated by a second-order Taylor expansion about $\mathbf{x}_0 = (0,0)$ [15]. In the following, we let the true gray-level surface be modeled as

$$f(\mathbf{x}) = f_0 + f_x x + f_y y + \tfrac{1}{2} f_{xx} x^2 + f_{xy} xy + \tfrac{1}{2} f_{yy} y^2 \tag{30}$$

where f_x, f_y are partial derivatives of $f(\mathbf{x})$ with respect to x and y, f_{xx}, f_{xy}, and f_{yy} are second partial derivatives.

We index samples of the observed gray level within a $(2P+1) \times (2P+1)$ pixel window for each feature point as

$$g(\mathbf{x}_j) = f(\mathbf{x}_j) + n_{gj} \qquad n_{gj} \sim N(0,\sigma_g^2), \qquad j = 1,2,\ldots,N \tag{31}$$

where integer coordinates $x_j = (j-1) \bmod (2P+1) - P$, $-P \le x_j \le P$, and $y_j = P - (j-1) \bmod (2P+1)$, $-P \le y_j \le P$, where P is an integer, and $N = (2P+1)^2$, representing the number of raster points in the window. For example, $P = 1$ and $P = 2$ correspond to 3×3 and 5×5 windows, respectively. In (31), n_{gj} are independent Gaussian random noise samples with variance σ_g^2 that correspond to model errors. The determination of σ_g is discussed in the next section.

In matrix-vector notation, we write (31) as

$$\mathbf{G} = \mathbf{A}\mathbf{f} + \mathbf{n}_g \tag{32}$$

where

$$\mathbf{G} = [g(x_1,y_1), g(x_2,y_2), \ldots, g(x_N,y_N)]^T$$

$$\mathbf{A} = \begin{bmatrix} 1 & x_1 & y_1 & \frac{1}{2}x_1^2 & x_1y_1 & \frac{1}{2}y_1^2 \\ 1 & x_2 & y_2 & \frac{1}{2}x_2^2 & x_2y_2 & \frac{1}{2}y_2^2 \\ \vdots & & & & & \\ 1 & x_N & y_N & \frac{1}{2}x_N^2 & x_Ny_N & \frac{1}{2}y_N^2 \end{bmatrix}$$

$\mathbf{f} = [f_0, f_x, f_y, f_{xx}, f_{xy}, f_{yy}]^T$ and $\mathbf{n}_g = [n_{g1}, n_{g2}, \ldots n_{gN}]^T$.

For a given estimate of $\mathbf{f}$, the squared error is

$$\mathbf{e}^2 = (\mathbf{G} - \mathbf{A}\mathbf{f})^T \mathbf{W} (\mathbf{G} - \mathbf{A}\mathbf{f}) \tag{33}$$

where $\mathbf{W}$ is a given weighting matrix set as $\mathbf{W} = 1/\sigma_g^2 \mathbf{I}$. Then, setting $\partial \mathbf{e}^2 / \partial \mathbf{f}$ to zero, the usual least-squares estimate is obtained

$$\hat{\mathbf{f}} = (\mathbf{A}^T\mathbf{W}\mathbf{A})^{-1}\mathbf{A}^T\mathbf{W}\mathbf{G} \tag{34}$$

which is also the maximum likelihood estimate. Under Gaussian noise assumptions,

$$\hat{\mathbf{f}} \sim N[\mathbf{f}, (\mathbf{A}^T\mathbf{W}\mathbf{A})^{-1}]. \tag{35}$$

The parameter error vector $\varepsilon_f = \mathbf{f} - \hat{\mathbf{f}}$ is zero-mean with covariance matrix [16]

$$\mathrm{cov}(\varepsilon_f) = (\mathbf{A}^T\mathbf{W}\mathbf{A})^{-1} \tag{36}$$

which is constant. For the 3×3 window size used in simulations in the next section,

$$\mathrm{cov}(\varepsilon_f) = \sigma_g^2 \begin{bmatrix} 40 & 0 & 0 & -48 & 0 & -48 \\ 0 & 12 & 0 & 0 & 0 & 0 \\ 0 & 0 & 12 & 0 & 0 & 0 \\ -48 & 0 & 0 & 144 & 0 & 0 \\ 0 & 0 & 0 & 0 & 18 & 0 \\ -48 & 0 & 0 & 0 & 0 & 144 \end{bmatrix}$$

Letting $\mathbf{f}(t_k) = [f_0(t_k) \ f_x(t_k) \ f_y(t_k) \ f_{xx}(t_k) \ f_{xy}(t_k) \ f_{yy}(t_k)]$ denote derivatives of the gray-value surface at time t_k, the squared error over the N points is

$$\sum_{j=1}^{N} [g(\mathbf{x}_j,t_k) - f_0(t_{k-1}) - f_x(t_{k-1})(x_j - u) - f_y(t_{k-1})(y_j - v) - \tfrac{1}{2} f_{xx}(t_{k-1})(x_j - u)^2 - f_{xy}(t_{k-1})(x_j - u)(y_j - v) - \tfrac{1}{2} f_{yy}(t_{k-1})(y_j - v)^2]^2. \tag{37}$$

Now, we assume there exists an extremum feature point on the gray-level surface in the local area of interest and we wish to find the displacement vector $\mathbf{u}$ and its position $\mathbf{x}$ in the local window. Rather than align the axes of the local coordinate system to force f_{xy} to zero as in [15], we instead differentiate (37). After some algebra, we obtain

$$-[f_{xx}(t_{k-1})u + f_{xy}(t_{k-1})v] [\bar{g}(t_k) - \bar{g}(t_{k-1}) - \tfrac{1}{2} f_{xx}(t_{k-1})u^2 - f_{xy}(t_{k-1})uv - \tfrac{1}{2} f_{yy}(t_{k-1})v^2] + \bar{x}^2 f_{xx}(t_{k-1})[f_x(t_k) + f_{xy}(t_{k-1})v + f_{xx}(t_{k-1})u] = 0 \tag{38}$$

and

$$-[f_{xy}(t_{k-1})u + f_{yy}(t_{k-1})v] [\bar{g}(t_k) - \bar{g}(t_{k-1}) - \tfrac{1}{2} f_{xx}(t_{k-1})u^2 - f_{xy}(t_{k-1})uv - \tfrac{1}{2} f_{yy}(t_{k-1})v^2] + \bar{y}^2 f_{yy}(t_{k-1})[f_y(t_k) + f_{xy}(t_{k-1})u + f_{yy}(t_{k-1})v] = 0 \tag{39}$$

where $\overline{g}(t_{k-1})$ and $\overline{g}(t_k)$ denote average gray levels over the N points, and $\overline{x}^2$, $\overline{y}^2$ denote mean-square values of integer coordinates in the $(2P+1)\times(2P+1)$ window. For the case of a 3×3 window, $\overline{x}^2 = \overline{y}^2 = 2/3$. In (38) and (39), we have used the conditions $f_x(t_{k-1}) = f_y(t_{k-1}) = 0$.

Expanding (30) about $\mathbf{x}$ in a second-order Taylor series, with the assumption that $f((\mathbf{x}-\mathbf{u}\Delta t, t_{k-1}) = f(\mathbf{x},t_k)$, where $\Delta t = t_k - t_{k-1}$ is small,

$$\begin{aligned} f(\mathbf{x},t_k) - f(\mathbf{x},t_{k-1}) &\approx \begin{bmatrix} f_x(t_{k-1}) \\ f_y(t_{k-1}) \end{bmatrix}^T \begin{bmatrix} u \\ v \end{bmatrix} \\ &\quad + \begin{bmatrix} u \\ v \end{bmatrix}^T \begin{bmatrix} f_{xx}(t_{k-1}) & f_{xy}(t_{k-1}) \\ f_{xy}(t_{k-1}) & f_{yy}(t_{k-1}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \\ &= f_x(t_{k-1})u + f_y(t_{k-1})v + \tfrac{1}{2}f_{xx}(t_{k-1})u^2 \\ &\quad + f_{xy}(t_{k-1})uv + \tfrac{1}{2}f_{yy}(t_{k-1})v^2. \end{aligned} \tag{40}$$

Using $f_x(t_{k-1}) = f_y(t_{k-1}) = 0$, and $f(\mathbf{x},t_k) - f(\mathbf{x},t_{k-1}) \approx \overline{g}(t_k) - \overline{g}(t_{k-1})$, (40) simplifies to

$$\begin{aligned} \overline{g}(t_k) - \overline{g}(t_{k-1}) - \tfrac{1}{2}f_{xx}(t_{k-1})u^2 \\ - f_{xy}(t_{k-1})uv - \tfrac{1}{2}f_{yy}(t_{k-1})v^2 = 0. \end{aligned} \tag{41}$$

Substituting (41) into (38) and (39), using the fact that $\overline{x}^2 f_{xx}(t_{k-1})$ and $\overline{y}^2 f_{yy}(t_{k-1})$ are non-zero, we solve for the optical flow:

$$u_k = \frac{f_y(t_k)f_{xy}(t_{k-1}) - f_x(t_k)f_{yy}(t_{k-1})}{f_{xx}(t_{k-1})f_{yy}(t_{k-1}) - f_{xy}(t_{k-1})^2} \tag{42}$$

$$v_k = \frac{f_x(t_k)f_{xy}(t_{k-1}) - f_y(t_k)f_{xx}(t_{k-1})}{f_{xx}(t_{k-1})f_{yy}(t_{k-1}) - f_{xy}(t_{k-1})^2} \tag{43}$$

and the corresponding extremum point position is given by equating the first-order spatial derivatives of (30) to zero:

$$x_k = \frac{f_y(t_k)f_{xy}(t_k) - f_x(t_k)f_{yy}(t_k)}{f_{xx}(t_k)f_{yy}(t_k) - (f_{xy}(t_k))^2} \tag{44}$$

$$y_k = \frac{f_x(t_k)f_{xy}(t_k) - f_y(t_k)f_{xx}(t_k)}{f_{xx}(t_k)f_{yy}(t_k) - (f_{xy}(t_k))^2}. \tag{45}$$

Define $\mathbf{y}_{\mathbf{f}_i} = [x_i \;\; y_i \;\; u_i \;\; v_i]^T$ as four components of the hybrid optical flow/position measurement vector corresponding to the ith extremum feature point based on measurement vector $\mathbf{f}_i$. By Taylor expansion about the estimated $\hat{\mathbf{f}}_i$

$$\mathbf{y}_{\mathbf{f}_i} \approx \mathbf{y}_{\hat{\mathbf{f}}_i} + \mathbf{J}|_{\mathbf{f}_i=\hat{\mathbf{f}}_i} \cdot \varepsilon_{\hat{\mathbf{f}}_i} \tag{46}$$

where $\mathbf{J}|_{\mathbf{f}_i=\hat{\mathbf{f}}_i}$ is the 4×6 Jacobian matrix of the partial derivatives of $\mathbf{y}_{\mathbf{f}_i}$ with respect to $\mathbf{f}_i$, and $\varepsilon_{\mathbf{f}_i} = \mathbf{f}_i - \hat{\mathbf{f}}_i$ is a Gaussian random vector with $\varepsilon_{\mathbf{f}_i} \sim \mathbf{N}[\mathbf{0}, (\mathbf{A}^T\mathbf{W}\mathbf{A})^{-1}]$,

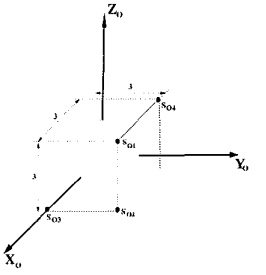


Fig. 2. Feature points in OCCS.

which is independent of i. Define measurement error $\varepsilon = \mathbf{y}_{\mathbf{f}_i} - \mathbf{y}_{\hat{\mathbf{f}}_i} = \mathbf{J}\cdot\varepsilon_{\mathbf{f}_i}$ which is also a random vector where

$$E(\varepsilon) = E(\mathbf{J}\cdot\varepsilon_{\mathbf{f}_i}) = \mathbf{J}E(\varepsilon_{\mathbf{f}_i}) = 0 \tag{47}$$

and

$$\begin{aligned} \mathrm{cov}(\varepsilon) &= \mathrm{cov}(\mathbf{J}|_{\mathbf{f}_i=\hat{\mathbf{f}}_i}\cdot\varepsilon_{\mathbf{f}_i}) \\ &= E[\mathbf{J}|_{\mathbf{f}_i=\hat{\mathbf{f}}_i}\cdot\varepsilon_{\mathbf{f}_i}\cdot\varepsilon_{\mathbf{f}_i}^T\mathbf{J}^T|_{\mathbf{f}_i=\hat{\mathbf{f}}_i}] \\ &= \mathbf{J}|_{\mathbf{f}_i=\hat{\mathbf{f}}_i}E(\varepsilon_{\mathbf{f}_i}\cdot\varepsilon_{\mathbf{f}_i}^T)\mathbf{J}^T|_{\mathbf{f}_i=\hat{\mathbf{f}}_i}. \end{aligned} \tag{48}$$

Note that $\mathrm{cov}(\varepsilon) = E(\varepsilon_{\mathbf{f}}\cdot\varepsilon_{\mathbf{f}}^T) = (\mathbf{A}_{\mathbf{f}}^T) = (\mathbf{A}^T\mathbf{W}\mathbf{A})^{-1}$, which is invariant to $\mathbf{f}_i$.

To generate values of $\mathbf{y}_{\mathbf{f}_i}$, we need to determine $\varepsilon_{\mathbf{f}_i}$. We may use Cholesky decomposition to factor $\mathrm{cov}(\varepsilon_{\mathbf{f}_i}) = \mathbf{S}\mathbf{S}^T$, where $\mathbf{S}$ is the lower triangular matrix square root of $(\mathbf{A}^T\mathbf{W}\mathbf{A})^{-1}$. Then,

$$\mathbf{y}_{\mathbf{f}_i} = \mathbf{y}_{\hat{\mathbf{f}}_i} + \mathbf{J}\cdot\mathbf{S}\mathbf{n}' \tag{49}$$

where $\mathbf{n}' \sim N(0,1)$ is a 6×1 vector of standard normal random variables.

We have shown that the estimate $\mathbf{y}_{\hat{\mathbf{f}}_i}$ is unbiased with error covariance $\mathbf{J}\mathrm{cov}(\varepsilon_{\mathbf{f}_i})\mathbf{J}^T$. Note that $\mathbf{J}|_{\mathbf{f}_i=\hat{\mathbf{f}}_i}$ is time varying, whereas $\mathrm{cov}(\varepsilon_{\hat{\mathbf{f}}_i})$ is time invariant. In summary, the new algorithm is obtained by replacing the $2M$-dimensional measurement vector (14) with the following $4M$-dimensional measurement vector:

$$\mathbf{y}_{\mathrm{hybrid}}(t_k) = [\mathbf{y}_{\mathbf{f}_1}(t_k)^T\mathbf{y}_{\mathbf{f}_2}(t_k)^T \ldots \mathbf{y}_{\mathbf{f}_M}(t_k)^T]^T \tag{50}$$

where $\mathbf{y}_{\mathbf{f}_i}(t_k)$ is formed for feature point i, $1 \le i \le M$ at time t_k via (42)–(45).

IV. EXPERIMENTAL RESULTS

A. Synthetic Simulation

We have developed a synthetic testbed to evaluate the proposed 3-D trajectory estimation algorithm, consisting of a stationary "pin-hole" camera and a transparent moving cube of size $3\times 3\times 3$. Four of the cube's vertices are arbitrarily chosen as feature points ($M = 4$) as shown in Fig. 2.

TABLE II
Spatial Resolutions for Experiments

Spatial Resolution (pixels)	σ_p (in focal lengths)
256×256	0.004
512×512	0.002
1024×1024	0.001

TABLE III
Gray-Value Resolutions for Experiments

Gray-Value Resolution (bit)	σ_g
8	0.2887
10	0.0722
12	0.018

Since the purpose of this study is limited to validating the new trajectory estimation approach, perfect feature point association from frame to frame is assumed. The measurements are generated by (49), where sources of noise are finite spatial resolution as well as gray-value resolution. The measurement noise covariance matrix is given by (48). In the simulations, a 3×3 pixel window is used ($N = 9$). The spatial resolution has interpixel distance with standard deviation σ_p given in units of focal length as in Table II. The corresponding gray-value resolutions are given in Table III, where a dynamic range of 0 (black) to 255 (white) is used.

The image plane is chosen to be a unit square (measured in terms of focal lengths) with the origin at the center. For a unit focal length, this configuration is equivalent to a camera system with a 53° field of view. Its origin is located at the center. The spatial resolution parameter σ_p is easily obtained for the unit square. For example, for a 256×256 pixel resolution, $\sigma_p = 0.004$. The gray-value resolution σ_g is obtained by determining the variance of the uniformly distributed quantization error and using a Gaussian model with the same variance. For example, 8 bit gray-value resolution achieves a standard deviation $\sigma_g = \sqrt{[0.5 - (-0.5)]^2/12} \approx 0.2887$ (see Table II.)

Uncertainty in the initial state estimates is simulated via

$$\hat{\mathbf{x}}_0(i) = [1 \pm (1 + r)e]\mathbf{x}_0(i),$$

$$i = 1, 2, \ldots, 5, 10, 11, \ldots, 12 + 3M \qquad (51)$$

where r is uniformly distributed in $(0, 1)$ and e is a chosen constant. A value of $e = 0.2$ is used, corresponding to a crude initial estimation with errors in the range 20%–40%. We note that in the complete absence of initialization information, batch techniques are more appropriate [6, 7]. No noise is added to initial state estimates (s_6–s_9) because the OCCS and CCCS are assumed to be initially aligned.

The selection of the process noise covariance matrix is important to the overall performance of the Kalman filter. Ideally, the components of the process noise should be correlated because of the constraints imposed by rigid motion on the feature points. However, the correlation is not easy to determine. Here we choose the process noise covariance matrix, somewhat arbitrarily, to be a constant diagonal matrix with all the diagonal entries having the value 5×10^{-6}. Given $\hat{\mathbf{x}}_0$ and $\mathbf{x}_0$, the initial error covariance matrix is computed as $P_0 = d^2 I$ where I is a $(12 + 3M) \times (12 + 3M)$ identity matrix and d is the largest component in the vector $|\hat{\mathbf{x}}_0 - \mathbf{x}_0|$. For the simulations, the average absolute error in position measurements, is defined as

$$\epsilon_{\text{pm}} = \sum_R \sum_N \sum_M (|p_x(\epsilon) - p_x| + |p_y(\epsilon) - p_y|) \qquad (52)$$

and is chosen since it is independent of the choice of origin. For the optical flow measurements, average relative error is used instead. A total of $R = 30$ Monte Carlo trials are used, the image sequences contain $N = 100$ frames, and $M = 4$ feature points are tracked. The bias and mean-squared error of the estimated states is the criteria used to measure the performance of the Kalman filter. Suppose we have obtained the value of one particular state variable $\hat{x}(t)$ at time t. The bias is calculated as

$$b = \frac{1}{R} \sum_{i=1}^{R} \hat{x}_i(t) - x(t) \qquad (53)$$

and the mean-squared error is calculated as

$$\text{mse} = \frac{1}{R} \sum_{i=1}^{R} [\hat{x}_i(t) - x(t)]^2 \qquad (54)$$

where R is the total number of Monte Carlo trials, $x(t)$ is the true value of this particular state variable at time t, and the subscript i denotes the trial number.

Throughout, the EKF is used. A detailed comparison among EKF, IEKF, and iterated linear filter smoother, has revealed similar performance of each [14].

We now briefly describe a typical experiment that compares the new algorithm to the BCC algorithm. The object, which is initially far away from the camera, approaches the camera at constant translational and rotational velocity. The motion of the object, as seen in the image plane, is shown in Fig. 3 where the object is represented as a wireframe model. The motion parameters are given in Table IV.

The bias and mean-squared error of the position states (s_1 and s_2), translational velocity states (s_3–s_5), and rotational velocity states (s_{10}–s_{12}) are shown in Figs. 4 (bias) and 5 (mean-squared error). In these plots, all position and translational velocity states have been scaled by $Z_R(t)$, the true depth of the object's rotation center. The dashed line

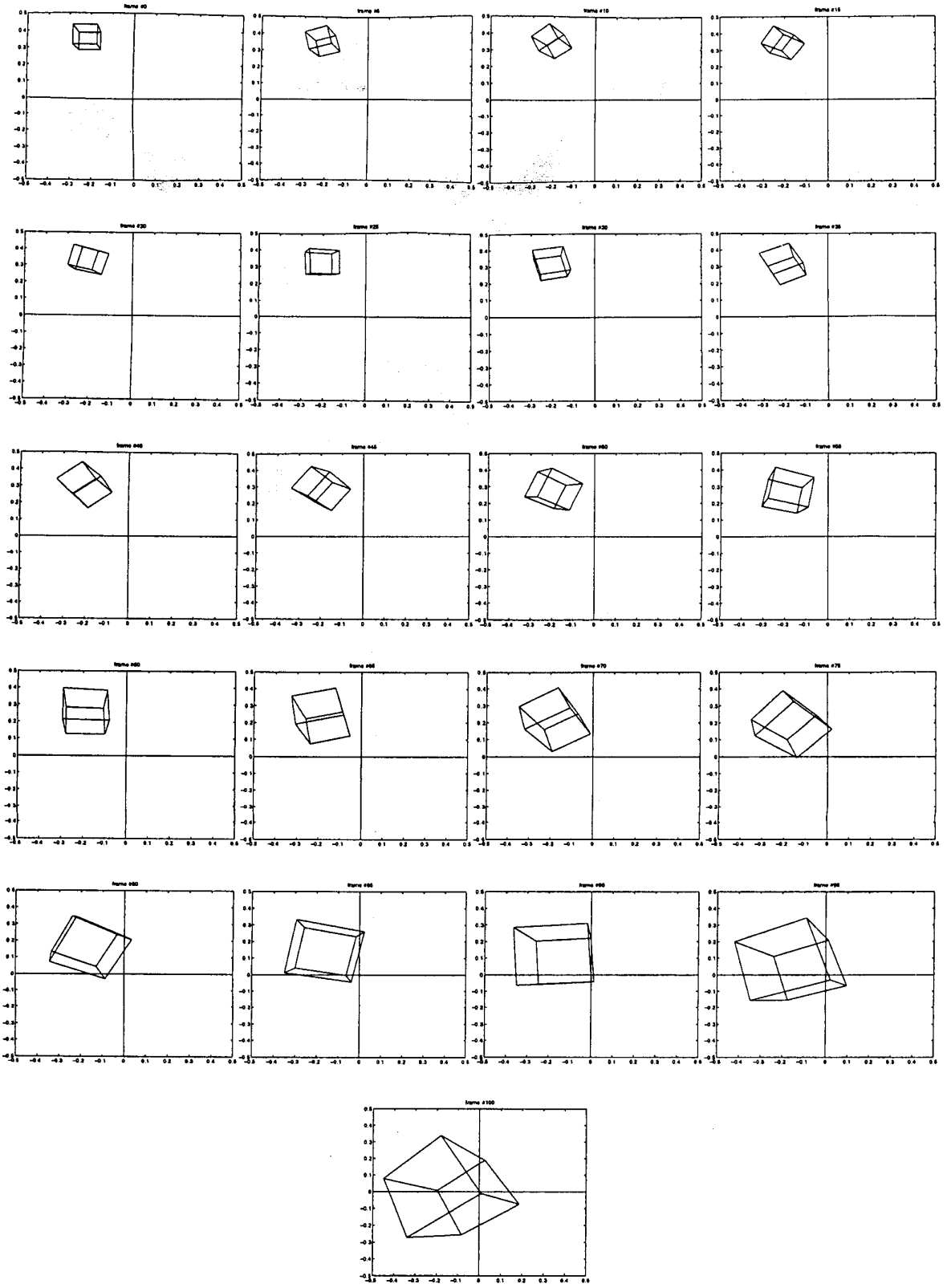


Fig. 3. Object motion in image plane.

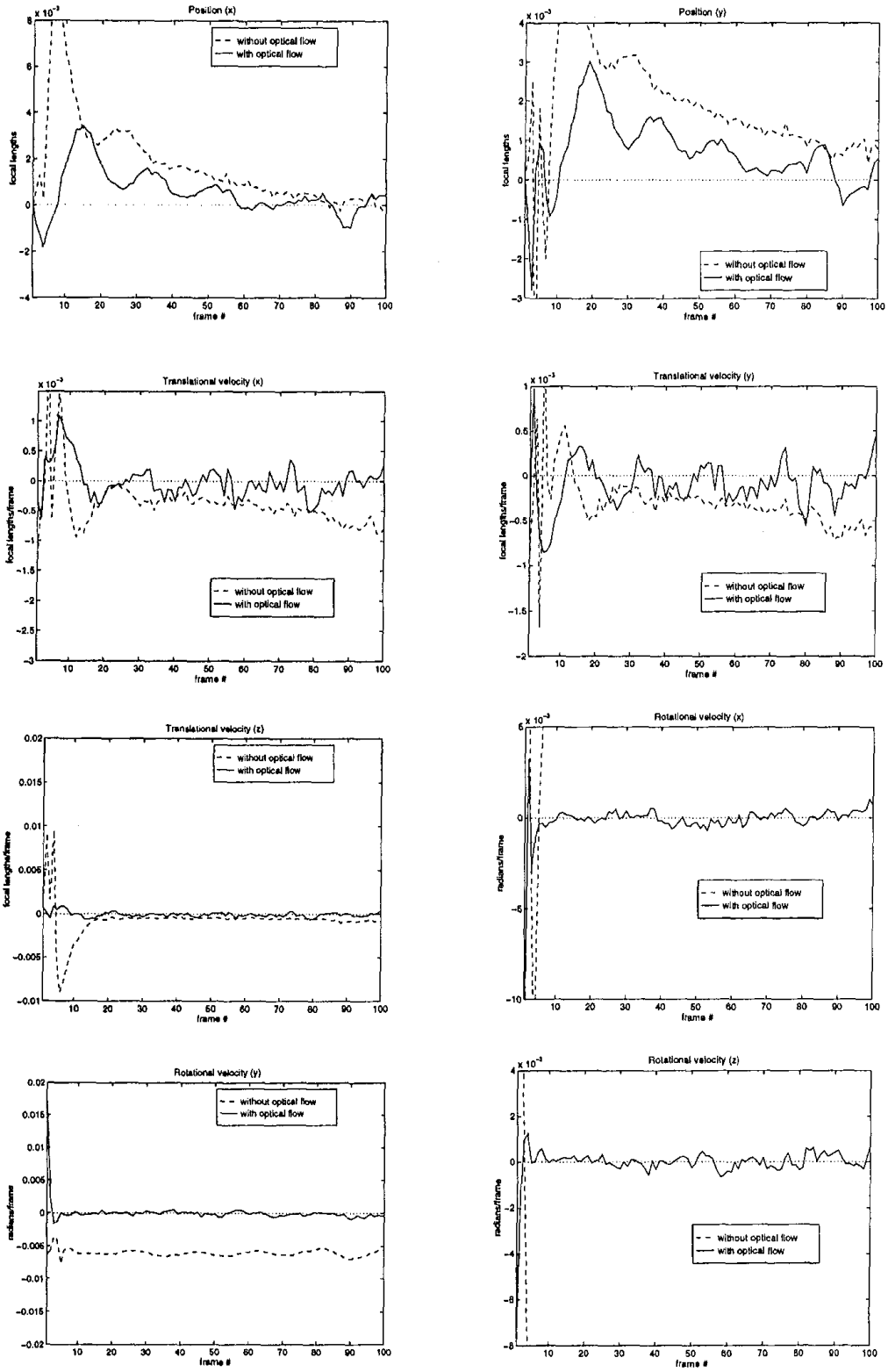


Fig. 4. Results in Monte Carlo simulations: bias (eq. (53)) averaged over 30 runs.

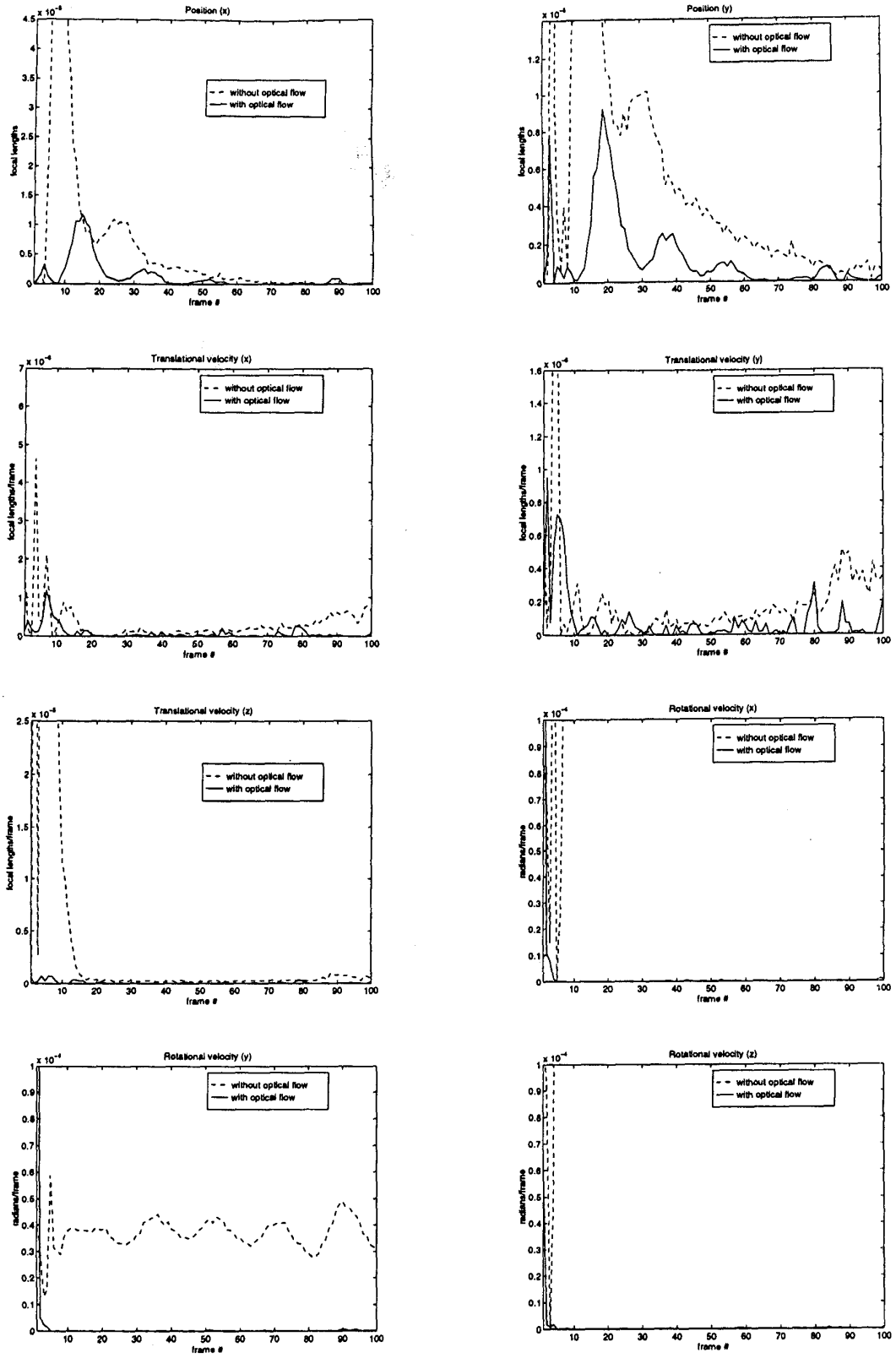


Fig. 5. Results of Monte Carlo simulations: sample mean-squared error (eq. (54)) averaged over 30 runs.

TABLE IV
Motion Parameters for Synthetic Experiment

initial position $\mathbf{s}_R(t_0)$	$[-6\ 10\ 28]^T$
translational velocity $\mathbf{T}$	$[0.05\ -0.1\ -0.2]^T$
rotational velocity ω	$[0.03\ 0.04\ 0.05]^T$

and solid line represent estimates obtained by the BCC algorithm and the hybrid feature/optical-flow algorithm, respectively. We remark that in [1], Monte Carlo results were not provided. Note that marked improvements can be achieved by the hybrid algorithm, especially in the velocity states. In the translational velocity states, significant improvements can be observed mainly during the filter initialization stage (the first 20 frames) where the bias and variance are significantly reduced. Improvements are even greater in the rotational velocity states. While the BCC algorithm diverges, the hybrid algorithm converges quickly and locks on to the true values.

To examine how different spatial and gray-value resolutions affect the estimation of state variables, a set of different spatial and gray-value resolutions were used. Typical results are shown in Fig. 6 where 8, 10 and 12 bit gray-value resolutions are used with a fixed spatial resolution of 512^2 pixels. Similar results were observed for the other states but are not shown due to space limitations. Table V shows the effect of varying the spatial resolution: 256^2, 512^2, and 1024^2 pixel resolutions were used with a 10 bit gray-level resolution. It is interesting to note that the choice among gray-level resolutions had a larger impact than the choice of spatial resolutions.

Finally, we have also examined the effect of increasing the number of Monte Carlo trials One might argue that the large initial estimation errors produced by the algorithms can be reduced or removed by simply increasing the number of Monte Carlo trials. However, this is *not* the case; a significant bias still exists even after averaging 1000 Monte Carlo trials.

Similar results were obtained for the case of unmodeled translational acceleration. In addition,

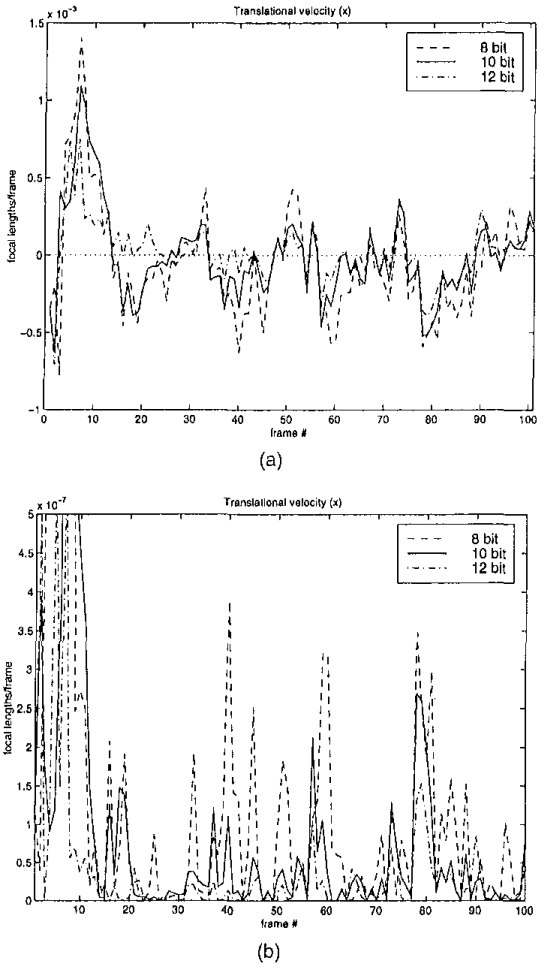


Fig. 6. Results of varying gray-level resolution for fixed 512^2 pixel spatial resolution. (a) Bias. (b) Mean-square error for x-translational velocity state.

we have also investigated the case where the object deviates from a 3-D motion model more drastically, corresponding to the case where the trajectory changes *abruptly* from one model to another, simulating a sudden object maneuver. Here, the optical flow measurements improved the accuracy of the state estimates.

TABLE V
Mean-Squared Error as Function of Spatial Resolution

Spatial Resolution (pixels)	Position (x)	Position (y)	Trans. Vel. (x)	Trans. Vel. (y)
256×256	1.25×10^{-6}	1.26×10^{-6}	4.03×10^{-8}	4.20×10^{-8}
512×512	1.24×10^{-6}	1.24×10^{-6}	4.03×10^{-8}	4.18×10^{-8}
1024×1024	1.21×10^{-6}	1.21×10^{-6}	4.01×10^{-8}	4.07×10^{-8}
Spatial Resolution (pixels)	Trans. Vel. (z)	Rot. Vel. (x)	Rot. Vel. (y)	Rot. Vel. (z)
256×256	5.53×10^{-8}	1.06×10^{-7}	1.02×10^{-7}	7.95×10^{-8}
512×512	5.53×10^{-8}	1.05×10^{-7}	1.01×10^{-7}	7.90×10^{-8}
1024×1024	5.51×10^{-8}	1.05×10^{-7}	1.01×10^{-7}	7.89×10^{-8}

Note: Gray-value resolution fixed to 10 bits.

TABLE VI
Motion Parameters for PUMA2 Experiment From [17]

initial position $\mathbf{s}_R(t_0)$	$[-2\ 0\ 19.1141]^T$
translational velocity $\mathbf{T}$	$[0\ 0\ 0]^T$
rotational velocity ω	$[0\ 0\ -0.0698]^T$

TABLE VII
Ground Truth of Four Chosen Feature Points at Frame 1 From [17]

Feature Points (number)	Structure Ground Truth (X)	Structure Ground Truth (Y)	Structure Ground Truth (Z)
1	−9.247	−3.341	29.494
2	−7.215	−0.457	31.020
3	−3.192	−2.258	31.800
4	1.230	3.624	19.613

B. Experiment of a Real Image Sequence

In this experiment, we obtained the 30-frame PUMA2 image sequence used in [17] and [18] which was taken by connecting a camera to the end of a puma robot arm. The robot arm was rotated for 120 deg over 30 frames. The rotation between two succesive frames is approximately 4 deg (0.0698 rad). The radius of the robot arm is about 2 ft. The image size is 512 by 484 pixels. The field of view is 41.67491 and 39.52927 deg in the x and y axes, respectively. The focal length in pixels is 673.0424 pixels. As in [17], the world coordinate system is set to be the first-frame image coordinate system with the origin located at the center of the image plane. Twelve points with known locations in the world coordinate are given in [17]. We use four of these given in Table VII and Fig. 7. Unlike [17] where the camera motion is being estimated, we assume that the camera is fixed, and the relative motion with respect of the camera is of interest. We may think of the motion as the room rotating around the rotation center O_o. An initial guess of the world coordinate, O_o, is obtained by measuring its location on the image plane from the first two frames and then using (3) without noise. The motion parameters of the PUMA2 sequence are listed in Table VI. The measurements of the feature point coordinates are obtained by using the method discussed in last section. The spatial resolution is 512^2 pixels, the gray-value resolution is 8 bits. The method of measuring optical flow discussed in last section is not feasible because the displacements were too large in this image sequence. We instead use the position difference,

$$\mathbf{u}(k) = \frac{\mathbf{p}(k+1) - \mathbf{p}(k-1)}{2\Delta t} \tag{55}$$

where $\mathbf{p}$ is the measurement of feature point position.

According to [18], there should exist only rotation around the z axis. But as pointed out by [17], the

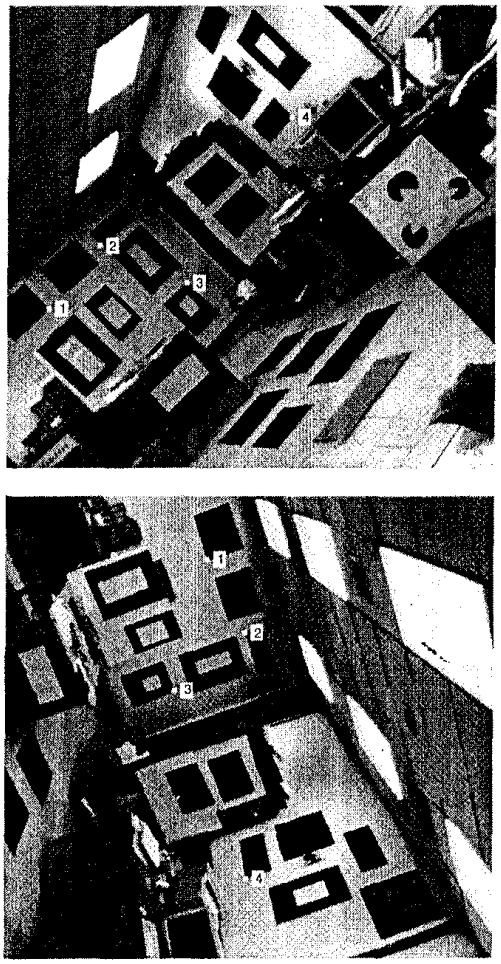


Fig. 7. Locations of feature points in first and last frame of PUMA2 sequence.

CCCS changes slightly from frame to frame. Thus there exists small amounts of translational motion *not accounted for* by the motion model. That makes a comparision of the rotation center's true position to its estimate unavailable. Our results (see Fig. 8) show that there exists negative and positive biases on translational velocity components s_3 and s_4, respectively. The results are presented in Figs. 8 (bias) and 9 (mean-squared error). The plots of the velocity states are scaled by $Z_R(t)$, the true depth of the rotation center. The dashed line and solid line represent estimates obtained by the BCC and our hybrid algorithm, respectively. Although the BCC algorithm gives accurate 3-D motion estimates in this case, the new hybrid algorithm performs somewhat better.

V. CONCLUSION

In conclusion, we have proposed a new algorithm that estimates 3-D motion and structure of a rigid

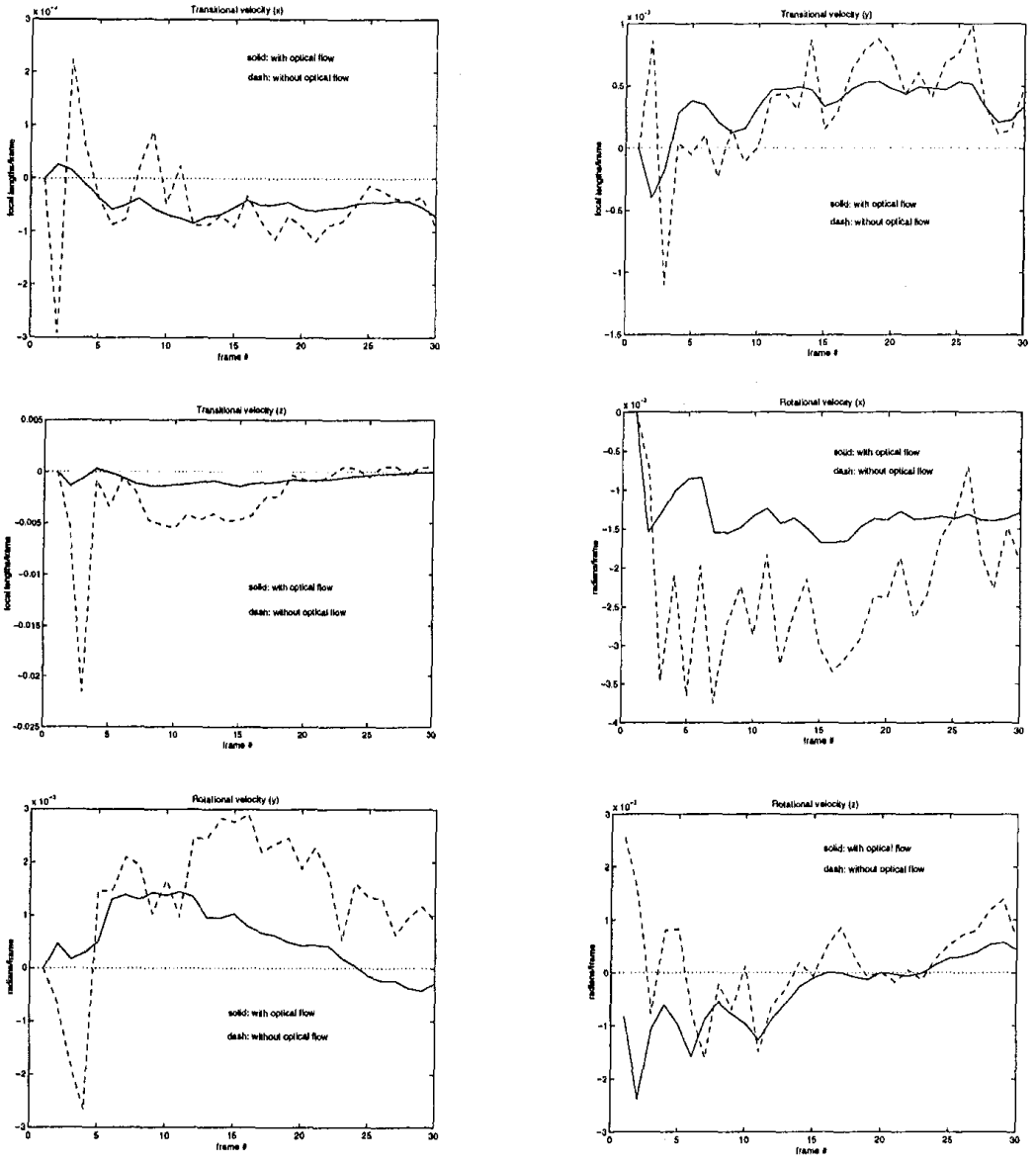


Fig. 8. PUMA2 results of Monte Carlo simulations: bias (eq. (53)) averaged over 30 runs.

object from a sequence of images when image plane velocity information is available at feature point locations. The main observed improvements were in the estimation of velocity states and in the reduction of occasional large swings in the estimation error. Experiments on actual images have shown that even with very crude estimates of optical flow such as by (19), performence improvements were observed.

REFERENCES

[1] Broida, T. J., Chandrashekhar, S., and Chellappa, R. (1990) Recursive 3-D motion estimation from a monocular image sequence. *IEEE Transactions on Aerospace and Electronic Systems*, **26**, 4 (1990), 639–656.

[2] Aggarwal, J. K., and Nandhakumar, N. (1988) On the computation of motion from sequences of images—A review. *Proceedings of the IEEE*, **76**, 8 (1988), 917–935.

[3] Heeger, D. J., and Jepson, A. D. (1992) Subspace methods for recovering rigid motion I: Algorithm and implementation. *International Journal of Computer Vision*, **7**, 2 (1992), 95–117.

[4] Iu, S., and Wohn, K. (1990) Estimation of general rigid body motion from a long sequence of images. In *International Conference on Pattern Recognition*, 1990, 217–219.

[5] Silvën, O., and Repo, T. (1992) Solutions for real-time visual tracking. In *SPIE*, vol. 1708, *Applications of Artificial Intelligence X: Machine Vision and Robotics*, 1992, 184–195.

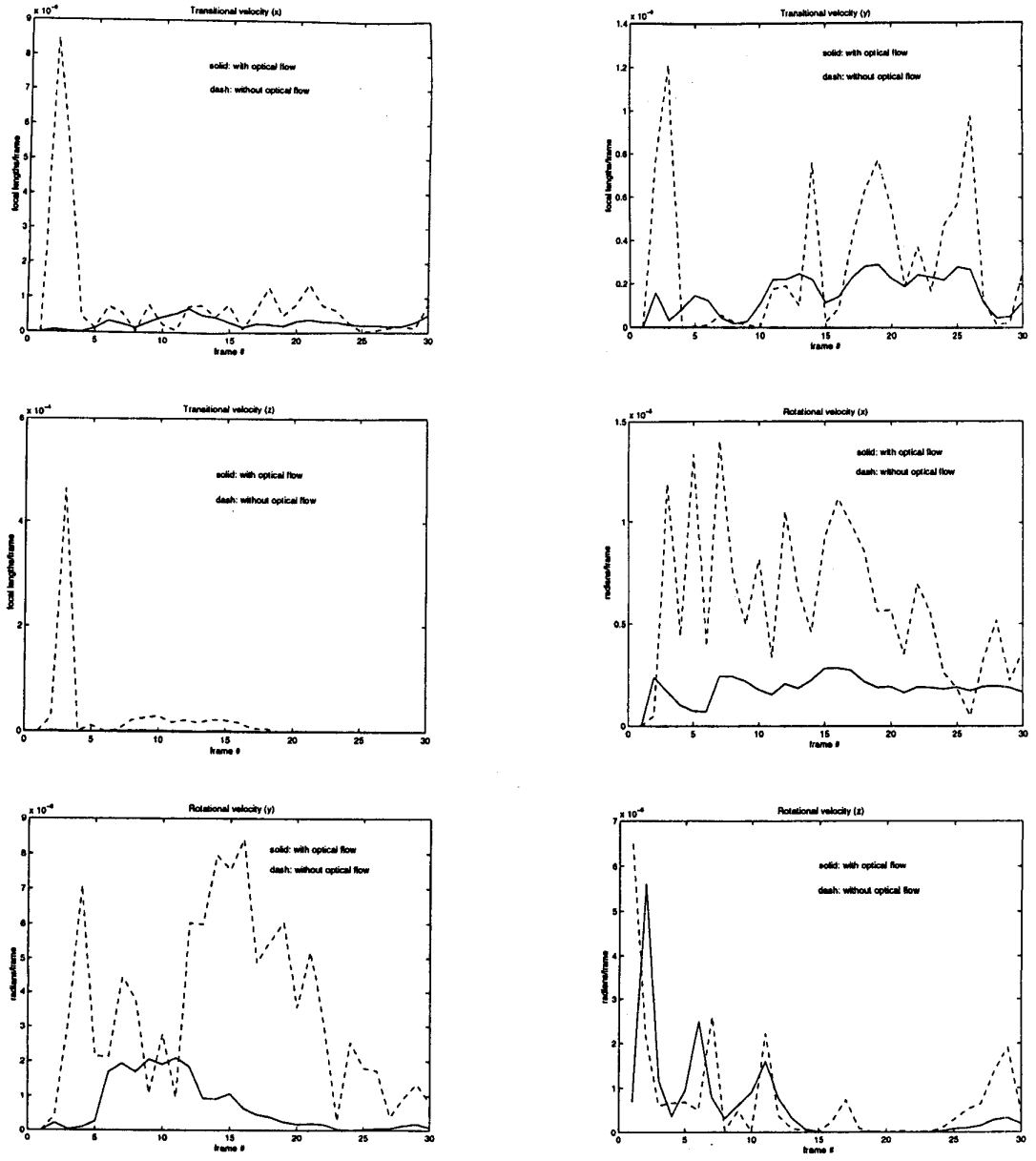


Fig. 9. PUMA2 results of Monte Carlo simulations: sample mean-squared error (eq. (54)) averaged over 30 runs.

[6] Young, G. S., Chellappa, R., and Wu, T. H. (1991)
Monocular motion estimation using a long sequence of noisy images.
In *International Conference on Acoustics, Speech and Signal Processing*, 1991, 2437–2440.

[7] Weng, J., Huang, T. S., and Ahuja, N. (1989)
Motion and structure from two perspective views: Algorithms, error analysis, and error estimation.
IEEE Transactions on Pattern Analysis and Machine Intelligence, **11**, 5 (1989), 451–476.

[8] Wu, J. J., Rink, R. E., Caelli, T. M., and Gourishankar, V. G. (1989)
Recovery of the 3-D location and motion of a rigid object through camera image (an extended Kalman filter approach).
International Journal of Computer Vision, **2**, 4 (1989), 373–394.

[9] Silvěn, O., and Repo, T. (1993)
Experiments with monocular visual tracking and environment modeling.
In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1993, 84–92.

[10] Chandrashekhar, S., and Chellappa, R. (1991)
Passive navigation in a partially known environment.
In *Proceedings of the IEEE Workshop on Visual Motion*, 1991, 2–7.

[11] Horn, B. K. P. (1986)
Robot Vision.
Cambridge, MA: The MIT Press, 1986.

[12] Barron, J. L., Fleet, D. J., and Beauchemin, S. S. (1994)
Performance of optical flow techniques.
International Journal of Computer Vision, **12**, 1 (1994), 43–77.

[13] Taalebinezhaad, M. A. (1992)
Direct recovery of motion and shape in the general space by fixation.
IEEE Transactions on Pattern Analysis and Machine Intelligence, **14**, 8 (1992), 847–853.

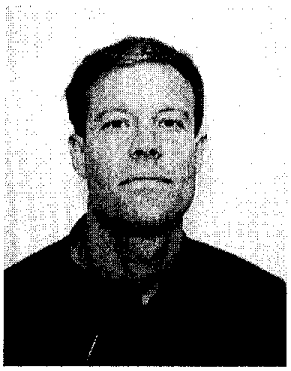
[14] Chann, R. M. (1994)
Recursive estimation of 3-d motion and structure in image sequences based on measurement transformations.
Master's thesis, Queen's University, Kingston, Ontario, 1994.

[15] Nagel, H. H. (1983)
Displacement Vector Derived from Second-order Intensity Variations in Image Sequences.
CVGIP, **21** (1983), 55–117.

[16] Scharf, L. L. (1991)
Statistical Signal Processing.
Reading, MA: Addison-Wesley, 1991.

[17] Wu, T. H., Chellappa, R., and Zheng, Q. (1995)
Experiments on estimating egomation and structure parameters using long monocular image sequences.
International Journal of Computer Vision, **15**, 1/2 (1995), 77–103.

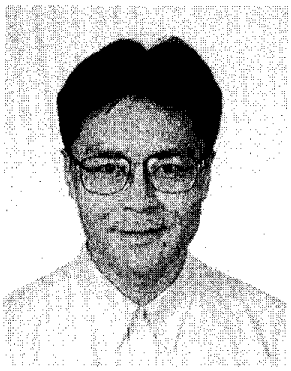
[18] Kumar, R., and Hanson, A. (1990)
Sensitivity of the pose refinement problem to accurate estimation of camera parameters.
In *International Conference on Computer Vision*, Osaka, Japan, 1990, 365–369.



Steven D. Blostein (S'83—M'88—SM'96) received his B.S. degree in electrical engineering from Cornell University, Ithaca, NY, in 1983, and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Illinois, Urbana-Champaign, in 1985 and 1988, respectively.

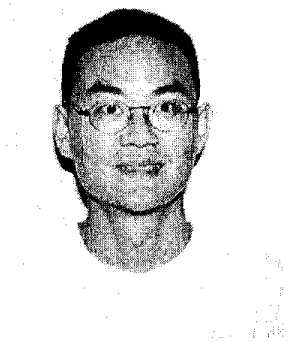
He has been on the Faculty at Queen's University since 1988 and currently holds the position of Associate Professor in the Department of Electrical and Computer Engineering. He has been a consultant to both industry and government in the areas of document image compression, motion estimation, and target tracking, and was a Visiting Associate Professor in the Department of Electrical Engineering at McGill University in 1995. His current interests lie in statistical signal processing, wireless communications and video image communications.

Dr. Blostein served as Chair of IEEE Kingston Section in 1993–1994. He currently leads the Multirate Wireless Data Access Major Project sponsored by the Canadian Institute for Telecommunications Research, is currently serving as Associate Editor for *IEEE Transactions on Image Processing*, and is a registered Professional Engineer in Ontario.



Lin Zhao received his B.Sc. (Eng) degree in electrical engineering from the Shandong Polytechnical University, Jinan, China in 1985 and the MSc degree in electrical and computer engineering from Queen's University, Kingston, Ontario, in 1998.

In 1985, he worked in Jinan Brewery Co. as an electrical engineer responsible for computer-aided process control. His current interests fall in digital signal processing and wireless communication.



Robert M. Chann (M'94) received the B.A.Sc. degree in honours computer engineering from the University of Waterloo in 1991, and the MSc degree in Electrical and Computer Engineering from Queen's University, Kingston, Ontario, in 1994.

In 1994, he worked at the Laboratory of Neuro Imaging at the University of California Los Angeles. Currently, he is employed at ISG Technologies, Inc. in Mississauga, Ontario, Canada. His research interests include medical image registration and volume visualization.