International Journal of Computer Vision, 7:2, 95–117 (1992)
© 1992 Kluwer Academic Publishers, Manufactured in The Netherlands.

Subspace Methods for Recovering Rigid Motion I: Algorithm and Implementation

DAVID J. HEEGER
NASA-Ames Research Center, mail stop 262-2, Moffett Field, CA 94035; and Psychology Department, Stanford University, Stanford, CA 94305

ALLAN D. JEPSON
Computer Science Department, University of Toronto, Toronto, Ontario M5S 1A4

Received September 21, 1990. Revised July 9, 1991.

Abstract
As an observer moves and explores the environment, the visual stimulation in his/her eye is constantly changing. Somehow he/she is able to perceive the spatial layout of the scene, and to discern his/her movement through space. Computational vision researchers have been trying to solve this problem for a number of years with only limited success. It is a difficult problem to solve because the optical flow field is nonlinearly related to the 3D motion and depth parameters.

Here, we show that the nonlinear equation describing the optical flow field can be split by an exact algebraic manipulation to form three sets of equations. The first set relates the flow field to only the translational component of 3D motion. Thus, depth and rotation need not be known or estimated prior to solving for translation. Once the translation has been recovered, the second set of equations can be used to solve for rotation. Finally, depth can be estimated with the third set of equations, given the recovered translation and rotation.

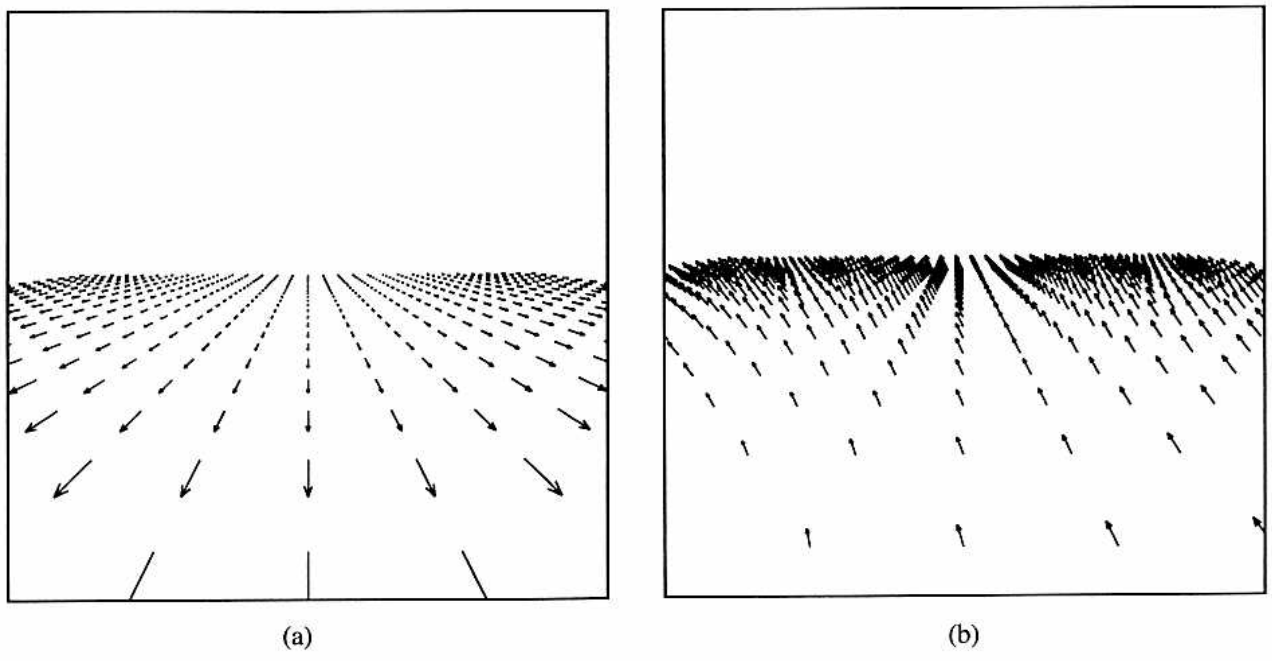
The algorithm applies to the general case of arbitrary motion with respect to an arbitrary scene. It is simple to compute, and it is plausible biologically. The results reported in this article demonstrate the potential of our new approach, and show that it performs favorably when compared with two other well-known algorithms.

1 Introduction

More than forty years ago, Gibson [1950, 1957] pointed out that visual motion perception is essential for an observer to explore and interact with his/her environment. Since that time, perception of motion has been studied extensively by researchers in the fields of visual psychophysics, visual neurophysiology, and computational vision. It is now well known that the visual system has mechanisms that are specifically suited for analyzing motion (for review, see Nakayama [1985]), and that human observers are capable of recovering accurate information about the world (e.g., three-dimensional trajectory, relative distance, shape) from motion in images (for example, Wallach and O'Connell [1953], Johansson [1975], Warren and Hannon [1988, 1990]).

The first stage of motion perception is generally believed to be the measurement of image motion. The result of this first stage is called the optical flow field, a collection of two-dimensional velocity vectors, one for each small region of the visual field. The second stage of motion perception is the interpretation of optical flow in terms of objects and surfaces in the three-dimensional world. The optical flow field depends nonlinearly on the distance from the observer to each point in the scene, on the translational velocity of the observer, and on the rotational velocity of the observer (observer rotation, throughout this article, means rotation about the center of projection of the observer's eye).

Figure 1 shows some example flow fields. Each vector represents the speed and direction of motion for each local patch of visual field. Figure 1a is a flow field



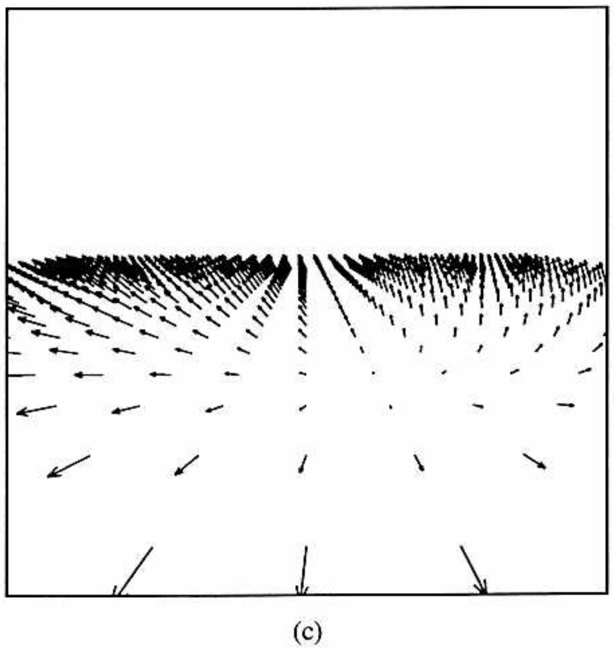


Fig. 1. Flow fields corresponding to: (a) observer translation, (b) observer rotation, (c) translation plus rotation. Each flow vector in (c) is the vector sum of the two correpsonding flow vectors in (a) and (b).

resulting from observer translation above a planar surface. Figure 1b is a flow field resulting from observer rotation, and figure 1c is a flow field resulting from simultaneous translation and rotation. Each flow vector in figure 1c is the vector sum of the two corresponding flow vectors in figures 1a and 1b.

When an observer is translating, features in the image move toward or away from a single point in the image, called the focus of expansion (FOE). The FOE in figure 1a is centered just above the horizon. When the observer is rotating as well as translating, the singularity in the flow field no longer corresponds to the translation direction. The flow field in figure 1c still contains information about the observer's translation, but that information is confounded by the rotation.

Gibson [1950, 1957] hypothesized that there is sufficient information in the flow field to recover a unique, physically correct interpretation of the 3D motion and scene structure. For pure translation (figure 1a), the FOE gives the observer motion, and the vector lengths give the scene structure. However, for translation plus rotation (figure 1c), it is not immediately clear that one could recover the 3D motion and scene structure.

This article presents a simple algorithm for recovering 3D motion and depth from optical flow, for the case of general motion (translation plus rotation) with respect to an arbitrary scene. In a companion article [Jepson & Heeger 1990], we prove that this algorithm gives unique and robust solutions. Some of this work has been reported previously [Jepson and Heeger 1989, 1990, 1991; Heeger and Jepson 1990a, 1990b, 1990c, 1991].

2 Literature Review

A large number of algorithms have been proposed for computing 3D motion and depth from image sequences; this article presents yet another. This new approach is based on the bilinear nature of the equation relating 3D motion to image motion. Since the equation is bilinear we can split the problem apart, solving first for translation, second for rotation, and third for depth.

This section provides a review of 3D motion algorithms. It is not an exhaustive review since there have been so many papers published on this topic over the past decade. The algorithms discussed here are examples that are representative of the literature. More extensive reviews have been written by Barron [1984] and by Aggarwal and Nandhakumar [1988].

2.1 Instantaneous Time vs. Discrete Time

Techniques for computing 3D motion from image sequences can be categorized either as discrete-time methods or as instantaneous-time methods. Discrete-time algorithms track a collection of isolated image features over a series of views. Given enough views of enough points, the solution for rigid structure is overdetermined.

Instantaneous-time algorithms, like the one proposed here, compute 3D motion and depth from optical flow. The flow can be measured using correlation or block-matching (e.g., see Anandan [1989]), in which each small patch of the image is compared with nearby patches. Gradient-based algorithms are a second approach to measuring flow fields (e.g., Horn and Schunk [1981], Lucas and Kanade [1981], Nagel [1987]). A third approach using spatiotemporal filtering has also been proposed (e.g., Watson and Ahumada [1985], Heeger [1987, 1988], Grzywacz and Yuille [1990], Fleet and Jepson [1990], Simoncelli et al. [1991]. Recently, Adelsen and Bergen [1986] and Simoncelli et al. [1991] have placed the spatiotemporal filter models and the gradient-based methods into a common framework.

Feature extraction and matching is an additional way to measure the flow field (for reviews of feature tracking methods, see Barron [1984] or Aggarwal and Nandhakumar [1988]. Feature tracking can thus be a starting point for an instantaneous-time motion algorithm as well as for a discrete-time algorithm. The difference between the two is the way in which the correspondence is used for subsequent processing. A discrete-time method uses the positions of the features, and an instantaneous-time method uses the velocities.

The real difference between instantaneous- and discrete-time methods is that the instantaneous approach is an approximation that is valid only for short time steps. In Section 6 we show how the instantaneous approximation breaks down when the motion from frame to frame is too great.

2.2 Discrete-Time Algorithms

Early examples of discrete-time 3D motion algorithms were proposed by Ullman [1979] and by Roach and Aggarwal [1980].

A linear discrete-time approach was later developed by several researchers [Longuet-Higgins 1981; Tsai and Huang 1984; Faugeras et al. 1987; and Weng et al. 1989]. The first step in these linear discrete-time algorithms is to compute a set of "essential" parameters (the E-matrix) from the locations of corresponding feature points in two successive frames. Then, the 3D motion parameters and depth are recovered from the E-matrix. The E-matrix algorithms are quite simple to implement and inexpensive to compute. However, the linear constraint they utilize is not the same as the constraint of rigid-body motion. The consequence of using a weaker constraint is that these algorithms are particularly sensitive to input errors. In section 6, we compare the performance of our new algorithm with the E-matrix approach.

2.3 Instantaneous-Time Algorithms

Koenderink and Van Dorn [1975, 1976, 1981] pioneered the instantaneous-time approach. They proposed using local deformations of the image (e.g., divergence, curl, shear) to measure invariant properties of motion and surface shape.

Making use of Koenderink and van Dorn's analysis, Longuet-Higgins and Prazdny [1980] and Waxman et al. [Waxman & Ullman 1985; Waxman & Wohn 1985, 1988; Waxman et al. 1987] proposed methods that solve for the motion of planar surfaces. A rigidly moving planar surface is a special case that gives rise to a second-order flow field; that is, the flow vectors vary as a quadratic function of image position. The method of Waxman et al. first subdivides the flow field into local patches that are well approximated as second order flow fields. The 3D structure and motion are then recovered, in closed form, from the parameters of the second order flow field. An advantage of these methods is that the motion in each patch is computed independently, so the methods can deal with multiple moving objects. A problem with these methods is that they are sensitive to errors in the flow-field measurements, particularly for small patches (see Waxman and Wohn [1988] and Barron et al. [1990]).

2.4 Subdividing the Problem

Several researchers have proposed splitting the problem apart, solving first for one set of motion parameters (either the translation or the rotation), and then solving for the other. Prazdny [1980] proposed solving first for the rotational component of motion. He derived a system of nonlinear equations that relates image velocity to rotation, independent of translation. This nonlinear system may be solved using iterative numerical techniques. Prazdny noted that this iterative method is computationally expensive and that it requires good initial guesses. Prazdny also noted that this method should not be applied across depth discontinuities, implicitly assuming that the surfaces in the scene are smooth.

Longuet-Higgins and Prazdny [1980] and Reiger and Lawton [1985] suggested solving first for translation, independent of rotation. Their algorithms are based on an observation that was first pointed out by Helmholtz [Southall 1962]. If two 3D points have the same image location but are at different depths, then the vector difference between the two flow vectors is oriented toward the FOE. If two 3D points project to nearby image locations, then the vector difference is oriented approximately toward the FOE. Reiger and Lawton showed that the approximation is valid when the depth difference is large, and when the two image locations are close to one another (i.e., at adjacent image locations that straddle an occlusion boundary). The Reiger-Lawton algorithm locates the FOE from local flow-vector differences. The problem with their approach is that it is particularly difficult to measure flow vectors near occlusion boundaries.

The Reiger-Lawton algorithm is similar, in spirit, to the approach presented in this article. We also subdivide the problem, solving first for translation. The difference is that our solution is not an approximation and we do not require that the flow vectors be from adjacent image locations. In section 6, we compare the performance of our algorithm with the Reiger-Lawton algorithm.

Prazdny [1983] showed that the difference between any two (not necessarily adjacent) flow vectors gives a constraint on translation, independent of rotation. Prazdny did not propose an algorithm that made use of this constraint. This constraint was the starting point for our current research.

Motivated by our research, Sundareswaran [1991] recently proposed two algorithms. The first is an approximate method that solves for rotation independent of translation, and the second is an exact method that solves for translation independent of rotation.

2.5 Least-Squares Algorithms

Bruss and Horn [1983] (see also Horn [1986]) proposed a global approach, that combines information throughout the visual field, to choose the 3D motion that best accounts (in a least-squares sense) for the measured flow field. They developed three algorithms for three different cases. The first solves for translation when the rotation is zero. The second solves for rotation when translation is zero. The third deals with the case of general 3D motion. Bruss and Horn derived closed-form solutions for the first two cases. For the general case, they reduced the least-squares residual function to one that involves only the unknown translation. They proposed an iterative numerical (e.g., gradient-descent) procedure to find the solution.

Adiv [1985] suggested an alternative algorithm for minimizing the same least-squares residual function that was originally used by Bruss and Horn [1983]. Like Waxman et al. (references above), Adiv subdivides the flow field and computes the motions independently for each patch. Adiv avoids using an iterative gradient descent procedure by sampling the solution space of all candidate translation directions. For each patch and for each candidate translation, he computes a residual value. The translation with the minimum residual value is chosen as the estimate for that patch. Patches that share the same solution are then grouped together since they correspond to surfaces in the scene undergoing the same 3D motion.

Maybank [1987] also considered the least-squares approach and derived some theoretical results and approximations that are closely related to our work (see Jepson & Heeger [1990]).

The algorithm described here minimizes the same least-squares residual function used by Bruss and Horn [1983], by Adiv [1985], and by Maybank [1987]. We also reduce the residual function to one that involves only the unknown translation. The important difference in our approach is the way by which we eliminate the unknown rotation and depth values. The resulting algorithm is considerably more efficient since most of the computation is done off line. We evaluate the residual function simply as a linear summation of the input flow vectors, weighted by a fixed set of precomputed coefficients. Like Adiv and Maybank, we avoid using an iterative gradient-descent procedure by sampling the solution space of all candidate translation directions.

Other researchers have also proposed techniques that pick solutions by minimizing a residual function over the discretely sampled solution space. Prazdny [1981] proposed a method that samples the space of all candidate rotations (a three-dimensional solution space). Subtracting the correct rotational component from the flow field gives vectors that all point toward or away from the FOE. Prazdny's residual function measures the extent to which these vectors intersect.

Ballard and Kimball [1983] used a generalized Hough transform to solve for the 3D motion parameters. The solution space (the five-dimensional space of all candidate translations and rotations) is sampled, and each input flow vector "votes" for all of the solutions with which it is consistent. The values that receive the greatest number of votes are taken to be solutions. An advantage of this approach is that it is possible to represent multiple solutions as multiple peaks, corresponding to differently moving objects. However, the method requires that the depth already be known, and it requires evaluating candidate solutions throughout the entire five-dimensional solution space.

2.6 Direct Methods

Horn et al. [Horn and Negahdaripour 1987; Horn and Weldon 1988; Negahdaripour and Horn 1989] proposed several "direct" methods for recovering the 3D motion parameters. In these methods, 3D motion is computed from the spatiotemporal gradients of image brightness, rather than from feature positions or from flow vectors. Horn et al. argue that these methods avoid the difficulty of computing the flow field. Each of these "direct" methods applies only for a special case, for example, when there is no rotation.

In appendix B, we extend out algorithm to be a "direct" method that works directly from the spatiotemporal gradients of image brightness and applies for the general case of arbitrary 3D motion.

2.7 Incremental Methods

A number of authors have suggested sequential/incremental methods for recovering 3D motion and/or structure that take advantage of information over extended periods of time. The goal of these efforts is to reduce the error by using more information. Ullman [1984] proposed a discrete-time, feature-based, incremental rigidity scheme. The incremental rigidity scheme has the advantage that even though it prefers rigid interpretations, it can recover the structure of nonrigid objects that change shape slowly over time. Broida and Chellappa [1986] and Faugeras et al. [1987] advocated feature-based extended Kalman filtering routines. Heel [1989b] proposed an instantaneous-time algorithm that uses an extended Kalman filter. In this method, an initial guess of the depth map is used to estimate the 3D motion parameters. Then the estimated motion parameters are used to update the depth estimates. The procedure goes back and forth like this until the depth estimates converge. Heel [1989a, 1990] also proposed a "direct" sequential method that computes 3D motion directly from the spatiotemporal gradients of image brightness.

It is straightforward to extend our approach to be a sequential estimator that takes advantage of information over extended periods of time. Unlike Heel, we do not need depth estimates to update the motion-parameter estimates. We can still take advantage of prior information on depth if that prior information is available, but we do not depend on it.

2.8 Evaluating the Algorithms

In summary, there are several criteria that are important when evaluating the various 3D motion algorithms.

Stability. The method must not be too sensitive to errors in the image velocity measurements. Algorithms that use only local measurements are particularly sensitive to input errors. Methods that rely on approximations are also error sensitive. Our method is comparatively stable because it is exact, and because it is global (we combine flow measurements over space and/or time).

Generality. A number of algorithms solve only a part of the problem, or solve it only under restricted conditions. Some algorithms are exact only for planar surfaces. Some methods can not deal with a scene in which there are objects moving differently with respect to one another. Some approaches solve for depth only if the 3D motion is already known. Other approaches solve for the 3D motion parameters only if the depth is already known. Other approaches solve for the translational component of motion only if the rotational component is already known, or vice versa.

Our new approach handles the case of arbitrary motion with respect to an arbitrary scene, and we are currently extending it to deal with multiple motions. Like Ballard and Kimball [1983], we could look for multiple minima in the residual function. Or like Adiv [1985] and Waxman and Wohn [1988], we could subdivide the image into local patches and compute the 3D motions independently for each patch. Any method is likely to be error prone when given input from a very limited field of view. Sequential processing over an extended period of time may compensate for instability that results from using small patches.

Efficiency. Algorithms that make use of iterative numerical techniques (e.g., gradient descent) in a high-dimensional solution space are not practical since they are prohibitively expensive in terms of compute time. In addition, iterative methods for solving nonlinear equations often require good initial guesses of the unknowns. We advocate a highly parallel algorithm, in which the individual processing elements need only perform a handful of computations.

3 3D Motion and Optical Flow

We first review the physics and geometry of instantaneous rigid-body motion under perspective projection, and derive an equation relating 3D motion to optical flow. Although this equation has been derived previously by a number of authors (e.g., Longuet-Higgins and Prazdny [1980], Bruss and Horn [1983], Waxman and Ullman [1985]), we write it in a new form that reveals its underlying simplicity.

Each point in a scene has an associated position vector, $\mathbf{X} = (X, Y, Z)^t$, relative to a viewer-centered coordinate frame as depicted in figure 2. Under perspective projection this surface point projects to a point in the image plane, $(x, y)^t$

$$x = fX/Z$$
$$y = fY/Z \qquad (1)$$

where f is the "focal length" of the projection.

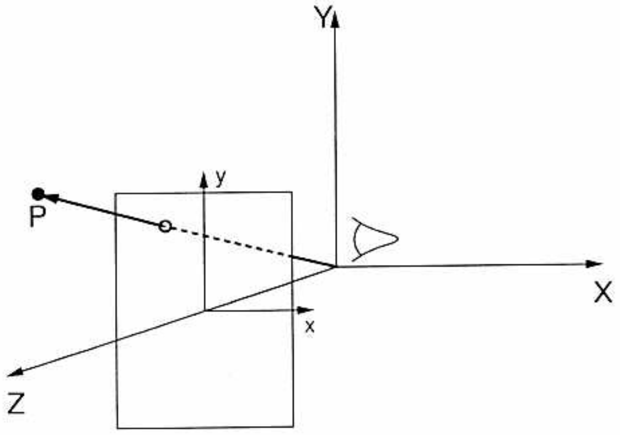


Fig. 2. Viewer centered coordinate frame and perspective projection.

Every point of a rigid body shares the same six motion parameters relative to the viewer-centered coordinate frame. Due to the motion of the observer, the relative motion of a surface point is

$$\left(\frac{dX}{dt}, \frac{dY}{dt}, \frac{dZ}{dt}\right)^t = -(\mathbf{\Omega} \times \mathbf{X} + \mathbf{T}) \qquad (2)$$

where $\mathbf{T} = (T_x, T_y, T_z)^t$ and $\mathbf{\Omega} = (\Omega_x, \Omega_y, \Omega_z)^t$ denote, respectively, the translation and rotation.

Image velocity, $\mathbf{v}(x, y)$, is defined as the derivatives, with respect to time, of the x- and y-components of the projection of a scene point. Taking derivatives of equation (1) with respect to time, and substituting from equation (2) gives

$$\mathbf{v}(x, y) = p(x, y)\mathbf{A}(x, y)\mathbf{T} + \mathbf{B}(x, y)\boldsymbol{\Omega} \qquad (3)$$

where $p(x, y) = 1/Z$ is inverse depth, and where

$$\mathbf{A}(x, y) = \begin{bmatrix} -f & 0 & x \\ 0 & -f & y \end{bmatrix}$$

$$\mathbf{B}(x, y) = \begin{bmatrix} (xy)/f & -(f + x^2/f) & y \\ f + y^2/f & -(xy)/f & -x \end{bmatrix}$$

The $\mathbf{A}(x, y)$ and $\mathbf{B}(x, y)$ matrixes depend only on the image position and the focal length, not on any of the unknowns.

Equation (3) describes the flow field as a function of 3D motion and depth. An important observation about equation (3) is that it is bilinear; $\mathbf{v}$ is a linear function of $\mathbf{T}$ and $\boldsymbol{\Omega}$ for fixed p, and it is a linear function of p and $\boldsymbol{\Omega}$ for fixed $\mathbf{T}$.

Since both $p(x, y)$ (the inverse depth) and $\mathbf{T}$ (the translation) are unknowns and since they are multiplied together in equation (3), they can each be determined only up to a scale factor; that is, we can solve for only the direction of translation and the relative depth, not for the absolute translation nor for the absolute depth. For the rest of this article, $\mathbf{T}$ denotes a unit vector in the translation direction (note that $\mathbf{T}$ now has only two degrees of freedom), and $p(x, y)$ denotes the relative inverse depth ($\|\mathbf{T}\|/Z$).

It is impossible to recover the 3D motion parameters, given the image velocity at only a single image location; there are six unknowns on the right-hand side of equation (3) and only two measurements (the two components of $\mathbf{v}$) on the left-hand side. Image velocity measurements at five image locations are necessary, although not always sufficient, to solve the problem [Prazdny 1983].

In the next section, we describe how our algorithm uses a number of flow vectors to solve for 3D translation. For each of N sample points, a separate equation can be written in the form of equation (3). These N equations can also be collected together into one matrix equation (reusing the symbols in equation (3) rather than introducing new notation):

$$\begin{aligned} \mathbf{v} &= \mathbf{A}(\mathbf{T})\mathbf{p} + \mathbf{B}\boldsymbol{\Omega} \\ &= \mathbf{C}(\mathbf{T})\mathbf{q} \qquad (4) \end{aligned}$$

where $\mathbf{v}$ is now a $2N$ vector containing the image velocity at each of the N sample points, and $\mathbf{p}$ is now an N vector containing the inverse depth at each sample point. $\mathbf{A}(\mathbf{T})$ (a $2N$-by-N matrix) is obtained by collecting together into a single matrix $\mathbf{A}(x, y)\mathbf{T}$ for each sample point. Each column of $\mathbf{A}(\mathbf{T})$ contains two nonzero entries (the two elements of $\mathbf{A}(x, y)\mathbf{T}$):

$$\mathbf{A}(\mathbf{T}) = \begin{bmatrix} \mathbf{A}(x_1, y_1)\mathbf{T} & \dots & 0 \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ 0 & \dots & \mathbf{A}(x_N, y_N)\mathbf{T} \end{bmatrix}$$

Similarly, $\mathbf{B}$ (a $2N$-by-3 matrix) is obtained by collecting together into a single matrix the $\mathbf{B}(x, y)$ matrixes:

$$\mathbf{B} = \begin{bmatrix} \mathbf{B}(x_1, y_1) \\ \cdot \\ \cdot \\ \cdot \\ \mathbf{B}(x_N, y_N) \end{bmatrix}$$

Finally, $\mathbf{q}$ (an N+3 vector) is obtained by collecting together into one vector the unknown rotation and depth values, and $\mathbf{C}(\mathbf{T})$ is obtained by placing the columns of $\mathbf{B}$ along side the columns of $\mathbf{A}(\mathbf{T})$:

$$\mathbf{C}(\mathbf{T}) = \begin{bmatrix} | & | \\ \mathbf{A}(\mathbf{T}) & \mathbf{B} \\ | & | \end{bmatrix}$$

4 Recovering the Direction of Translation

In this section, we present our method for recovering 3D translation, $\mathbf{T}$. The rotation and depth values need not be known or estimated prior to solving for translation.

The space of all candidate translation directions is the unit sphere. We need only consider half of the sphere since solutions on the front and back halves are the same, as can be seen by multiplying both $p(x, y)$ and $\mathbf{T}$ by minus one, in equation (3).

We define a residual function, $E(\mathbf{T})$, over the discretely sampled space of all candidate translation directions. Each candidate translation is used, along with flow field measurements, to compute a residual value (or badness rating). The residual value indicates whether or not the candidate translation is consistent with the flow vectors. A residual value of zero for a particular candidate translation indicates that the candidate translation is a plausible interpretation of the flow vectors.

Figure 3 is an illustration of the residual function. Each point in figure 3 corresponds to a different translation direction, and the brightness at each point corresponds to a residual value. The minimum in this residual function corresponds to the solution.

The residual values are computed in parallel for each candidate translation, and in parallel for different patches of the flow field. The resulting residual functions are then summed, as illustrated in figure 4, giving a global least-squares estimate for translation.

The solution space is small (the unit hemisphere is a compact two-dimensional space), so the residual function can be evaluated using a practical amount of memory and compute time.

4.1 The Residual Function

The residual function, $E(\mathbf{T})$, is defined to yield a least-squares estimate for translation, such that (in the absence of noise) $E(\mathbf{T}_0)$ is minimized for $\mathbf{T}_0$ equal to the actual translation. Equation (4) relates the image velocities at N sample points (concatenated in $\mathbf{v}$) to the product of a matrix, $\mathbf{C}(\mathbf{T})$ (that depends on the unknown 3D translation), times a vector, $\mathbf{q}$, (the unknown rotation and depth values). The least-squares estimate for translation minimizes the following expression over all candidate translations, rotations, and depth values:

$$E(\mathbf{T}, \mathbf{q}) = \|\mathbf{v} - \mathbf{C}(\mathbf{T})\mathbf{q}\|^2 \tag{5}$$

We seek the choice of $\mathbf{T}$ that minimizes this expression over all candidate $\mathbf{T}$ and $\mathbf{q}$.

We show in appendix A that minimizing expression (5) over all $\mathbf{T}$ and $\mathbf{q}$ is equivalent to minimizing the following residual function over $\mathbf{T}$ alone:

$$E(\mathbf{T}) = \|\mathbf{v}^t\mathbf{C}^{\perp}(\mathbf{T})\|^2 \tag{6}$$

where $\mathbf{C}^{\perp}(\mathbf{T})$ is the orthogonal complement to $\mathbf{C}(\mathbf{T})$, that is easily computed [Strang 1980] from the elements of $\mathbf{C}(\mathbf{T})$.

The matrices $\mathbf{C}(\mathbf{T})$ and $\mathbf{C}^{\perp}(\mathbf{T})$ depend only on the locations of the N sample points and on the candidate $\mathbf{T}$. These matrices do not depend on the flow-field measurements. Therefore, the $\mathbf{C}^{\perp}(\mathbf{T})$ matrices may be precomputed and stored for each image patch, and for each candidate translation.

As new flow-field measurements become available from incoming images the residual values are computed in two steps: (1) the linear summations given by $\mathbf{v}^t\mathbf{C}^{\perp}(\mathbf{T})$, and (2) the sum of the squares of the resulting numbers.

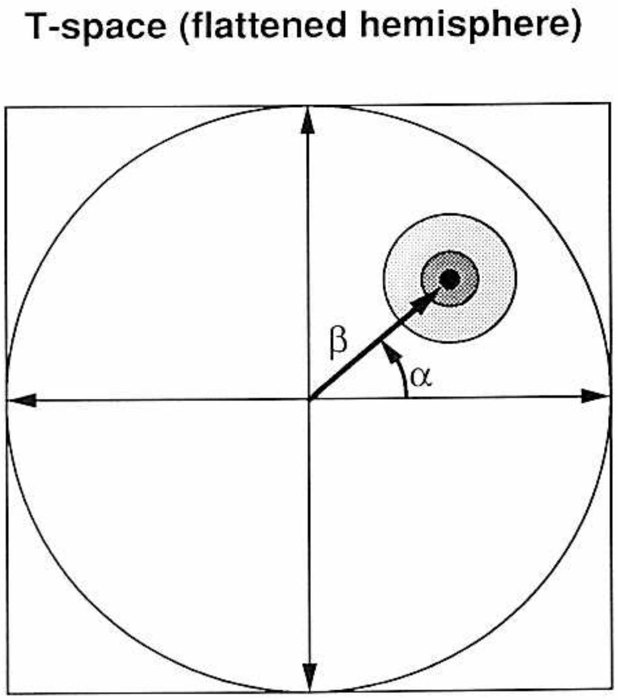


Fig. 3. The space of all candidate translation directions is made up of the points on the unit hemisphere. The residual function, $E(\mathbf{T})$, is zero for the true direction of translation. The compact two-dimensional solution space is parameterized by α and β, the angles in spherical coordinates that specify each point on the unit hemisphere.

4.2 Uniqueness and Robustness

In our companion article [Jepson and Heeger 1990] we prove that our subspace method gives unique and robust solutions. Here, we briefly summarize these theoretical results on uniqueness and robustness.

First, it is important to know if there are incorrect translational velocities that might also have a residual of zero. When the inverse depth values (for a given image patch) are sufficiently nonplanar, it can be shown that under specific conditions, the zeroes of $E(\mathbf{T})$ are concentrated near the great circle that passes through the correct translation direction ($\mathbf{T} = \mathbf{T}_0$) and through the direction that points toward the center of the patch [Jepson and Heeger 1990] (see also Maybank [1987]).

The solution is disambiguated by summing residual surfaces from different image patches. Two or more patches, in significantly different visual directions, have

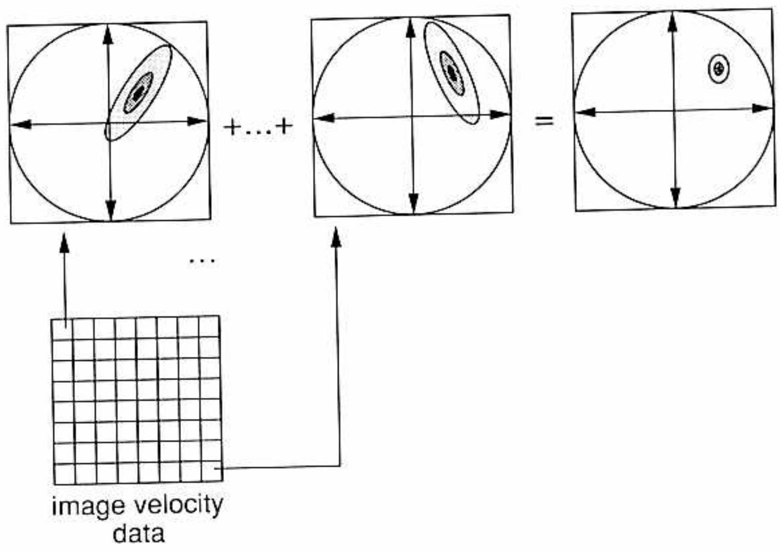


Fig. 4. The direction of translation is recovered by subdividing the flow field into patches. A residual surface is computed for each patch using flow field measurements at several sample points from within that patch. The residual surfaces from each patch are then summed to give a single solution.

zeroes concentrated near different great circles. They have simultaneous zeroes only near the intersection of the great circles, namely near $\mathbf{T} = \mathbf{T}_0$. In cases where the depth is nearly planar, or where velocity values are available only within a narrow visual angle, it may be impossible to obtain a unique solution. These cases will be the exception rather than the rule in natural scenes.

Second, it is important to know how finely we need to sample the solution space of candidate translations. In our companion article, we show that the entire residual surface can be approximately reconstructed from only a few samples (see also Maybank [1987]).

Third, we show in the companion article that the Hessian of the residual surface is large at the point of smallest residual, so this point will be stable with respect to errors in the image velocity measurements.

5 Recovering Rotation and Depth

Once translation has been determined, one can solve for rotation as well. We now proceed to eliminate the depth, $p(x, y)$, from equation (3), leaving us with a linear constraint on rotation, $\mathbf{\Omega}$. To this end, we define $\mathbf{d}(x, y, \mathbf{T})$ to be a unit vector perpendicular to the translational component of the flow field,

$$\mathbf{d}^t(x,y,\mathbf{T})\mathbf{A}(x, y)\mathbf{T} = 0$$
$$\|\mathbf{d}(x, y, \mathbf{T})\| = 1$$

Multiplying by $\mathbf{d}^t(x, y, \mathbf{T})$ on both sides of equation (3) eliminates the dependence on $p(x, y)$:

$$\mathbf{d}^t(x, y, \mathbf{T})\mathbf{v}(x, y) = \mathbf{d}^t(x, y, \mathbf{T})\mathbf{B}(x, y)\mathbf{\Omega} \quad (7)$$

Equation (7) is a linear constraint on the rotation, $\mathbf{\Omega}$, given the translation, $\mathbf{T}$, and a velocity vector, $\mathbf{v}(x, y)$. However, it is impossible to solve for the rotation given just one velocity vector, since $\mathbf{\Omega}$ has three unknown components and equation (7) provides only one constraint.

Several flow vectors may be utilized in concert to solve for the rotation. If there is no error in the input flow field then three flow vectors are sufficient. We compute a least-squares estimate for rotation, using a large number of flow vectors. For each flow vector, we write an equation in the form of equation (7) (dropping the arguments x, y, and $\mathbf{T}$, for simplicity):

$$\mathbf{d}_i^t\mathbf{v}_i = \mathbf{d}_i^t\mathbf{B}_i\mathbf{\Omega}$$

where i now indexes over the flow vectors at different sample points. The least-squares solution for $\mathbf{\Omega}$ is obtained by minimizing:

$$E(\mathbf{\Omega}) = \sum_i \|\mathbf{d}_i^t\mathbf{B}_i\mathbf{\Omega} - \mathbf{d}_i^t\mathbf{v}_i\|^2$$

The estimate is given by

$$\hat{\mathbf{\Omega}} = \left(\sum_i \mathbf{B}_i^t\mathbf{d}_i\mathbf{d}_i^t\mathbf{B}_i\right)^{-1}\left(\sum_i \mathbf{B}_i^t\mathbf{d}_i\mathbf{d}_i^t\mathbf{v}_i\right) \quad (8)$$

The first factor on the right-hand side of equation (8) is a 3×3 matrix that does not depend on the input flow field, but it does depend on the recovered translation. The second factor on the right-hand side of equation (8) is a linear sum of the input flow vectors. The coefficients in this linear sum depend on the recovered translation. As new flow-field measurements (and new estimates of translation) become available from incoming images, rotation is estimated as a linear combination of the flow vectors. The recovered translation triggers which set of weights to use in this linear combination.

An alternative way of formulating the least-squares estimate for rotation is to start with equation (4), which relates 3D motion and depth to flow at N sample points. Solving this large system of linear equations for rotation can be proved equivalent to the solution given by equation (8). The latter has the advantage of being a highly parallel algorithm in which $\mathbf{d}(x, y, \mathbf{T})$ is computed independently for each image location.

Finally, once the translation and rotation are both known, equation (3) provides two linear constraints to solve for the unknown depth at each image point. The translation and rotation estimates are reasonably insensitive to random errors in the flow-field measurements (see results below) because we combine those inputs over the entire field of view. The depth estimates, on the other hand, are quite sensitive to input errors.

We can use information over an extended period of time to improve the depth estimates. Bolles et al. [1987] recovered depth, given the known camera motion, by tracking features through the space-time volume of image data. Matthies et al. [1989] used sequential estimation (Kalman filtering) to recover depth from sequences of flow fields, given the known camera motion. Taken together, their results demonstrate that it is practical to recover depth once the camera motion is known.

6 Results

We implemented the algorithms described in sections 4 and 5 on the connection machine. Due to the simple nature of the computations it would be straightforward to design parallel hardware capable of performing these operations in real time. The limiting factor in building such hardware is the memory requirement of storing the $\mathbf{C}^{\perp}(\mathbf{T})$ coefficients for each candidate $\mathbf{T}$.

The current implementation assigns each processor to a particular candidate translation, and to a particular flow-field patch. The number of patches of the flow field, the amount of overlap of the patches, the number of flow-field samples in each patch, and the spacing of the samples within each patch were chosen arbitrarily. For the results presented below, unless otherwise stated, we used nonoverlapping, five-point, dice-shaped sample patterns. Proper sampling of the solution space and proper sampling of the flow field are interesting issues for further research.

Note that there are a wide variety of alternative implementations of our proposed approach. We could use gradient descent (or some other minimization scheme) rather than sampling the solution space. We could also use multiple scales, sampling both the flow field and the solution space at a variety of different resolutions.

6.1 Comparison with E-matrix

This section reports simulation results that compare the performance of our 3D motion algorithm with the linear discrete-time *E*-matrix approach [Longuet-Higgins 1981; Tsai and Huang 1984; Weng et al. 1989]. Our implementation of the *E*-matrix algorithm is exactly as described by Weng et al.

The *E*-matrix approach is formulated as follows. Let $\mathbf{X} = (X, Y, Z)^t$ be the position of a surface point relative to the viewer-centered coordinate system depicted in figure 2, and let $\mathbf{X}' = (X', Y', Z')^t$ be the position of that point some time later. The rigid motion constraint relates the two positions:

$$\mathbf{X}' = \mathbf{R}\mathbf{X} + \mathbf{T} \tag{9}$$

where $\mathbf{R}$ is a rotation matrix and $\mathbf{T}$ is a translation vector. The basic constraint of the *E*-matrix approach is

$$(\mathbf{X}')^t[\mathbf{T} \times (\mathbf{R}\mathbf{X})] = 0$$

This constraint is easily derived by observing that the vectors $(\mathbf{R}\mathbf{X})$, $\mathbf{T}$, and $\mathbf{X}'$ all lie in the same plane. This constraint is usually expressed as

$$(\mathbf{X}')^t\mathbf{E}\mathbf{X} = 0 \tag{10}$$

where $\mathbf{E}$ (the matrix of "essential parameters") is the appropriate combination of $\mathbf{R}$ and $\mathbf{T}$.

The two constraints written above, equations (9) and (10), are not equivalent. A nonrigid configuration of points can still satisfy the *E*-matrix constraint. For example, consider a rigid configuration of points, $(\mathbf{X}_1, \ldots, \mathbf{X}_n)$ that move to $(\mathbf{X}_1', \ldots, \mathbf{X}_n')$. Replace $\mathbf{X}_1'$ by any other point that lies in the plane normal to $[\mathbf{T} \times (\mathbf{R}\mathbf{X}_1)]$. There is no longer any rigid-body motion that transforms the points $\mathbf{X}_i$ to the points $\mathbf{X}_i'$, even though the *E*-matrix constraint is still satisfied.

The *E*-matrix algorithms work by solving first for the elements of the *E*-matrix, given the positions of image features in two frames. Then the 3D translation is computed from **E**. The advantage of the *E*-matrix algorithms is that they are simple and inexpensive to compute, since the *E*-matrix constraint is linear. However, the consequence of using a weaker constraint is that these algorithms are particularly sensitive to errors in the input.

Figure 5 shows the comparison between our algorithm and the *E*-matrix algorithm. A flow field was synthesized from a random-depth map, given a particular 3D motion. Uniform noise of various amounts was added to each component of each velocity vector. The field of view was chosen to be relatively large in these simulations, since this improves the performance of the *E*-matrix algorithm. Both algorithms were given exactly the same input flow fields. The results from our algorithm were obtained by subdividing the flow field into nonoverlapping 3×3 patches, and summing the residual surfaces from each patch.

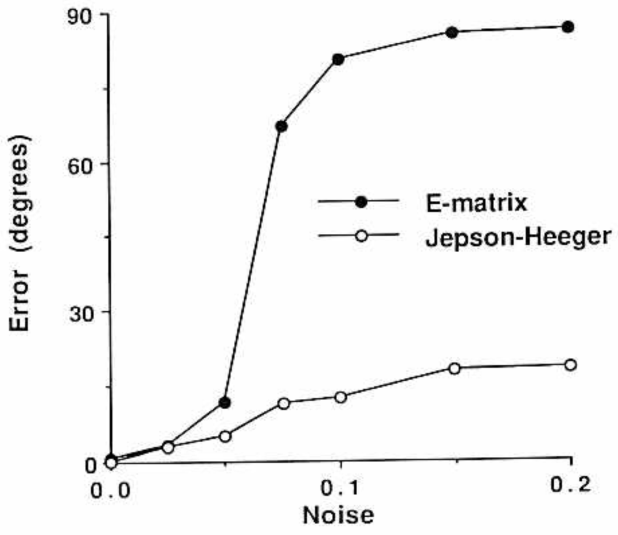


Fig. 5. Comparison of our algorithm with the *E*-matrix algorithm, using noisy synthetic flow fields. Graph plots error in the estimate of translation direction as a function of the noise level, averaged over 5 trials at each noise level. Noise level of 0.1 means that the width of the uniform distribution was proportional to one-tenth of the average speed (averaged over the entire flow field). It is clear that our algorithm performs much better than the *E*-matrix algorithm. This result is not surprising since the *E*-matrix constraint is weaker than the rigid-motion constraint. Synthetic flow fields were generated using a random-depth map. Depth varied randomly between 100 and 200 (in units of pixels). Focal length was 9 (in units of pixels). Entire image was 9×9, that subtended 53 degrees of visual angle. Translation was in the *X* direction, $\mathbf{T} = (1, 0, 0)$, and speed was 1 (in units of pixels). Rotation was about the *Y* axis at 0.6 degrees/frame.

For the *E*-matrix algorithm, relatively low noise levels result in extremely large errors in the estimate of translation direction. The absolute scale on the axes of the graph in figure 5 is not critical. For a set noise level we can reduce the error simply by using more input flow vectors. It is the relative difference between the performance of the two algorithms that is important.

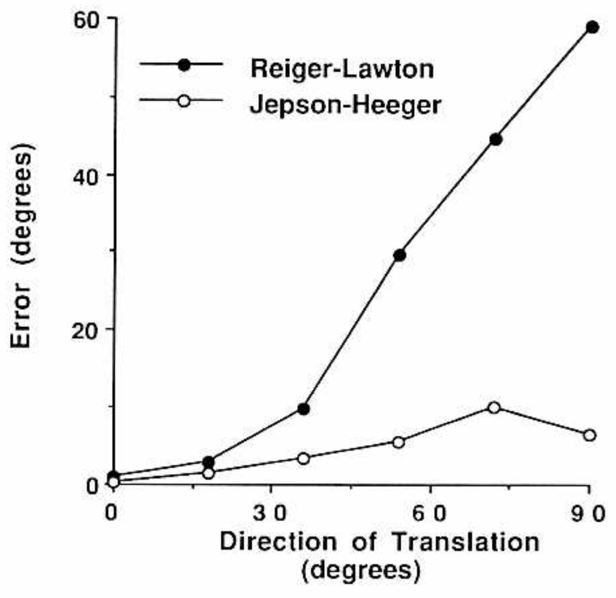
6.2 Comparison with Reiger-Lawton

This section reports simulation results that compare the performance of our 3D motion algorithm with the algorithm of Reiger and Lawton [1985]. For each flow vector, the Reiger-Lawton algorithm computes the difference vectors with neighboring flow vectors. If the neighborhood is small enough, the difference vectors all point approximately toward the FOE. Our implementation of the Reiger-Lawton algorithm is described in appendix C.

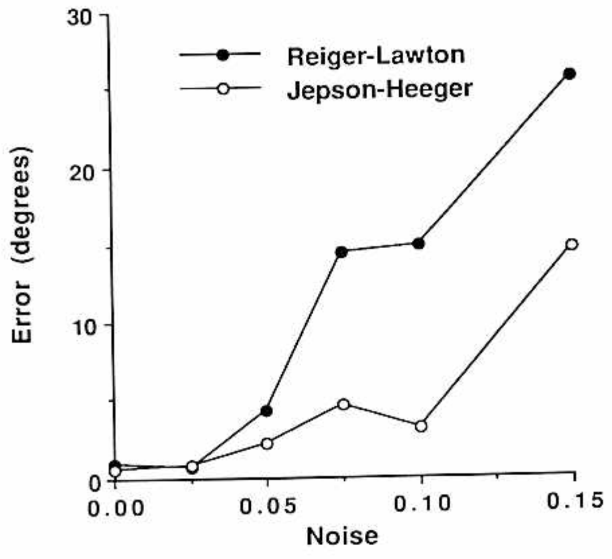
Figure 6 shows the comparison. Flow fields were synthesized from a random-depth map, given various translation directions. Uniform noise of various amounts was added to each component of each velocity vector. The results from our algorithm were obtained by subdividing the flow fields into patches, as described above. The patches also served as the neighborhoods for the Reiger-Lawton algorithm. The size of the neighborhoods was also varied. An attempt was made to provide the Reiger-Lawton algorithm with a most favorable input: there was significant depth variation from point to point in the random-depth map, the rotational motion was always zero, and the neighborhood sizes were quite small.

Figure 6a shows the comparison for a fixed noise level, while varying the direction of translation. For the straight-ahead direction, both algorithms perform equally well. For translation directions away from straight ahead, our algorithm performs much better than the Reiger-Lawton algorithm.

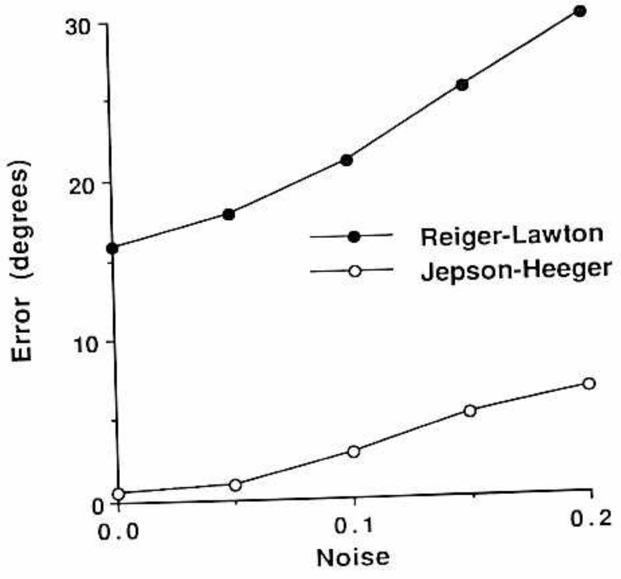
We are not certain why the Reiger-Lawton algorithm performs worse for sideways motion. The procedure that Reiger and Lawton [1985] originally described computes the FOE from a set of difference vectors. That procedure fails for sideways motion because the FOE is at infinity. However, this is not a problem for our implementation of the Reiger-Lawton algorithm; we avoided this potential problem by solving for the trans-



(a)



(b)



(c)

Fig. 6. Comparison of our algorithm with the Reiger-Lawton algorithm, using noisy synthetic flow fields. Graphs plot error in the estimate of the translation direction as a function of: (a) the actual translation direction; (b) and (c) the noise level. Neighborhood size used for (c) was about 3 times larger than that used for (b). Each data point is an average over 5 trials. Noise level of 0.1 means that the width of the uniform distribution was proportional to one-tenth of the average speed (averaged over the entire flow field). It is clear that our algorithm performs better than the Reiger-Lawton algorithm, especially for the larger neighborhood size. This result is not surprising since the Reiger-Lawton algorithm is based on an approximation that is valid only when the neighborhood is small. Synthetic flow fields were generated using a random-depth map. Depth varied randomly between 100 and 200 (in unit of pixels). Focal length was 50 (in units of pixels). Entire image was 28×28 pixels, that subtended 31 degrees of visual angle. The translation direction varied from straight ahead, $\mathbf{T} = (0, 0, 1)$, to sideways, $\mathbf{T} = (1, 0, 0)$, and speed was 1 (in units of pixels). There was no rotation. Neighborhood size in (a) and (b) was 3×3 pixels; neighborhood at the center of the image subtended 3.5 degrees of visual angle. Neighborhood size in (c) was 9×9 pixels; neighborhood at the center of the image subtended 10 degrees of visual angle. Noise level in (a) fixed at 0.1. Translation direction in (b) and (c) fixed at $\mathbf{T} = (0.6, 0, 0.8)$.

lation direction, rather than solving for the FOE (see appendix C). Even so, our implementation of the Reiger-Lawton algorithm still performs worse than our new algorithm. One likely explanation is the approximation of treating all the velocity samples as if they came from the same image position. It seems plausible that the algorithm is more sensitive to input errors as the angle increases between the translation direction and the optical axis.

Figures 6b and 6c show the comparison for a fixed translation direction while varying the noise level. The neighborhood size used for 6c was 3 times larger than that used for 6b. In 6b, both algorithms perform equally well at low noise levels. As the noise level increases, our algorithm performs somewhat better. When the neighborhood size is increased in 6c, our algorithm improves noticeably whereas the Reiger-Lawton algorithm gets much worse. This result is not surprising since the Reiger-Lawton algorithm is based on an approximation that is valid only when the neighborhood size is small.

Again, the absolute scales on the axes in these figures is not critical. It is the relative difference between the performance of the two algorithms that is important.

6.3 Zero Translation

One concern about 3D motion algorithms is that they behave in a reasonable way when there is little or no translational component in the flow field. This can happen for slow 3D translations, or it can happen when the entire scene is very distant. In either case, the rotational component of the flow field will be large compared to the translational component.

Simulations using our algorithm on noiseless flow fields show that when the translation is zero, the residual surface is everywhere nearly zero (there is no well-defined minimum in the residual surface). For noisy flow fields, there are typically several local minima in the residual surface.

Simulations also show that the rotational velocity is recovered robustly when the actual translation is zero. We can use any choice of **T** as input when the actual translation is very small. The choice of **T** has little bearing on the estimate for $\mathbf{\Omega}$, even for relatively noisy flow fields.

Once the rotational velocity has been recovered, it is a simple matter to check whether or not there was any translation. Test whether a pure rotation can account for the flow field, and if so, ignore the translation and depth estimates.

6.4 Planar Surface

It is well known that a moving planar surface gives rise to an ambiguous flow field [Hay 1966; Adiv 1989; Longuet-Higgins 1984, 1988; Horn 1987; Maybank 1985; Waxman et al. 1987]. Two different planes with different motions can each result in the same flow field. The slopes of the two planes and the components of motion parallel to the image plane play interchangeable roles in the two solutions. If the slope of one plane is denoted (m_x, m_y) and the translation is (T_x, T_y, T_z), then the structure and motion of the second plane is given by

$$\begin{aligned} T'_x &= -m_x T_z \qquad & m'_x &= -T_x/T_z \\ T'_y &= -m_y T_x \qquad & m'_y &= -T_y/T_z \\ T'_z &= T_z \end{aligned}$$

Figure 7 shows residual surfaces for two different planar surfaces. The two peaks in each residual surface correspond to the two possible interpretations of the flow field. The ambiguity is easily resolved since only one of the solutions will give consistent interpretations over time [Longuet-Higgins 1984; Waxman et al. 1987].

6.5 Instantaneous Approximation

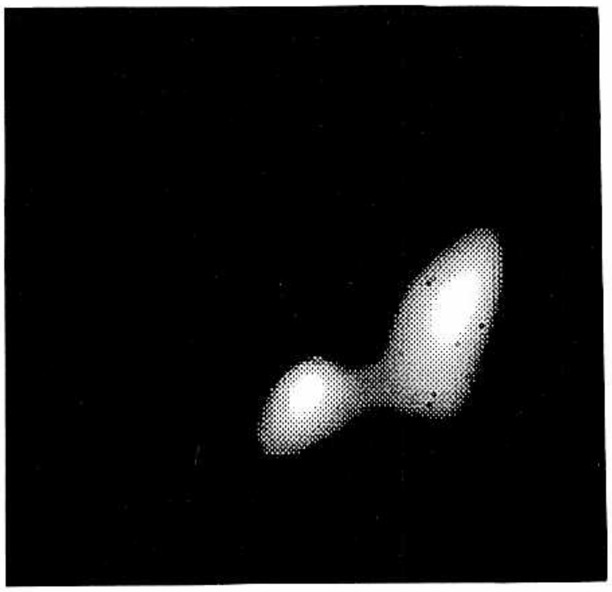
In section 3 we derive the relationship between instantaneous 3D motion and optical flow. Image velocity is defined in equation (3) as the derivatives, with respect to time, of the x- and y-components of the projection of a surface point.

For the discrete-time formulation, equation (9) gives the relationship between a surface point's position before and after a motion, and equation (1) expresses how the surface point projects onto the image. Image displacement is the difference between the projected positions of the surface point.

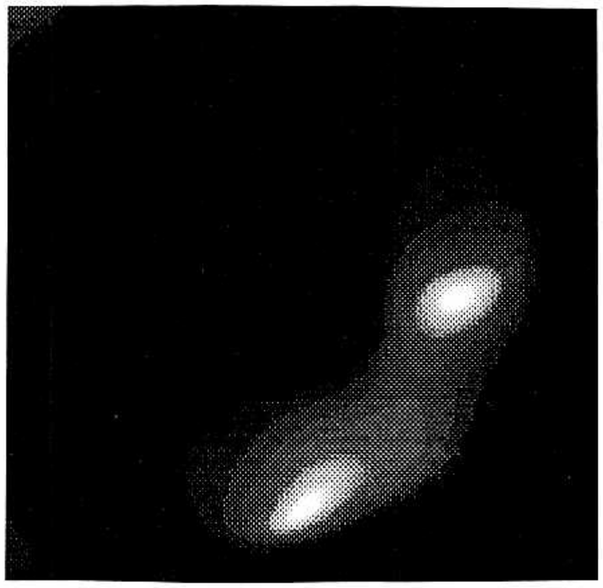
The instantaneous formulation is a valid approximation to the discrete-time formulation for short time steps. The rotation rate must be small enough so that the rotation matrix in equation (9) can be approximated using rotational velocities:

$$\mathbf{R} \approx \begin{pmatrix} 1 & -\Omega_z\,\Delta t & \Omega_y\,\Delta t \\ \Omega_z\,\Delta t & 1 & -\Omega_x\,\Delta t \\ -\Omega_y\,\Delta t & \Omega_x\,\Delta t & 1 \end{pmatrix}$$

where $\mathbf{R}$ is a rotation matrix, $\mathbf{\Omega} = (\Omega_x, \Omega_y, \Omega_z)^t$ is rotational velocity, and Δt is the length of the time step.



(a)



(b)

Fig. 7. Residuals for moving planar surfaces. Brightness is proportional to $-\log\,[E(\mathbf{T})]$. The two peaks correspond to the two possible solutions. (a) Slope of the plane is (0, 1/2). (b) Slope of the plane is (0, 2). In both cases the actual translation was (1, 0, 1) and there was no rotation. Focal length was 50 (in units of pixels). Entire image was 28×28 pixels, that subtended 31 degrees of visual angle.

Adiv [1985] points out that the instantaneous approximation is justified when this and two additional conditions are met. First, the component of translation along the line of sight must be small relative to the depths, that is,

$$T_z\,\Delta t \ll Z$$

and second, the x- and y-components of rotation must be small relative to the imaging geometry, that is,

$$\forall y\ |y\Omega_x\,\Delta t| \ll f$$

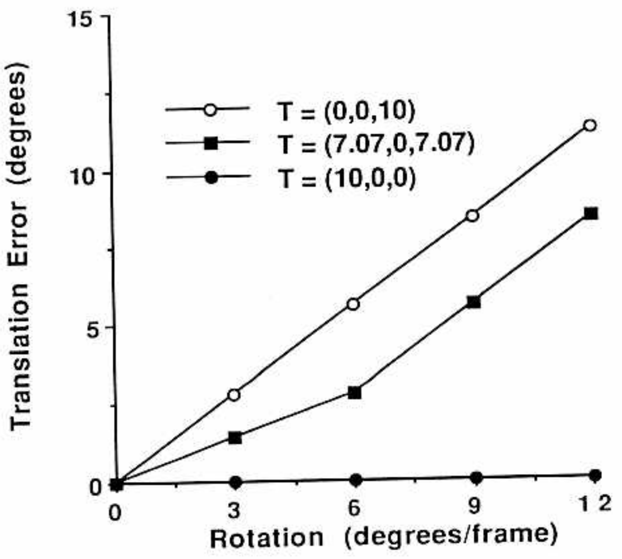
$$\forall x\ |x\Omega_y\,\Delta t| \ll f$$

where f is focal length and (x, y) is image position.

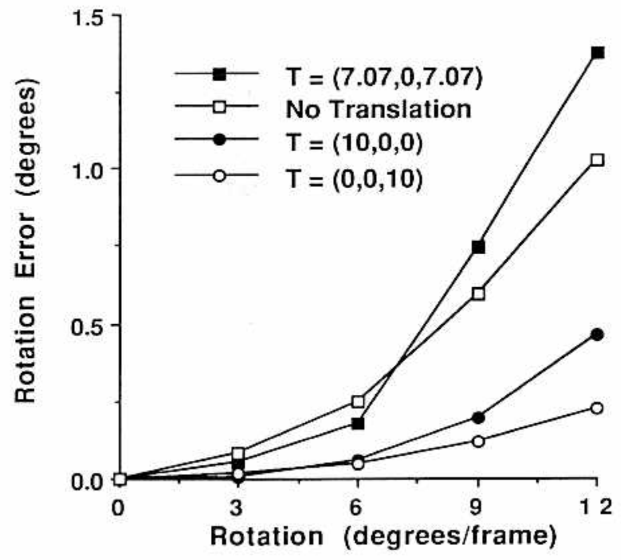
Figure 8 shows how our algorithm behaves when the instantaneous approximation breaks down. Flow fields were synthesized using the discrete-time formulation, equations (9) and (1), given various translations and rotations. Results are shown for translation in the X-Z plane, and for rotation about the Y-axis. These results are typical of results for a variety of translation directions, translation speeds, and rotation axes. The errors introduced by the instantaneous approximation are quite small if the rotation rate is less than 3 degrees per frame (90 degrees per second at video rates); the error in translation direction is less than 3 degrees and the error in the rotation estimate is insignificant. At 6 degrees per frame (180 degrees per second) the error in translation is still less than 6 degrees and the error in rotation rate is less than 10%.

6.6 Realistic Scene

The simulations reported above were done with random and planar depth maps. Figure 9 shows simulation results using a more realistic scene. Figure 9a shows a computer-graphics generated image. Flow fields were computed from the actual, known depth map, given particular translations and rotations. Figure 9b shows a flow field corresponding to pure translatory motion. Figure 9c shows a flow field corresponding to translation plus rotation. By comparing 9b and 9c it is evident that the rotational component in 9c is quite significant. The flow fields were quantized to 8 bits of precision. Depth and motion were computed from 9c, the translation plus rotation case. Figures 9d and 9e show, respectively, the residual surface and the recovered depth map. Translation direction was recovered to within a tenth of a degree. The slight error is due to the quantization of the flow field and quantization of the solution space.



(a)



(b)

Fig. 8. Error introduced by the instantaneous approximation. Flow fields synthesized using the discrete-time formulation and estimates recovered using our instantaneous-time algorithm. Results are shown for 3 different translation directions. Speed of translation was either 10 (in units of pixels) or zero (for the no-translation case). (a) Error in the estimate of translation direction as a function of the actual rotation rate. (b) Error in the estimate of rotation rate as a function of the actual rotation rate. Errors introduced by the instantaneous-time approximation are quite small if the rotation rate is less than 3 degrees per frame (90 degrees per second at video rates). Synthetic flow fields were generated using a random-depth map. Depth from point to point varied randomly from 100 to 200 (in units of pixels). Focal length was 50 (in units of pixels). Entire image was 28×28 pixels, that subtended 31 degrees of visual angle.

6.7 Realistic Image Sequence

All of the above results were computed from synthetic flow fields. Figure 10 shows a result computed from an image sequence. Figure 10a is a frame from a computer graphics generated image sequence of a flight through Yosemite valley.

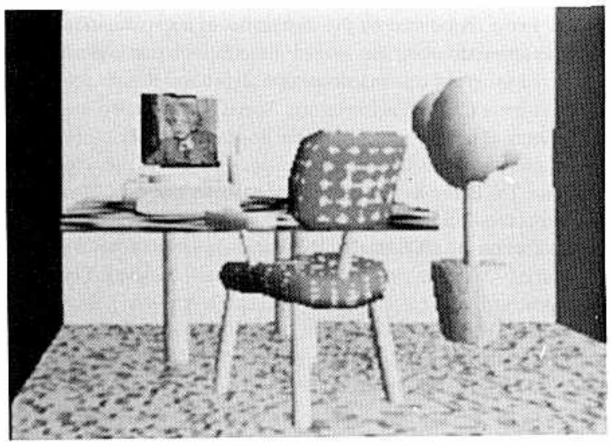
A sequence of flow fields were measured from the image sequence using the spatiotemporal filtering algorithm described by Simoncelli et al. [1991]. Figure 10b shows the correct flow field computed from the known motion and depth values. Figure 10c shows an example of one of the measured flow fields, and figure 10d shows both 10b and 10c superimposed. Errors near the edges are due to edge effects of the spatiotemporal convolutions.

The measured flow fields were used as input to compute a sequence of residual surfaces. Since the motion was constant over the entire 15-frame sequence the residual surfaces were summed to give a single residual surface, shown in figure 10e. The actual translation direction was $\mathbf{T} = (0, 0.17, 0.98)$, and the actual speed of motion was 34.8 (in units of pixels). The recovered translation direction was (0.06, 0.18, 0.98), in error by 3.5 degrees.

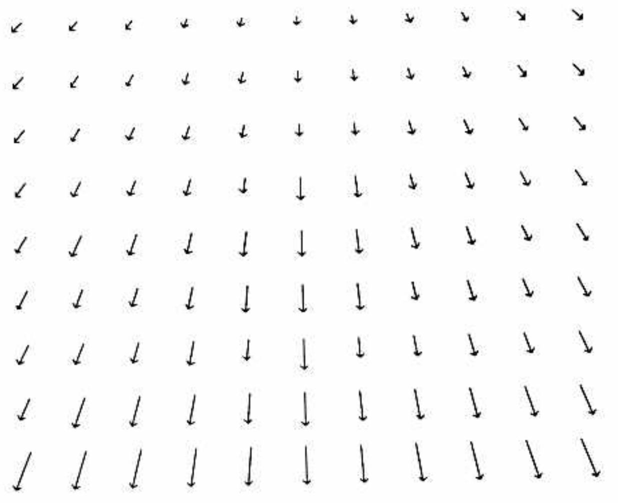
Similar results were obtained averaging only a few (e.g., three consecutive) residual surfaces at a time. If the 3.5-degree error were due to random errors in the flow measurements, then using more flow fields would help. The 3.5-degree error is, therefore, due to a systematic bias in the flow field measurements.

The recovered translation was then used to estimate rotation. The actual rotation axis was (0.14, 0.98, 0.17), and the actual rotation rate was .095 degrees/frame. The estimates of rotation from different frames were all about the same (less than 10% variation from one frame to the next). The average estimated rotation axis was $(-0.29, 0.85, -0.43)$, and the average estimated rotation rate was .15 degrees/frame.

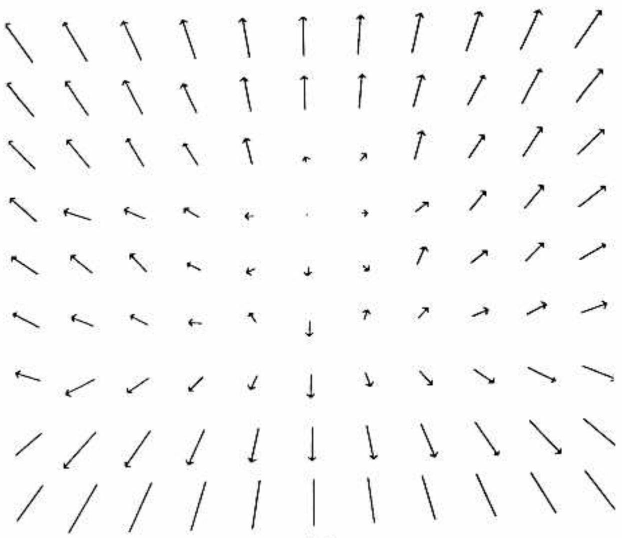
Depth values were estimated using the recovered translation and rotation. Figure 10f shows the relative percentage error in the depth estimates. Large errors



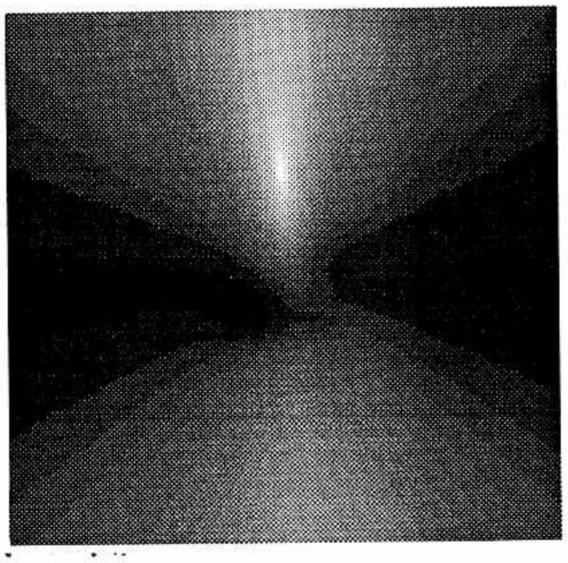
(a)



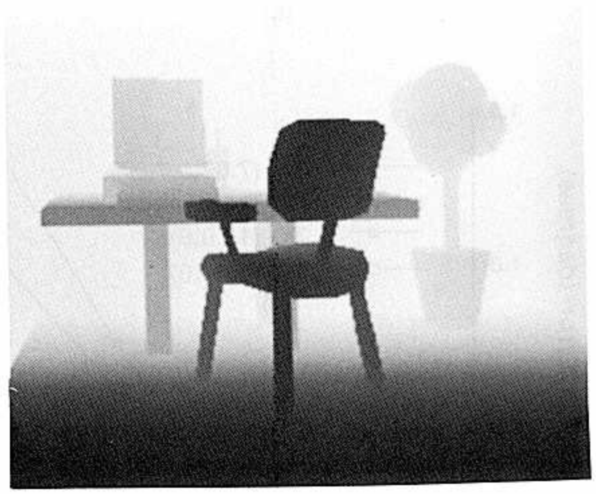
(b)



(c)



(d)



(e)

Fig. 9. (a) Computer-graphics generated image, simulating a 35-mm camera (20-degree field of view). (b) Flow field computed from actual known depth map, for pure translatory motion. (c) Flow field for translation plus rotation. (d) Residual surface computed from flow field in (c). Brightness is proportional to $-\log [E(\mathbf{T})]$. Peak location corresponds to the estimate of translation direction. (e) Recovered depth map, brightness is proportional to distance. Since the proper (8-bit precision) flow field was used as input there is essentially no error in these results. Translation direction was $\mathbf{T} = (0, 0.3, 0.95)$, and speed was 9.5 (in units of pixels). Rotation was about the X axis at 0.3 degrees/frame. Depth values ranged from 340 to 825 (in units of pixels).

along the boundaries are due to edge effects of measuring the flow fields. Large errors in the background arise because small errors in the flow measurements can result in large depth errors near the heading direction.

The recovered depth maps were then used to rerender the scene from novel viewpoints. Examples are shown in figures 10g and 10h. The basic structures of the scene (e.g., the valley, the cliff face on the left) are easily visible. The depth errors are evident along the boundaries and in the background (the peak in the background of 10g should not be there).

Since the rotational component of the flow field is quite small in this case, it is difficult to evaluate the significance of the rotation errors. By way of comparison, we computed depth using the correct flow field, the correct translation, and a rotation of zero. Using zero rotation yields significant errors in the depth values. Indeed, *negative* depth values were obtained for many of the distant points. Our recovered translation and rotation gave far better depth maps.

7 Summary

We propose a simple method for recovering 3D motion and depth from optical flow fields. Our approach is to subdivide the problem, solving first for translation, independent of depth and rotation.

Translation is recovered by evaluating a residual function over the discretely sampled space of all candidate translation directions. The residual function is used to assess how well each candidate translation accounts for the flow field. A residual value of zero for a particular candidate translation indicates that the flow field is consistent with that translation. The minimum of the residual function corresponds to the least-squares estimate of translation.

The residual function is evaluated as a linear summation of the input flow vectors, weighted by a fixed set of precomputed coefficients. Optical flow can be expressed as the product of a matrix that depends on the unknown translation, times a vector of the unknown rotation and inverse depth values. The coefficients for evaluating the residual function are a basis for the orthogonal complement of this matrix.

It is interesting to note that the color constancy problem can be solved in an analogous manner. Maloney and Wandell [1986] and Wandell [1987] explain that image color can be expressed as the product of matrix that depends on the spectral composition of the light source, times a vector that depends on the spectral reflectances of the surfaces. Maloney and Wandell solve for the lighting by finding the orthogonal complement to this matrix.

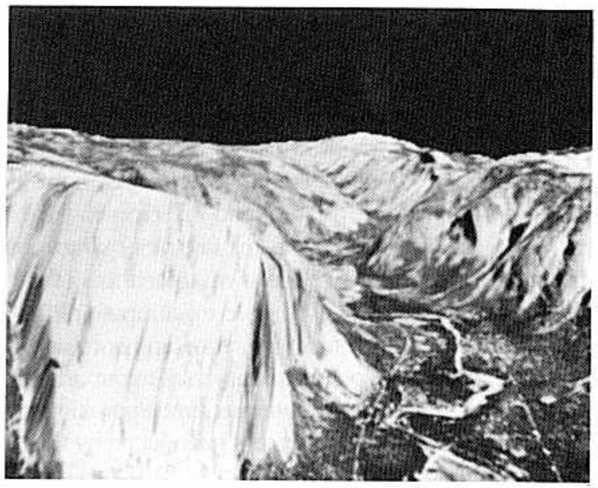
Unlike many previously proposed algorithms, our approach to motion analysis applies to the general case of arbitrary motion with respect to an arbitrary scene. There is no assumption of smooth or planar surfaces (see figure 9). Our results demonstrate that our algorithm is stable with respect to random errors in the flow field measurements. It performs favorably when compared with two other well-known algorithms (figures 5 and 6). Our method is simple to compute and it is highly parallel, not requiring iteration and not requiring an initial guess. One potential drawback to our approach is that it relies on the instantaneous-time approximation. However, we find that the errors introduced by this approximation are quite small (figure 8) if the rotation is less than 3 degrees per frame (90 degrees per second at video rates).

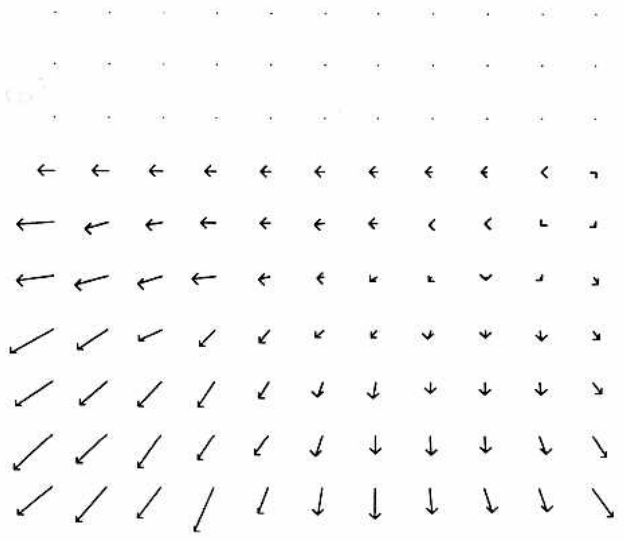
Our algorithm is certainly simple enough (a linear summation followed by squaring/rectification followed by a summation) to be implemented in visual cortex. A single cell could compute the residual value for a given candidate translation, and for a given patch of the visual field. That cell would be specifically suited for assessing a particular 3D translation. A whole battery of such cells, operating in parallel, would encode the perceived direction of motion as the minimum in the distribution of their responses.

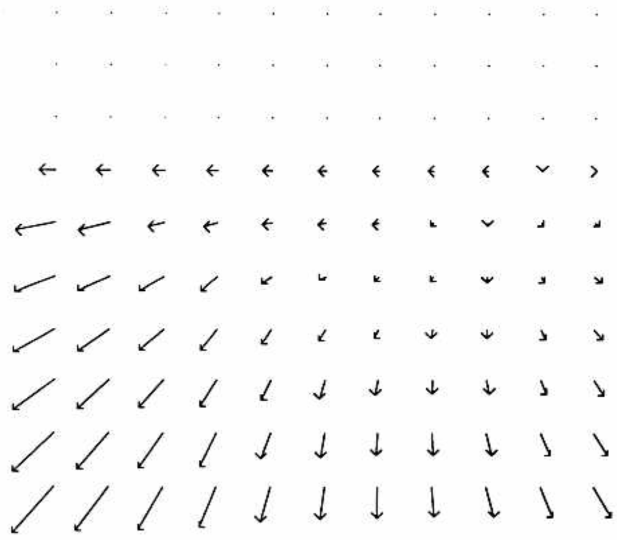
While it is known that cells in several cortical areas (e.g., areas MT and MST) of the primate brain are selective for 2D image velocity, physiologists have not yet tested whether these cells are selective for 3D motion. Cells in area MT are known to have velocity-selective receptive fields with a center-surround spatial organization. The cells prefer motion in one direction in the center, and motion in the opposite direction in the surround. We have shown [Jepson and Heeger 1990] that an analogous center-surround organization is a natural consequence of our analysis.

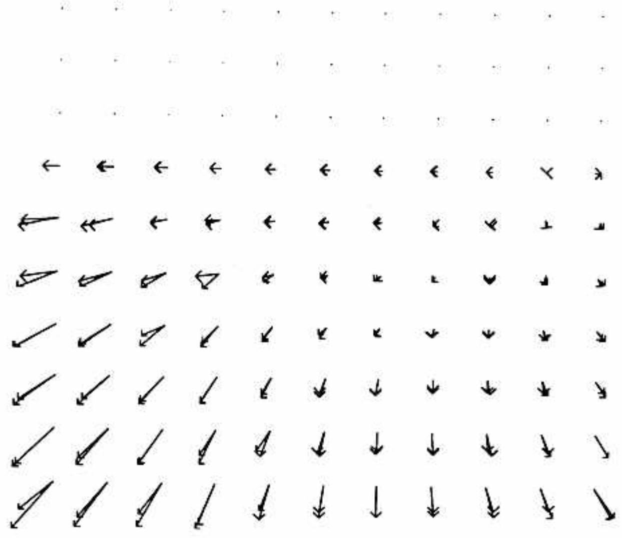
In our companion article [Jepson and Heeger 1990] we prove a number of mathematical results on the uniqueness and robustness of our approach.

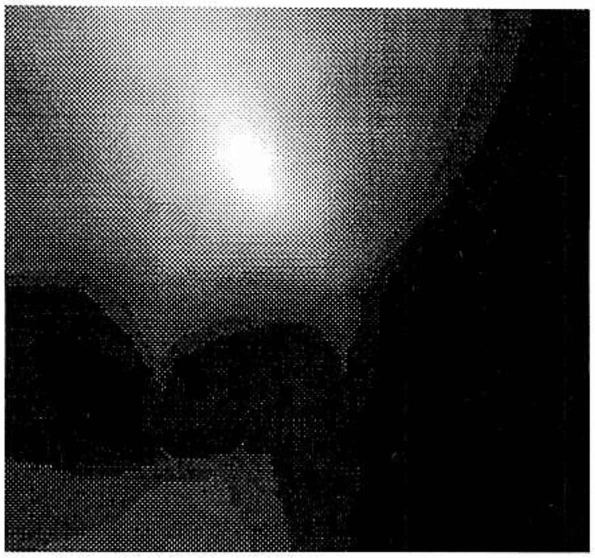
We are currently working on several extensions to the work presented here including "direct" methods like that outlined in appendix B, a sequential method for updating 3D motion and depth estimates over time, and extensions for dealing with scenes in which there are multiple motions. We have also recently developed a new subspace method that solves for translation in closed form (without sampling the solution space), thereby requiring much less computation [Jepson and Heeger 1991].


(a)

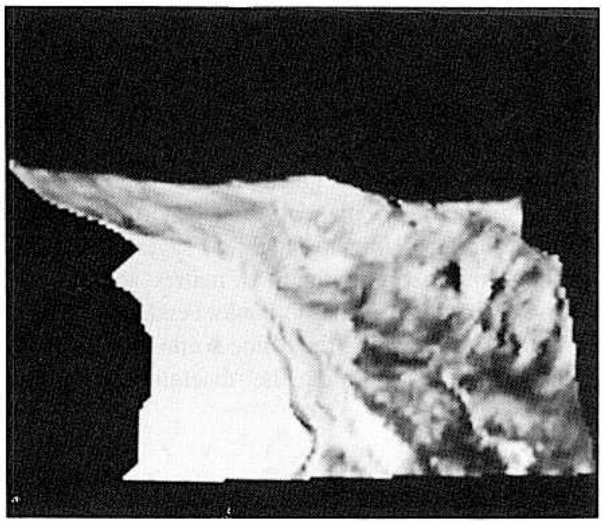

(c)


(b)

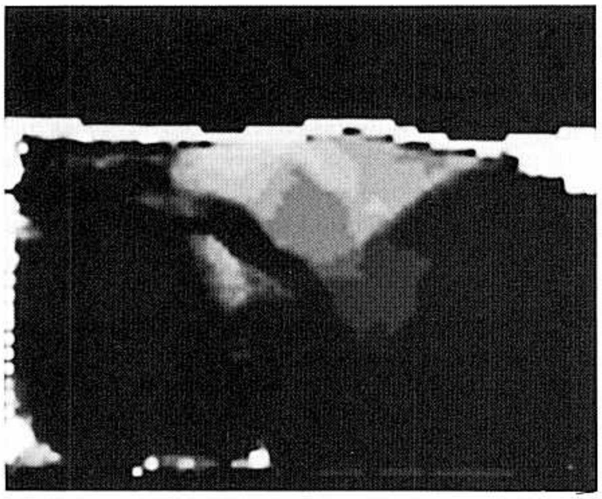

(d)



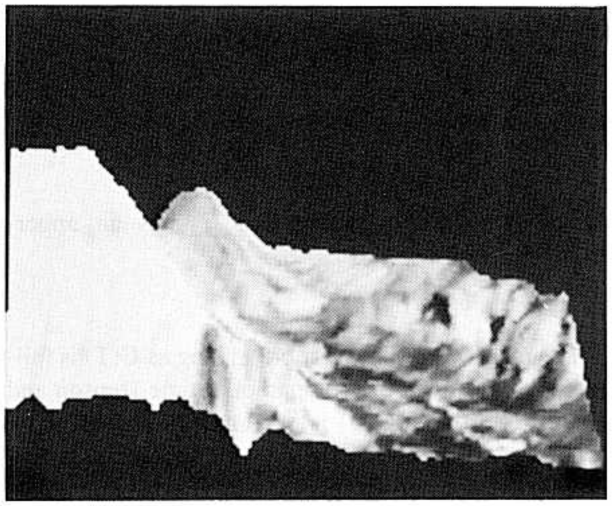
(e)



(g)



(f)



(h)

Fig. 10. (a) Frame from computer-graphics generated image sequence flying through Yosemite valley, simulating a camera with 40 degree field of view. (b) Flow field computed from actual known motion and depth values. (c) Flow field measured from the image sequence using method developed by Simoncelli et al. [1991]. (d) Correct flow superimposed on recovered flow. (e) Residual surface computed from recovered flow. Brightness is proportional to $-\log\,[E(\mathbf{T})]$. Peak location corresponds to the estimate of translation direction, in error by 3.5 degrees. (f) Relative percent error in the depth estimates, brightness is proportional to error. (g) and (h) Rerender scene from novel viewpoints using recovered depth map. Basic scene structure is recovered correctly. Errors along the boundaries due to edge effects of measuring the flow fields. Errors in the background arise because small errors in the flow measurements can result in large depth errors, near the heading direction.

Acknowledgments

David Heeger was supported by the MIT Media Lab, by NASA-Ames grant NCC2-307, and by NASA RTOP 505-64-13. Allan Jepson was supported by NSERC Canada, and by CIAR.

This research benefited greatly from discussions with Eero Simoncelli, P. Anandan, and David Marimont. Thanks to Shimon Ullman and Tomi Poggio for suggesting the comparisons with the *E*-matrix and Reiger-Lawton algorithms. Thanks to Sandy Pentland and Brad Horowitz for generating the office-scene image, and to Lynn Quam for generating the Yosemite fly-through image sequence.

Appendix A: Least-Squares Estimate

The least-squares estimate for translation minimizes the following expression over all candidate translations, rotations, and depth values:

$$E(\mathbf{T}, \mathbf{q}) = \|\mathbf{v} - \mathbf{C}(\mathbf{T})\mathbf{q}\|^2$$

Using the QR-decomposition [Strang 1980], the matrix $\mathbf{C}(\mathbf{T})$ can be written as the product,

$$\mathbf{C}(\mathbf{T}) = \bar{\mathbf{C}}(\mathbf{T})\mathbf{U}(\mathbf{T})$$

of an orthonormal matrix $\bar{\mathbf{C}}(\mathbf{T})$, times an upper-triangular matrix, $\mathbf{U}(\mathbf{T})$. Also, let

$$\bar{\mathbf{q}} = \mathbf{U}(\mathbf{T})\mathbf{q}$$

The matrix $\mathbf{U}(\mathbf{T})$ is invertible as long as $\mathbf{C}(\mathbf{T})$ is full-rank. We prove in the companion article [Jepson and Heeger 1990] that $\mathbf{C}(\mathbf{T})$ is guaranteed to be full-rank for almost all choices of sample points and almost any $\mathbf{T}$. In particular, an arrangement of five sample points like that shown on dice is satisfactory.

Given that $\mathbf{C}(\mathbf{T})$ is full-rank, we rewrite the above expression as

$$E(\mathbf{T}, \bar{\mathbf{q}}) = \|\mathbf{v} - \bar{\mathbf{C}}(\mathbf{T})\bar{\mathbf{q}}\|^2$$

We seek to minimize this expression over all $\mathbf{T}$ and $\bar{\mathbf{q}}$. First we minimize over $\bar{\mathbf{q}}$ by taking the derivative with respect to $\bar{\mathbf{q}}$, setting it equal to zero, and solving for $\bar{\mathbf{q}}$ to give

$$\hat{\mathbf{q}}(\mathbf{T}) = \bar{\mathbf{C}}^t(\mathbf{T})\mathbf{v}$$

For a fixed choice of $\mathbf{T}$, the least-squares expression, $E(\mathbf{T}, \bar{\mathbf{q}})$ is minimized by setting $\bar{\mathbf{q}} = \hat{\mathbf{q}}(\mathbf{T})$. Substituting $\hat{\mathbf{q}}(\mathbf{T})$ for $\bar{\mathbf{q}}$ gives the residual function in terms of $\mathbf{T}$ alone:

$$\begin{aligned} E(\mathbf{T}) &= \|\mathbf{v} - \bar{\mathbf{C}}(\mathbf{T})\bar{\mathbf{C}}^t(\mathbf{T})\mathbf{v}\|^2 \\ &= \|(\mathbf{I} - \bar{\mathbf{C}}\bar{\mathbf{C}}^t)\mathbf{v}\|^2 \end{aligned}$$

where $\mathbf{I}$ is the identity matrix.

Recalling that $\bar{\mathbf{C}}$ is orthonormal, $(\mathbf{I} - \bar{\mathbf{C}}\bar{\mathbf{C}}^t)$ is a projection onto the null space of $\mathbf{C}^t$. Therefore, the above expression reduces to:

$$E(\mathbf{T}) = \|\mathbf{v}^t\mathbf{C}^\perp(\mathbf{T})\|^2$$

where $\mathbf{C}^\perp$ is an orthonormal basis for the nullspace of $\mathbf{C}^t$.

The matrix, $\mathbf{C}(\mathbf{T})$, divides $\mathbf{v}$-space into two subspaces. The subspace that is spanned by the columns of $\mathbf{C}(\mathbf{T})$ is called the range of $\mathbf{C}(\mathbf{T})$. The leftover orthogonal subspace is called the orthogonal complement of $\mathbf{C}(\mathbf{T})$. $\mathbf{C}^\perp$ is a basis for this orthogonal complement. It is straightforward, using techniques of numerical linear algebra [Strang 1980], to choose a $\mathbf{C}^\perp$ matrix given $\mathbf{C}(\mathbf{T})$. The residual function is defined in terms of this orthogonal complement.

Appendix B: Direct Method

In this appendix, we extend our algorithm to be a "direct" method that works directly from the spatiotemporal gradients of image intensity. First we review a simple algorithm for measuring the flow field from the spatiotemporal gradients of image intensity. Then we show how to compose the two steps of visual motion analysis.

Following Lucas and Kanade [1981], a least-squares estimate for image velocity minimizes

$$E(\mathbf{v}) = \sum_{xy} [I(x + v_x, y + v_y) - I'(x, y)]^2$$

where $I(x, y)$ and $I'(x, y)$ are consecutive frames from the image sequence, and $\mathbf{v}(x, y) = (v_x, v_y)$ is the image velocity at (x, y). The summation is taken over a local image patch, for example, in a Gaussian weighted window. Under the approximation that the image intensity surface is locally planar, the minimum is given by

$$\mathbf{G}_t(x, y) = \mathbf{G}_{xy}(x, y)\mathbf{v}(x, y) \tag{11}$$

where,

$$\mathbf{G}_t(x, y) = \begin{bmatrix} \Sigma\, I_x I_t \\ \Sigma\, I_y I_t \end{bmatrix}$$

$$\mathbf{G}_{xy}(x, y) = \begin{bmatrix} \Sigma\, I_x I_x & \Sigma\, I_x I_y \\ \Sigma\, I_x I_y & \Sigma\, I_y I_y \end{bmatrix}$$

I_x, I_y, and I_t are the spatial and temporal derivatives of image brightness, and the summations are local averages of the gradient products. The velocity estimate is then given by

$$\hat{v}(x, y) = \mathbf{G}_{xy}^{-1}(x, y)\mathbf{G}_t(x, y)$$

assuming that $\mathbf{G}_{xy}(x, y)$ is invertible. The case of $\mathbf{G}_{xy}(x, y)$ not invertible corresponds to the aperture problem; there are not enough constraints to solve for both unknowns. So long as $\mathbf{G}_{xy}(x, y)$ is not singular, equation (11) gives an estimate for image velocity. For each of N sample points, a separate equation can be written in the form of equation (11). The equations can also be collected together into one matrix equation (reusing the symbols in equation (11) rather than introducing new notation);

$$\mathbf{G}_t = \mathbf{G}_{xy}\mathbf{v} \tag{12}$$

where,

$$\mathbf{v} = [v_x(x_1, y_1), v_y(x_1, y_1), \ldots, v_x(x_N, y_N), v_y(x_N, y_N)]^t$$

$$\mathbf{G}_t = \begin{bmatrix} \mathbf{G}_t(x_1, y_1) \\ \cdot \\ \cdot \\ \cdot \\ \mathbf{G}_t(x_N, y_N) \end{bmatrix}$$

$$\mathbf{G}_{xy} = \begin{bmatrix} \mathbf{G}_{xy}(x_1, y_1) & \ldots & 0 \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ 0 & \ldots & \mathbf{G}_{xy}(x_N, y_N) \end{bmatrix}$$

Equation (12) relates the spatiotemporal intensity gradients and the flow field. Equation (4) relates the flow field and the 3D motion parameters. Composing the two equations gives

$$\mathbf{G}_t = \mathbf{G}_{xy}\mathbf{C}(\mathbf{T})\mathbf{q}$$

Following the presentation in appendix A, we define a new residual function:

$$E(\mathbf{T}) = \|[\mathbf{G}_{xy}^{-1}\mathbf{G}_t]^t\mathbf{C}^{\perp}(\mathbf{T})\|^2$$

The estimate of translation is the choice of $\mathbf{T}$ that minimizes this residual function. As above, we assume that $\mathbf{G}_{xy}(x, y)$ is invertible.

The advantage of this "direct" method is that it avoids the difficulty of computing the flow field. Variants of this "direct" method could be derived by starting with different flow algorithms. Some flow algorithms are based on a relationship between image velocity and the first and second spatial and temporal derivatives of image intensity. Other flow algorithms are based on a relationship between velocity and the frequency response of motion-selective spatiotemporal filters. In any case, the given relationship can be composed with equation (4) to yield a "direct" method.

Appendix C: Reiger-Lawton Implementation

This appendix briefly describes our implementation of the Reiger-Lawton [1985] algorithm.

First, the flow field is subdivided into M nonoverlapping patches, each containing N flow-vector samples. For each patch we take the flow vector $\mathbf{v}(x_i, y_i)$ at the center of the patch and compute difference vectors with other flow vectors $\mathbf{v}(x_j, y_j)$ in that patch. Let the difference vectors be denoted by

$$(l_x^{ij}, l_y^{ij}) = \mathbf{v}(x_i, y_i) - \mathbf{v}(x_j, y_j)$$

Next we determine an average difference vector $\mathbf{l}_i$ for the entire patch by fitting a line through (x_i, y_i) that minimizes the sum of squared perpendicular distances to the set of difference vectors. To do this we first compute the unit vector $\mathbf{e}_i$ that minimizes

$$\|\mathbf{L}_i\mathbf{e}_i\|$$

where

$$\mathbf{L}_i = \begin{pmatrix} l_x^{i0} & l_y^{i0} \\ l_x^{i1} & l_y^{i1} \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ l_x^{iN} & l_y^{iN} \end{pmatrix}$$

The solution is the eigenvector corresponding to the smallest eigenvalue of $\mathbf{L}_i^t\mathbf{L}_i$ [Strang 1980]. The average vector $\mathbf{l}_i$ for the patch is perpendicular to unit vector $\mathbf{e}_i$.

Next, we combine the average difference vectors from different patches to solve for the heading direction. Let $\mathbf{s}_i = (x_i, y_i, f)$ be the sample positions at the center of each patch (f is focal length). For each patch $\mathbf{T}$ is constrained to lie in the plane that contains both s_i and $\mathbf{l}_i$. That is, $\mathbf{T}$ is perpendicular to $\mathbf{s}_i \times \mathbf{l}_i$. Collecting these constraints over all the image patches, we compute the unit vector $\mathbf{T}$ that minimizes

$$\|\mathbf{QT}\|$$

where

$$\mathbf{Q} = \begin{pmatrix} \mathbf{s}_0 \times \mathbf{l}_0 \\ \mathbf{s}_1 \times \mathbf{l}_1 \\ \cdot \\ \cdot \\ \cdot \\ \mathbf{s}_M \times \mathbf{l}_M \end{pmatrix}$$

The solution is the eigenvector corresponding to the smallest eigenvalue of $\mathbf{Q}^t\mathbf{Q}$.

An alternative to this last calculation would be to locate the FOE as the point where the difference vectors intersect. However, it is not always possible to find an intersection point for the FOE. When the observer is looking straight ahead and moving directly to the side, for example, the FOE is at infinity. The above computation, by contrast, is well defined even when the FOE is at infinity.

References

Adelson, E.H., and Bergen, J.R., 1986. The extraction of spatiotemporal energy in human and machine vision. *Proc. IEEE Workshop on Motion: Representation and Analysis*, Charleston, S. Carolina, pp. 151–156.

Adiv, G., 1985. Determining three-dimensional motion and structure from optical flow generated by several moving objects. *IEEE Trans. Patt. Anal. Mach. Intell.* 7:384–401.

Adiv, G., 1989. Inherent ambiguities in recovering 3D motion and structure from a noisy flow field. *IEEE Trans. Anal. Mach. Intell.* 11:477–489.

Aggarwal, J.K. and Nandhakumar, N., 1988. On the computation of motion from sequences of images—a review. *Proc. IEEE.* 76: 917–935.

Anandan, P., 1989. A computational framework and an algorithm for the measurement of visual motion. *Intern. J. Comput. Vision.* 2:283–310.

Ballard, D.H. and Kimball, O.A., 1983. Rigid body motion from depth and optical flow. *Comput. Vision, Graph. Image Process.* 22:95–115.

Barron, J., 1984. A survey of approaches for determining optic flow, environmental layout and egomotion. Techn. *Rept.* RBCV-TR-84-5, Department of Computer Science, University of Toronto.

Barron, J.L., Jepson, A.D., and Tsotsos, J.K., 1990. The feasibility of motion and structure from noisy time-varying image velocity information. *Intern. J. Comput. Vision* 5:239–269.

Bolles, R.C., Baker, H.H., and Marimont, D.H., 1987. Epipolar-plane image analysis: An approach to determining structure from motion. *Intern. J. Comput. Vision* 1:7–55.

Broida, T.J., and Chellappa, R., 1986. Estimation of object motion parameters from noisy images. *IEEE Trans. Patt. Anal. Mach. Intell.* 8:90–99.

Bruss, A.R., and Horn, B.K.P., 1983. Passive navigation. *Comput. Vision, Graph. Image Process.* 21:3–20.

Faugeras, O.D., Lustman, F., and Toscani, G., 1987. Motion and structure from motion from point and line matches. *Proc. 1st Intern. Conf. Comput. Vision*, London, June, pp. 25–34.

Fleet, D.J. and Jepson, A.D., 1990. Computation of component image velocity from local phase information. *Intern. J. Comput. Vision.* 5:77–104.

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

Gibson, J.J., and Gibson, E.J., 1957. Continuous perspective transformation and the perception of rigid motions. *J. Exp. Psychol.* 54: 129–138.

Grzywacz, N.M., and Yuille, A.L., 1990. A model for the estimate of local image velocity by cells in the visual cortex. *Proc. Roy. Soc. London A*, 239:129–161.

Hay, J.C., 1966. Optical motions and space perception: An extension of Gibson's analysis. *Psychological Review*, 73:550–565.

Heeger, D.J., 1987. Model for the extraction of image flow. *J. Opt. Soc. Amer. A* 4:1455–1471.

Heeger, D.J., 1988. Optical flow using spatiotemporal filters. *Intern. J. Comput. Vision* 1:279–302.

Heeger, D.J., and Jepson, A., 1990a. Visual perception of three-dimensional motion. *Neural Computation* 2:129–137.

Heeger, D.J., and Jepson, A., 1990b. Visual perception of 3D motion and depth. *Invest. Opthal. Vis. Sci. Suppl.* 31:173.

Heeger, D.J., and Jepson, A., 1990c. Simple method for computing 3D motion and depth. *Proc. 3rd. Intern. Conf. Comput. Vision*, Osaka, Japan, December, pp. 96–100.

Heeger, D.J., and Jepson, A., 1991. Recovering observer translation with center-surround motion-opponent mechanisms. *Invest. Opthal. Vis. Sci. Suppl.* 32:823.

Heel, J., 1989a. Direct estimation of structure and motion for multiple frames. Tech. Rep. 1190, MIT AI Lab.

Heel, J., 1989b. Dynamic motion vision. *Proc. SPIE*. Philadelphia.

Heel, J., 1990. Direct dynamic motion vision. *Proc. IEEE Conf. Robot. Autom.* Cincinnati.

Horn, B.K.P., 1986. *Robot Vision*. MIT Press: Cambridge, Ma.

Horn, B.K.P., 1987. Motion fields are hardly ever ambiguous. *Intern. J. Comput. Vision*, 1:259–274.

Horn, B.K.P., and Negahdaripour, S., 1987. Direct passive navigation: Analytical solution for planes. *IEEE Trans. Patt. Anal. Mach. Intell.* 9:168–176.

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

Horn, B.K.P., and Weldon, E.J., 1988. Direct methods for recovering motion. *Intern. J. Comput. Vision* 2:51–76.

Jepson, A., and Heeger, D.J., 1989. Egomotion without depth estimation. *Optics News* 15:A–20.

Jepson, A., and Heeger, D.J., 1990. Subspace methods for recovering rigid motion II: Theory. Submitted to *International Journal of Computer Vision*, available as Tech. *Rept.* RBCV-TR-90-36, Department of Computer Science, University of Toronto.

Jepson, A. and Heeger, D.J. 1991. A fast subspace algorithm for recovering rigid motion. *Proc. IEEE Workshop on Visual Motion*, Princeton, N.J., pp. 124–131.

Johansson, G., 1975. Visual motion perception. *Scientific American* 232:76–88.

Koenderink, J.J. and van Dorn, A.J., 1975. Invariant properties of the motion parallax field due to the movement of rigid bodies relative to an observer. *Optica Acta* 22:773–791.

Koenderink, J.J., and van Dorn, A.J., 1976. Local structure of movement parallax of the plane. *J. Opt. Soc. Amer.* 66:717–723.

Koenderink, J.J., and van Dorn, A.J., 1981. Exterospecific component of the motion parallax field. *J. Opt. Soc. Amer.* 71:953–957.

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

Longuet-Higgins, H.C., 1984. The visual ambiguity of a moving plane. *Proc. Roy. Soc. London B* 223:165–175.

Longuet-Higgins, H.C., 1988. Multiple interpretations of a pair of images of a surface. *Proc. Roy. Soc. London A* 418;1–15.

Longuet-Higgins, and Prazdny, K. 1980. The interpretation of a moving retinal image. *Proc. Roy. Soc. London B* 208;385–397.

Lucas, B.D., and Kanade, T., 1981. An iterative image registration technique with an application to stereo vision. *Proc. 7th Intern. Joint Conf. Artif. Intell.* Vancouver, pp. 674–679.

Maloney, L.T., and Wandell, B.A., 1986. Color constancy: a method for recovering surface spectral reflectance. *J. Opt. Soc. Amer. A* 1:29–33.

Matthies, L., Szeliski, R., and Kanade, T., 1989. Kalman filter-based algorithms for estimating depth from image sequences. *Intern. J. Comput. Vision* 3:209–238.

Maybank, S.J., 1985. The angular velocity associated with the optical flow field arising from motion through a rigid environment. *Proc. Roy. Soc. London A* 410:317–326.

Maybank, S.J., 1987. *A Theoretical Study of Optical flow.* Ph.D. thesis, University of London.

Nagel, H.H., 1987. On the estimation of optical flow: relations between different approaches and some new results. *Artificial Intelligence* 33:299–324.

Nakayama, K., 1985. Biological image motion processing: A review. *Vision Research* 25:625–660.

Negahdaripour, S., and Horn, B.K.P., 1989. A direct method for locating the focus of expansion. *Comput. Vision, Graph. Image Process.* 46:303–326.

Prazdny, K., 1980. Egomotion and relative depth from optical flow. *Biological Cybernetics* 36:87–102.

Prazdny, K., 1981. Determining the instantaneous direction of motion from optical flow generated by a curvilinearly moving observer. *Comput. Graph. Image Process.* 17:238–248.

Prazdny, K., 1983. On the information in optical flows. *Comput. Graph. Image Process.* 22:239–259.

Reiger, J.H., and Lawton, D.T., 1985. Processing differential image motion. *J. Opt. Soc. Amer. A* 2:354–359.

Roach, J.W., and Aggarwal, J.K., 1980. Determining the movement of objects from a sequence of images. *IEEE Trans. Patt. Anal. Mach. Intell.* 2:554–562.

Simoncelli, E.P., and Adelson, E.H., 1991. Relationship between gradient, spatio-temporal energy, and regression models for motion perception. *Invest. Opthal. Vis. Sci. Suppl.* 32:893.

Simoncelli, E.P., Adelson, E.H., and Heeger, D.J., 1991. Probability distributions of optical flow. *Proc. Comput. Vision Patt. Recog.*. Maui, HI, June, pp. 310–315.

Southall, J.P.C., editor, 1962. *Helmholtz's Treatise on Physiological Optics.* Dover Publications: NY Originally published by the Optical Society of America in 1925.

Strang, G., 1980. *Linear Algebra and Its Applications.* Academic Press: New York.

Sundareswaran, V. 1991. Egomotion from global flow field data. *Proc. IEEE Workshop on Visual Motion*, Princeton, N.J., pp. 140–145.

Tsai, R.Y., and Huang, T.S., 1984. Uniqueness and estimation of three-dimensional motion parameters of rigid objects with curved surfaces. *IEEE Trans. Patt. Anal. Mach. Intell.* 6:13–27.

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

Ullman, S., 1984. Maximizing rigidity: the incremental recovery of 3-D structure from rigid and rubbery motion. *Perception* 13:255–274.

Wallach, H., and O'Connell, D.N., 1953. The kinetic depth effect. *J. Exp. Psychol.* 45:205–217.

Wandell, B.A., 1987. The synthesis and analysis of color images. *IEEE Trans. Patt. Anal. Mach. Intell.* 9:2–13.

Warren, W.H., and Hannon, D.J., 1988. Direction of self-motion is perceived from optical flow. *Nature* 336:162–163.

Warren, W.H., and Hannon, D.J., 1990. Eye movements and optical flow. *J. Opt. Soc. Amer. A* 7:160–169.

Watson, A.B., and Ahumada, A.J., 1985. Model of human visual-motion sensing. *J. Opt. Soc. Amer. A* 2:322–342.

Waxman, A.M., and Ullman, S., 1985. Surface structure and three-dimensional motion from image flow kinematics. *Intern. J. Robot. Res.* 4:72–94.

Waxman, A.M., and Wohn, K., 1985. Contour evolution, neighborhood deformation, and global image flow: planar surfaces in motion. *Intern J. Robot. Res.* 4:95–108.

Waxman, A.M., and Wohn, K., 1988. Image flow theory: A framework for 3-D inference from time-varying imagery. In *Advances in Computer Vision*. vol. 1, pp. 165–224. Lawrence Erlbaum Assoc.: Hillsdale, NJ.

Waxman, A.M., Kamgar-Parsi, B., and Subbarao, M., 1987. Closed-form solutions to image flow equations and 3D structure and motion. *Intern. J. Comput. Vision* 1:239–258.

Weng, J., Huang, T.S., and Ahuja, N., 1989. Motion and structure from two perspective views: Algorithms, error analysis, and error estimation. *IEEE Trans. Patt. Anal. Mach. Intell.* 11:451–476.