COMPUTER VISION AND IMAGE UNDERSTANDING
Vol. 64, No. 1, July, pp. 21–52, 1996
ARTICLE NO. 0044

Computing Egomotion and Detecting Independent Motion from Image Motion Using Collinear Points

NIELS DA VITORIA LOBO* AND JOHN K. TSOTSOS

Department of Computer Science, University of Toronto, Toronto, Canada M5S 1A4

Received March 17, 1994; accepted April 28, 1995

This paper presents and analyzes a new constraint (the Collinear Point Constraint) for use in computing egomotion and detecting independently moving objects from image motion. The constraint provides an elegant means of cancelling the rotation of the observer while simultaneously providing evidence for the location of the Translational Focus of Expansion. It also provides the minimal computation necessary for detecting independent motion. © 1996 Academic Press, Inc.

1. INTRODUCTION

Where are you going? This is a question normally-sighted humans have no difficulty in answering regardless of whether we are actually moving or if the environment is moving around us. Whatever computations we perform in reaching an answer to this question seem effortless and vary fast. This observation implies that an algorithm that purports to solve this problem must also be fast and inexpensive computationally.

How do we use the answer to such a question? Most often, we direct our gaze into the direction of the motion (imagine driving a car and not focusing on the road directly in front of you!). The importance of focusing our gaze on new territory as we move seems self-evident. Any changes in gaze must be computed in an equally fast and effortless manner.

This paper presents a new method for the computation of egomotion that uses a simple constraint, the Collinear Point Constraint. If there is sufficient depth variation in the world through which an agent moves, simple computations on collinear point structures in the image lead to an elegant solution to computing the translation of the observer by simultaneously canceling out rotation. Further, the constraint is extended to solve the problem of detecting and localizing independently moving regions.

We present an empirical and theoretical analysis of this method and show that the computations can be performed in time which is linear in the number of points for a dense flow field, thus satisfying the need for a fast solution. The detection of independently moving regions is based on a theorem which we derive giving the minimal number of points needed for the detection of such regions. The theory is compared to that of previous methods; experimental verification of its performance using synthetic input is presented.

These two computations provide important inputs for an attention system (such as that described in [45, 5]) in that two separate representations of attentional cues are developed in a highly efficient manner, direction of egomotion (a cue for gaze maintenance) and locations of independently moving regions (cues for shifts of attention and perhaps gaze). Thus, the importance of this algorithm for agents who behave in the real world based on visual input is clear.

In subsequent sections, we first present a review of the models used for computing rigid motion and a review of previous work. Then, we introduce and discuss the new solution to computing egomotion using collinear points. After that, we examine the problem of independently moving objects and their detection. We then present a review of previous efforts to solve this problem and introduce our solution.

2. MODELS FOR COMPUTING RIGID MOTION

Models for computing rigid 3-D motion relate image measurables, such as the image motion of a point or the positions of a point in several images, to the depth in the scene and the 3-D motion. Image motion is commonly modeled in two ways, infinitesimally in time and over discrete time intervals. In the first case, it is termed the *2-D instantaneous motion field* or *optical flow,* and in the latter the *displacement map.*

2.1. The 2-D Instantaneous Motion Field

A left-handed observer-centered 3-D Cartesian coordinate system is used which has its origin at the focal point

* The first author can now be contacted at the Department of Computer Science, University of Central Florida, Orlando, FL 32816.

of the observer's imaging device. The Z-axis coincides with the optical axis, and the imaging surface (for convenience, chosen to be a plane) is modeled at $Z = 1$. The 2-D coordinate system on the imaging surface originates where the optical axis intersects the image. Scene parameters are denoted by capital letters, while the projected parameters are symbolized by lower-case letters. Thus a point P at (X, Y, Z) in the scene projects to point p at $(x = X/Z, y = Y/Z)$ in the image plane.

Relative motion between the observer and the scene can be modeled as motion of the observer. Thus, when the observer moves with instantaneous motion parameters (U, V, W, A, B, C) a point in space moves relative to an observer. Then the image velocity (u, v) for a point at (x, y) is derived as follows (for the derivation, see [24] or [4]):

$$\begin{aligned} u &= \frac{-U + xW}{Z} - Axy + B(x^2 + 1) - Cy, \\ v &= \frac{-V + yW}{Z} - A(y^2 + 1) + Bxy + Cx. \end{aligned} \tag{1}$$

These equations have already been in use in photogrammetry (see [10]).

2.2. The 2-D Displacement Map

Another way to model motion is as the discrete transformation that points in the scene undergo between two views. Then, due to observer motion, a point P at (X, Y, Z) is displaced to (X', Y', Z') and in the image plane from (x, y) to (x', y'). The 3-D transformation can be expressed as a rotation around the three axes followed by a translation. The 2-D image displacement is related to 3-D parameters as follows:

$$\begin{aligned} u &= x' - x \\ &= \frac{x \cos B \cos C - y \cos B \sin C + \sin B - (U/Z)}{x(-\cos A \sin B \cos C + \sin A \sin C) + y(\cos A \sin B \sin C + \sin A \cos C) + \cos A \cos B - W/Z} - x \\ v &= y' - y \qquad (2) \\ &= \frac{x(\sin A \sin B \cos C + \cos A \sin C) + y(-\sin A \sin B \sin C + \cos A \cos C) - \sin A \cos B - V/Z}{x(-\cos A \sin B \cos C + \sin A \sin C) + y(\cos A \sin B \sin C + \sin A \cos C) + \cos A \cos B - W/Z} - y. \end{aligned}$$

$$(3)$$

Current technology for measuring image information discretely in time makes it impossible to obtain 2-D instantaneous motion (velocity) information, while, at least in principle, displacement maps are computable for situations where the spatial intensity structure provides dense texture. Hence, the displacement formulation can be argued to be the more practical of the two models. However, approximations to instantaneous flow fields are obtainable by employing higher temporal sampling and averaging the image motion over the course of several views.

While instantaneous flow is not computable currently (and, where there is insufficient texture, instantaneous flow may be impossible to compute), its property of separability of the translational and rotational components make it an attractive model of interpretation. In addition, computing instantaneous flow is, at least in theory, less susceptible to the problem of significant changes between views: these changes make correspondence difficult when attempting to compute the displacement map. In [1], [21], and [53], the relationship between the instantaneous and the displacement model has been derived using slightly different assumptions. The ability to acquire motion data under these assumptions is improving, making the instantaneous model increasingly realistic. For this research, the instantaneous model using perspective projection is adopted.

2.3. The Focus of Expansion

In the instantaneous model using perspective projection, an important invariant (invariant relative to the flow field) is the Focus of Expansion (FOE), defined here to be where the imaging surface (which could be planar, hemispherical, etc.) is intersected by the direction of instantaneous observer translation. This defines the FOE to be independent of the viewer's instantaneous rotation.

For an image plane at $Z = 1$, let the FOE be at (x_f, y_f). The 3-D translation vector (U, V, W) can be expressed as (Wx_f, Wy_f, W), for some W (the translation component along Z), and using $U = Wx_f$ and $V = Wy_f$. When there is no observer translation, the FOE does not exist. When the translation has no component in depth, the FOE lies at infinity (in the direction of (U, V)) for the case of a planar imaging surface; however, for a hemispherical imaging surface, the FOE lies at the rim of the hemisphere. There is a one-to-one mapping between the direction of translation (up to a sign factor) and the FOE. Figure 1 helps to explain how rotation is completely irrelevant to the position of the FOE as defined above.

3. PREVIOUS WORK

In this section, representative examples of previous work in solving the motion equations presented in the previous section are reviewed. Reviewing the perspective case first, work with the instantaneous equations is described followed by work using the displacement equations. Then, past research conducted under the assumption of orthogra-

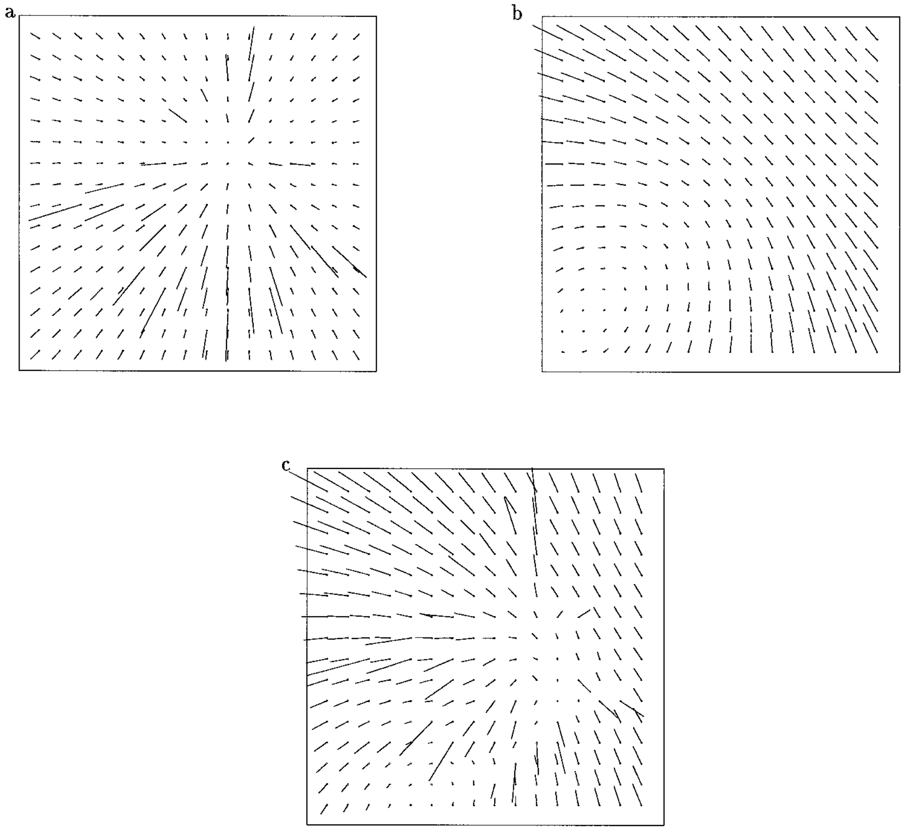


FIG. 1. This figure shows flow fields generated when observer (a) only translates (translation parameters $[U, V, W] = [2.2, 2.2, 10.1]$), (b) only rotates (rotation parameters $[A, B, C] = [-0.04, -0.04, 0.04]$), and (c) moves with both rotation and translation, the most typical situation. In the second case there is no FOE because there is no observer translation, while in the first and third cases the FOE is in the same position for both cases, just above and to the right of the image center; i.e., this definition of FOE renders its position in the image completely independent of observer rotation for the case of instantaneous motion projected in perspective. The image motion data here is synthesized according to the description in Section 4.9.1 from depth data shown in Fig. 4a.

phy is recounted. Since this paper pursues instantaneous analysis, the review of these approaches receives greater attention than those using the displacement model.

3.1. Instantaneous Approaches

In [27], spherical projection was used to show that applying a center-surround symmetric operator to the image velocity field cancels out the 3-D rotational component of motion. In spherical projection, observer rotation is constant along lines of latitude with respect to the axis of rotation and is locally uniform. Their "convexity" operator was relegated to finding discontinuities in depth.

In [22], a decomposition of the motion parallax field caused by an observer purely translating relative to a planar surface has been considered. The components of the decomposition, deformation, curl, and divergence were found to encode information useful to the observer; for example, for known translation, the deformation specifies the slant of the surface uniquely.

In [24], the instantaneous motion equations (1) are used, along with the idea that a pair of points at the same (or very close) retinal location but at different depths (termed motion parallax, as might occur with transparent surfaces) can have their flow values subtracted to yield a vector pointing to the FOE. Thus, several separate difference computations, taken together, localize the FOE. Then, they calculated the rotation parameters from the spatially linear variation of the flow component that is orthogonal to the coordinate axes centered at the FOE. Subsequent subtraction of the rotation from the original flow field permits simple extraction of relative depth. To consider cases where motion parallax cues are not readily available or where parts of the scene are in relative motion, they employed first and second derivatives of the flow field at a single point and again made use of the linear variation of rotation in the special case along a reoriented vertical axis that passes through the FOE. Subsequently, [31] pursued an approach that conducted a three-parameter search through all possible candidate rotations. For each hypothesized rotation, the rotational field was subtracted from the input field and the resulting field was tested for how well it approximated a pure translational field.

Dense flow is assumed in [4], using minimization to accommodate input noise. For pure translation, they proposed to minimize the integral (across the field) of the component of flow orthogonal to a radial line through the hypothesized FOE. They derived the closed form solution to this, and a similar solution has recently been proposed for the case of translation with minute rotations [39]. Pure rotation was handled by a simple linear least-squares solution. For combined motion, the minimized expression led to seven expressions, three of which were linear in the rotation parameters; as such, rotation could be expressed in terms of translation while the other four would yield the translation parameters by an unspecified numerical method.

To compensate for the noise in flow around depth boundaries for motion parallax, [34] first computed flow differences between each local image velocity measurement and others measured within a small neighbourhood, then computing the dominant orientation of the resulting set of difference vectors. Strongly consistent directions were saved for each region and used to compute the FOE. However, the assumption that rotation is cancelled out by simple differencing is generally incorrect because the errors in the direction vectors of differences grow with the distance between the separate points.

In [1], the problem of motion segmentation for flow fields generated by several moving objects in the scene is attempted. Subdividing the image into regions and computing the 3-D motions independently for each region, an approximating residual function for all possible candidate directions of translation was minimized. Regions with common translation estimates were then grouped.

In exactly reducing the dimension of the search space, [2] showed that tracking a point (and knowing the rotation needed to do the tracking) reduced the dimension of the nonlinear search space by one. In [26], it is shown that assuming rigidity permits the elimination of two of the unknowns in the nonlinear system of equations.

Assuming smooth surfaces, [48, 47] showed that curved patches in rigid motion generated flow fields that were smooth within "boundaries of analyticity." They showed that the motion and shape parameters were computable in closed form from certain observables (combinations of derivatives) of the flow field. In addition to the fact that computing derivatives is extremely noise-sensitive, since much of the real world is not smooth, the smooth-surface assumption is not realistic.

A direct method for the computation of motion and shape which avoids the stage of computing optical flow explicitly has been proposed [28]. The instantaneous velocity equations are combined with the brightness constancy assumption to get a *brightness change constraint equation.* While earlier work with this framework was restricted to special cases of motions or shapes, recent work in [40] uses fixation to address the general problem.

The authors of [11] cancelled rotation for points in general configurations, and their work complements that reported here. Starting with five flow vectors they show that the resulting system of equations can be written as a transformation that is linear in the inverse depths and rotations, where the linear transformation itself is a function of the unknown translation. Thus, a least-squares estimate for translation can be formulated which minimizes the expression due to the transformation, over all candidate translations, rotations, and depth values. An equivalent expression is obtained by elimination of the rotation components, depth, and the magnitude of translation; it is minimized over the direction of translation alone. The residual function they used resembles Adiv's in concept but is exact and is efficiently computed. In a subsequent paper [18], they thoroughly analyze their method, proving several mathematical results on the uniqueness and robustness of the approach. More recently, in [19], they have continued their investigations producing an efficient linear approach to the problem of estimating the motion parameters. This novel technique has a natural bias in its output, which is compensated for by a computation that, as the authors report, needs further development before it can be applied to computed optical flow fields.

In [14], two operators are defined, one that finds the rotational parameters and another that, like the work presented in this paper, is an FOE locator. The first, called the flow circulation algorithm, computes the curl of the flow field which under many conditions is shown to be approximately a linear function. Circulation values are computed for various centroids in the image and the rotation parameters are estimated. The FOE operator consists of the computation of several concentric circular components via contour integrals around candidate FOEs, which are then fit to a polynomial to account for the rotation component. Alternatively, during the computation of the concentric integrals a center-surround kernel is applied concentrically so that the rotation component is cancelled. Their approach does not address the detection of independently moving objects.

In [52], a representation, which is a first-order approximation to the image motion field, is defined. It is based on the observation that for a very small field of view, the quadratic terms in the instantaneous velocity equations disappear. The algorithm first computes local parameters for smooth regions while exploiting the local straightness of iso-depth contours. Their method has not been empirically and analytically examined for sensitivity arising from their approximation.

3.2. Displacement Map Solutions

In early attempts to solve the displacement equations, [36] and [13] formulated solutions to systems of these equations that first eliminated depth and were transcendental in terms of the Euler angles of rotation and linear and homogeneous in terms of the translation parameters. With five point correspondences there are five equations in five unknowns (after setting a scale factor). These equations could be solved by iterative techniques but they are not amenable to proofs of uniqueness or the number of solutions.

In [20], the second-degree polynomial of [36] was employed, using quarternions to represent rotations. This technique was found to be very unstable.

[44] pursued an approach introduced in [23] for obtaining a linear method to solve the system of equations if more than eight displacement values are given. This is achieved by the formulation of an intermediate matrix E of nine parameters called the essential parameters. It was found that the computation was so sensitive that even 1% error in the displacements could lead to over 50% error in the recovered relative depth. The authors of [50] have since proposed nonlinear methods for solving these equations in order to cope with noise.

In [38] it was shown that there was a linear algorithm for estimating structure and motion using line correspondences. [51] proposed a similar algorithm.

[49] used collinear points to classify the normal curvature of the surface into convex, concave, and 3D-collinear labels; then, whole regions were classified with more complex labels such as hyperbolic, parabolic, and cylindrical. Rotating a 2D-collinear point finder in the image about a center point, it was shown that different surfaces produced different radial signatures in response to sequences of calculations (similar to those employed in our work) on sets of three points. The signatures for any of the nonplanar patches have specific points where the signature indicated the direction of the FOE. Thus, with a few such signatures, the FOE could be located from the intersection of directions. This work suffers from several problems. It requires the explicit search for particular types of patches; also, these have to have sufficient depth variation to indicate the direction of the FOE. It relies on local information and is susceptible to the problem that any noise in the flow field is amplified when second derivatives are computed locally. Additionally, shape computation is inherently a local computation and cannot exploit global information to overcome noise. The presence of independently moving objects can cause distorted signatures so that the FOE computation would be contaminated. Lastly, the collinear point constraint presented in our paper is for the instantaneous motion model and does not hold for the case of image motion from displacement. Thus, the derivation in [49] using displacement maps was inexact.

3.3. Orthographic Solutions

[46] proposed that orthographic projection was a far more suitable model for human perception of structure from motion, for a number of reasons. It was shown that four point correspondences over three views yield a unique solution to the motion and structure up to a reflection about the image plane. A nonlinear algorithm was also provided to accomplish the task.

[46] also pointed out that from two views with orthography there are an uncountably infinite number of solutions independent of how many point correspondences are given. The coefficient matrix has rank 3. The authors of [43] have recently shown that this rank principle can be exploited. The overdetermined system can be solved when more views than two are specified using a singular value decomposition to obtain a matrix describing scene structure and another describing the motion of the camera between frames. This approach needs many views (at least 50) of the same points.

In general, recently there has been a resurgence of work in 3-D motion related to parameter elimination and the discovery that solutions can be found in fewer dimensions than the original space of the problem. Our paper presents and analyzes a novel framework of this character. The central idea pursued here is that for the instantaneous model under perspective projection, collinear point structures can be employed to cancel scene rotation in an efficient, robust, and intuitive manner. The cancellation of rotation permits the FOE to be detected, after which independent motion can be detected, rotation can be computed robustly, and relative depth becomes available.

A description of previous work in detecting independently moving objects will be presented in the section on that topic, as it is more appropriate to present this after our framework is clear.

4. COMPUTING THE DIRECTION OF TRANSLATION

This section presents a theory for computing the instantaneous direction of translation for an observer moving with unrestricted motion. The theory is based on cancelling linear variation of the flow field, as this eliminates rotation, and allows the FOE to be located in an exact manner. In principle, once translation is known, the computation of observer rotation is a computation that simply involves inverting a linear system, hence this topic is not addressed here. In the next section, the work of this section is extended to include an indication of the regions within which there is independent motion. In this section, in addition to theory, an implementation is presented and studied analytically and empirically.

4.1. Assumptions about the Input

The input is assumed to be a 2-D instantaneous motion field. The field is dense in that at many points in the interior of the field there should be many (a theoretical minimum of two) line directions (in the image) through the point, with the additional requirement that there be sufficiently many (a theoretical minimum of three) optical flow measurements along each of these lines. The theoretical minima mentioned here will often be insufficient for certain patterns of scene depth variation, even in the case of a noise-free motion field. For now, the assumption is made that the entire motion field corresponds to a rigid scene. Another assumption made is that the relative motion is such that if there is translation, its FOE is retinotopically within the field of view represented by the motion field.

4.2. FOE from Collinear Measurements

Three image points are referred to as a *triplet.* It is first shown how a computation involving three *collinear* image velocity measurements relates 3-D translation to depth, by cancelling rotation. Then it is shown how this computation can be used to find the FOE. The computation is a generalization of an approximation to the second derivative of the velocity component that is normal to the line through the collinear triplet, the "derivative" being taken in the direction of the triplet line. The measurements from the collinear image triplet are assumed to be associated with three scene points on the *same* rigid structure, although not necessarily on the same surface nor on physically connected surfaces. For points subscripted by i, ($i = 1, 2, 3$), let the image coordinates of the points be (x_i, y_i), their image velocities be (u_i, v_i), and the depth values of their counterparts in the scene be Z_i.

This result is first derived for the simplified case where the three measurements are chosen at locations such that one point is at the origin of the image plane and the other two are at equal distances on either side of it on the y-axis of the image plane. Once the basic idea of cancellation of linear variation (and hence of rotation) is clear, the generalization to any triplet of collinear points in the image plane is shown.

4.3. The Simple Case along the *y*-Axis

Consider Fig. 2a, which depicts the projection of three scene points such that one lies at the image origin and the other two lie along the y-axis equidistant from the origin. Let these points have image coordinates at $(0, j)$, $(0, 0)$, and $(0, -j)$. Then their respective 3-D coordinates must be $(0, jZ_1, Z_1)$, $(0, 0, Z_2)$, and $(0, -jZ_3, Z_3)$ for some positive Z_1, Z_2, Z_3. Label the scene points as P_1, P_2, and P_3 and their image counterparts as p_1, p_2, and p_3. Then the projected velocity in the image plane at position (x, y) appears as (u, v), given by Eq. (1). For the three image points p_i, the x-components of the image velocities will be

$$u_1 = \frac{-U}{Z_1} + B - jC$$

$$u_2 = \frac{-U}{Z_2} + B$$

$$u_3 = \frac{-U}{Z_3} + B + jC.$$

Now taking the weighted sum (the finite difference approximation to the second derivative), $Sum \overset{\text{def}}{=} -u_1 + 2u_2 - u_3$, it can be seen to be

$$Sum = \frac{U}{Z_1} - \frac{2U}{Z_2} + \frac{U}{Z_3}$$
$$= \left(\frac{1}{Z_1} - \frac{2}{Z_2} + \frac{1}{Z_3}\right)(U),$$

i.e., $Sum = \alpha U$ where $\alpha = ((1/Z_1) - (2/Z_2) + (1/Z_3))$.

Now examine the sign of *Sum*. If *Sum* is zero, then if one assumes that U is not zero, α must be zero, that is,

$$\frac{1}{Z_1} + \frac{1}{Z_3} = \frac{2}{Z_2},$$

which means that the points must lie along a line in 3-space. If the calculated *Sum* is not zero, and *if* the sign of U is known, then one can infer the sign of α. But the sign of α indicates whether point P_2 lies in front of or behind the line-segment joining P_1 and P_3; that is, this tells us whether the points are in a convex relationship or a concave one.

The elimination of rotation is possible in this manner because rotation's contribution to the orthogonal component of instantaneous image motion, orthogonal to an image line, varies linearly along the line. Thus, taking the second derivative along the line cancels the rotation.

4.4. The Collinear Point Constraint (CPC)

The general constraint can be stated in a theorem. See Fig. 2b for intuition about this constraint.

THEOREM. *For a collinear image triplet consisting of points p_1, p_2, and p_3 in sequence, where m is the distance between the first and second image points and n the distance between the second and third, the weighted sum computation below produces a quantity from which the rotation terms have been eliminated,*

$$Sum \overset{\text{def}}{=} (-\sin\theta)(nu_1 - (m+n)u_2 + mu_3) + (\cos\theta)(nv_1 - (m+n)v_2 + mv_3), \tag{4}$$

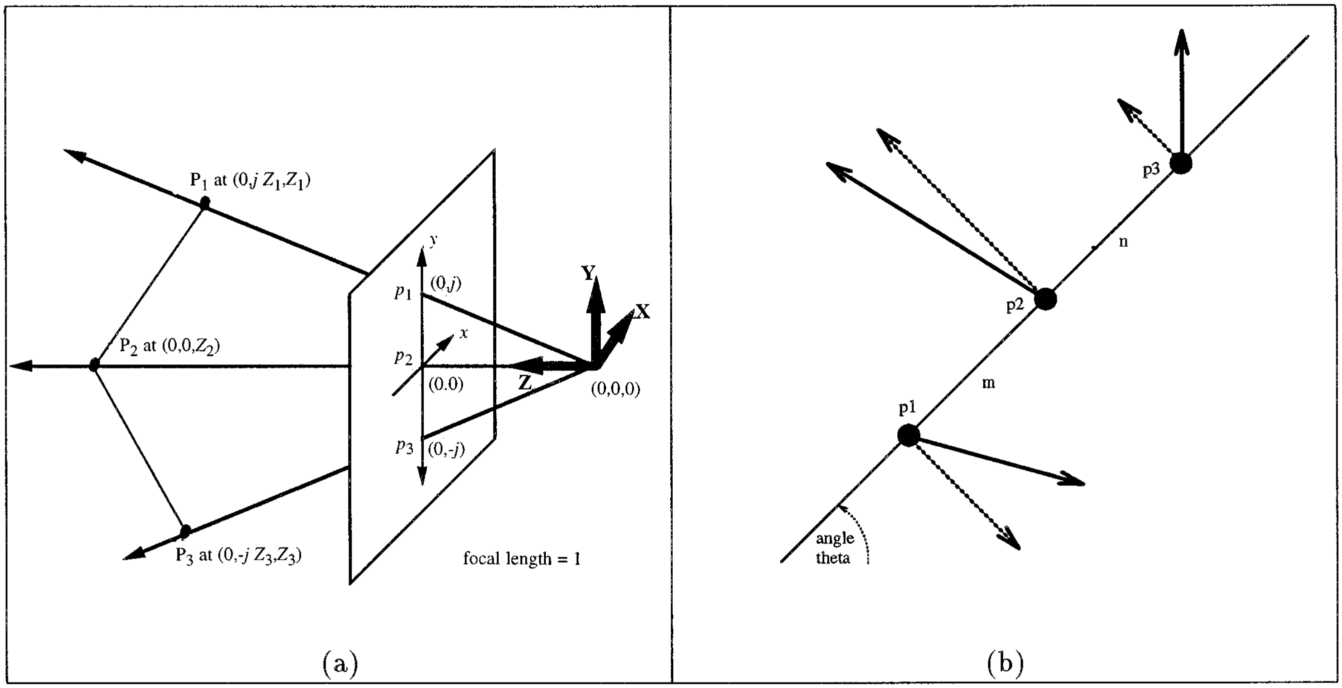


FIG. 2. (a) Points p_1, p_2, and p_3 at $(0, j)$, $(0, 0)$, and $(0, -j)$ in the image are projections of some scene points P_1, P_2, and P_3 that must have scene coordinates $(0, jZ_1, Z_1)$, $(0, 0, Z_2)$, and $(0, -jZ_3, Z_3)$, respectively, for some positive Z_1, Z_2, Z_3. For this example, κ_y, defined later as the difference between the gradients in Y of the two limbs, according to $(Z_1 - Z_2/Y_1 - Y_2) - (Z_2 - Z_3/Y_2 - Y_3)$, is $(Z_1 - Z_2/jZ_1) - (Z_2 - Z_3/jZ_3)$. (b) For any three collinear image flow measurements (the three solid arrows), their projections in the direction orthogonal to the line of collinearity (the three dotted arrows) must satisfy the following. If the three flow vectors were from a purely rotational field, then the tips of the projections would lie on a line. That is, the orthogonal projections of pure rotation vary linearly along any straight line. That is, the orthogonal projections can be cancelled by taking the discrete approximation to the second derivative. In the particular example here, then, it is clear that the three flow vectors are not due to a purely rigid rotational motion.

where θ is the angle that the line through the three image points makes with the image x-axis, going from the x-axis to the line (i.e., the components of flow used are those that are orthogonal to the line.) When the collinear points are equispaced, the weights are $(1, -2, 1)$, which is an approximation to the second derivative [35]. Other approximations to the second derivative, involving more points, are possible.

Proof. The p_i lie on a line with slope $\sin\theta/\cos\theta$ (in this proof, the case where $\cos\theta = 0$ is ignored, as in that case the verification is trivial). So, one can use the substitutions $y_1 = (\sin\theta/\cos\theta)(x_1 - x_3) + y_3$ and $y_2 = (\sin\theta/\cos\theta)(x_2 - x_3) + y_3$. Also, assume without loss of generality that the ordering of the points is such that $x_1 < x_2 < x_3$. Then, one can use $x_3 = x_2 - n\cos\theta$ and $x_2 = x_1 - m\cos\theta$ to obtain the result that the generalized *Sum* is

$$Sum = (U\sin\theta - V\cos\theta + Wy_3\cos\theta - Wx_3\sin\theta) \times \left(\frac{n}{Z_1} - \frac{(n+m)}{Z_2} + \frac{m}{Z_3}\right), \tag{5}$$

i.e., *Sum* is the product of a translation factor and a depth factor. ∎

4.5. Properties of the Depth Factor

Consider the general collinear triplet. Then the image points p_1 at (x_1, y_1), p_2 at (x_2, y_2), p_3 at (x_3, y_3) are associated with scene points labelled P_1, P_2, P_3, lying at (x_1Z_1, y_1Z_1, Z_1), (x_2Z_2, y_2Z_2, Z_2), (x_3Z_3, y_3Z_3, Z_3) for some positive Z_1, Z_2, Z_3. So, if the distance between p_1 and p_2 is m, and the distance between p_2 and p_3 is n, then $x_1 = x_2 + m\cos\theta$, $y_1 = y_2 + m\sin\theta$, and $x_3 = x_2 - n\cos\theta$, $y_3 = y_2 - n\sin\theta$, and so the 3-D points lie at $((x_2 + m\cos\theta)Z_1, (y_2 + m\sin\theta)Z_1, Z_1)$, (x_2, y_2Z_2, Z_2), and $((x_2 - n\cos\theta)Z_3, (y_2 - n\sin\theta)Z_3, Z_3)$. Consider the gradient of the line-segment $\overline{P_1P_2}$ and of the line-segment $\overline{P_2P_3}$, where gradient is defined as $(\Delta Z/\Delta X, \Delta Z/\Delta Y)$. Then, the following theorem provides an interpretation for the depth factor, relating it to the difference in gradients of the two limbs, to the depths of the three points, and to the distances between the scene points. This theorem will provide the tools necessary for obtaining a qualitative interpretation of the depth factor and the analytical power needed to describe the robustness of computations presented later.

THEOREM. *Assuming that θ is neither zero nor $\pi/2$, let the difference between the gradients in Y of the two limbs be κ_y. Then, the following is true*:

$$-\frac{n}{Z_1}+\frac{m+n}{Z_2}-\frac{m}{Z_3} = \kappa_y \frac{(y_2Z_2-(y_2-n\sin\theta)Z_3)((y_2+m\sin\theta)Z_1-y_2Z_2)}{\sin\theta Z_1Z_2Z_3}$$

Alternatively, this can be written as

$$-\frac{n}{Z_1}+\frac{m+n}{Z_2}-\frac{m}{Z_3}=\kappa_y\frac{(Y_2-Y_3)(Y_1-Y_2)}{\sin\theta Z_1Z_2Z_3}. \quad (6)$$

Proof. From the definition of κ_y,

$$\frac{Z_1-Z_2}{(y_2+m\sin\theta)Z_1-y_2Z_2}-\frac{Z_2-Z_3}{y_2Z_2-(y_2-n\sin\theta)Z_3}=\kappa_y.$$

Thus

$$\begin{aligned}&(Z_1-Z_2)(y_2Z_2-(y_2-n\sin\theta)Z_3)\\&\quad-(Z_2-Z_3)((y_2+m\sin\theta)Z_1-y_2Z_2)\\&\quad=(y_2Z_2-(y_2-n\sin\theta)Z_3)((y_2+m\sin\theta)Z_1-y_2Z_2)\kappa_y,\end{aligned}$$

which on simplification gives

$$\begin{aligned}&\sin\theta(nZ_1Z_3-nZ_2Z_3-mZ_1Z_2+mZ_1Z_3)\\&\quad=(y_2Z_2-(y_2-n\sin\theta)Z_3)((y_2+m\sin\theta)Z_1-y_2Z_2)\kappa_y.\end{aligned}$$

Dividing through by $\sin\theta Z_1Z_2Z_3$ (assuming that $\sin\theta$ is not zero), gives

$$-\frac{n}{Z_1}+\frac{m+n}{Z_2}-\frac{m}{Z_3} = \kappa_y \frac{(y_2Z_2-(y_2-n\sin\theta)Z_3)((y_2+m\sin\theta)Z_1-y_2Z_2)}{\sin\theta Z_1Z_2Z_3}$$

to complete the proof. ■

Since θ is neither zero nor $\pi/2$, the right-hand side of Eq. (6) can only be zero if κ_y is zero, i.e., the gradients of the two limbs have to be identical. Hence, the left-hand side is zero only if κ_y is zero. Since $\kappa_x = \kappa_y(\Delta Y/\Delta X)$, the left-hand side is zero only if the three points are collinear in the scene.

Similarly, the conditions for the three scene points lying in a convex or concave relationship can be verified by the following. $Z_2 = (m + n)Z_1Z_3/(mZ_1 + nZ_3)$ is equivalent to $n/Z_1 - (m + n)/Z_2 + m/Z_3 = 0$, which has just been shown, implies that P_2 lies on the line segment $\overline{P_1P_3}$. Hence $Z_2 < (m + n)Z_1Z_3/(mZ_1 + nZ_3)$ must mean that P_2 is in front of the line-segment $\overline{P_1P_3}$ (from the point of view of the observer at the origin.) That is, when $n/Z_1 - (m + n/Z_2 + m/Z_3 < 0$, the points are in a convex relationship. Similarly, $Z_2 > (m + n)Z_1Z_3/(mZ_1 + nZ_3)$ (i.e., $n/Z_1 - (m + n)/Z_2 + m/Z_3 > 0$) must imply that P_2 is behind the line-segment $\overline{P_1P_3}$, i.e., that the points are in a concave relationship.

In addition, for a nonzero difference in slopes (in Y) of κ_y, the measured quantity's contribution from the depth factor is $\kappa_y\ (y_2Z_2 - (y_2 - n \sin\theta)Z_3)((y_2 + m\sin\theta) Z_1 - y_2Z_2)/\sin\theta Z_1Z_2Z_3$, and this gives an expression for analyzing the effects of noise.

4.6. Collinear Triplets and FOE

The following Lemma provides the result needed for detecting the FOE.

LEMMA. *The zero conditions of Sum are the following: the points in the scene are collinear, or there is no translation component, or the triplet line intersects the FOE.*

Proof. *Sum* can be seen to be zero when either the depth factor is zero (in which case the scene points are collinear as shown above) or the translation factor is zero. The translation factor is an inner product of $(U, V, W)^T$ and $(\sin\theta, -\cos\theta, y_3\cos\theta - x_3\sin\theta)^T$. Besides the case when the translation (U, V, W) itself is zero, this translation factor is zero only when the line passing through the three image points also passes through the FOE. To see this, equate the translation factor to zero, then substitute into the translation factor of Eq. (5) $U = Wx_f$, $V = Wy_f$ (by definition of FOE), and observe that $y_3 = (\sin\theta/\cos\theta)(x_3 - x_f) + y_f$ must follow, which implies that (x_f, y_f) lies on the triplet line. ■

Thus, in general, for a scene containing depth variation, if many such collinear triplet sums across the complete image are computed, the FOE will be in the position of intersection of many triplet lines for which, in each case, the triplet sum is zero. That is, the location that has the least counterevidence for being the location of the FOE is being sought. Next an infinitesimal version of the result is provided to improve insight, and subsequently an operator and an algorithm to detect the FOE are described.

4.7. A Spatially Infinitesimal Interpretation of the CPC

In this section, a spatially continuous version of the CPC for equispaced points is presented to aid intuition. Let $\vec{v}(x, y) = (u(x, y), v(x, y))^T$ be the continuous image velocity vector field. Then *Sum* is defined as

$$Sum = \frac{\partial^2}{\partial \vec{s}^2}(\vec{v}\cdot\vec{n}),$$

where $\vec{s}$ is the direction along which the second derivative is taken and $\vec{n}$ is the normal to $\vec{s}$. $\vec{s}$ is given by $(\cos\theta, \sin\theta)^{\mathrm{T}}$ and $\vec{n}$ is given by $(-\sin\theta, \cos\theta)^{\mathrm{T}}$. Then, using

$$\frac{\partial_2 f}{\partial \vec{s}^2} = \nabla(\nabla f\cdot\vec{s})\cdot\vec{s}$$

gives

$$\begin{aligned} Sum = {} & -u_{xx}\sin\theta\cos^2\theta + v_{xx}\cos^3\theta \\ & - u_{yy}\sin^3\theta v_{yy}\sin^2\theta\cos\theta \\ & -2u_{xy}\sin^2\theta\cos\theta + 2v_{xy}\sin\theta\cos^2\theta. \end{aligned}$$

Now,

$$\begin{aligned} u_{xx} &= (-U + xW)\frac{\partial^2}{\partial^2 x}\left(\frac{1}{Z}\right) + 2W\frac{\partial}{\partial x}\left(\frac{1}{Z}\right) + 2B \\ u_{yy} &= (-U + xW)\frac{\partial^2}{\partial^2 y}\left(\frac{1}{Z}\right) \\ v_{xx} &= (-V + yW)\frac{\partial^2}{\partial^2 x}\left(\frac{1}{Z}\right) \\ v_{yy} &= (-\mathrm{V} + yW)\frac{\partial^2}{\partial^2 y}\left(\frac{1}{Z}\right) + 2W\frac{\partial}{\partial y}\left(\frac{1}{Z}\right) - 2A \\ v_{xy} &= (-V + yW)\frac{\partial^2}{\partial x\partial y}\left(\frac{1}{Z}\right) + W\frac{\partial}{\partial x}\left(\frac{1}{Z}\right) + B \\ u_{xy} &= (-U + xW)\frac{\partial^2}{\partial x\partial y}\left(\frac{1}{Z}\right) + W\frac{\partial}{\partial y}\left(\frac{1}{Z}\right) - A, \end{aligned}$$

yielding

$$\begin{aligned} Sum = {} & -\frac{\partial^2}{\partial x^2}\left(\frac{1}{Z}\right)(-U + xW)\sin\theta\cos^2\theta \\ & -\frac{\partial^2}{\partial y^2}\left(\frac{1}{Z}\right)(-U + xW)\sin^3\theta \\ & +\frac{\partial^2}{\partial y^2}\left(\frac{1}{Z}\right)(-V + yW)\sin^2\theta\cos\theta \\ & +\frac{\partial^2}{\partial x^2}\left(\frac{1}{Z}\right)(-V + yW)\cos^3\theta \\ & -2\frac{\partial^2}{\partial x\partial y}\left(\frac{1}{Z}\right)(-U + xW)\sin^2\theta\cos\theta \\ & +2\frac{\partial^2}{\partial x\partial y}\left(\frac{1}{Z}\right)(-V + yW)\sin\theta\cos^2\theta, \end{aligned}$$

which on rearranging can be written as

$$\begin{aligned} Sum &= ((-U + xW, -V + yW)^{\mathrm{T}}\cdot\vec{n})\left(\nabla\left(\nabla\frac{1}{Z}\cdot\vec{s}\right)\cdot\vec{s}\right) \\ &= ((-U + xW, -V + yW)^{\mathrm{T}}\cdot\vec{n})\frac{\partial^2}{\partial\vec{s}^2}\left(\frac{1}{Z}\right), \end{aligned}$$

which is the analogue of the discretized Eq. (5) for equispaced points. Next, an operator and an algorithm to detect the FOE are described.

4.8. The FOE Operator

The FOE algorithm is implemented in the following manner. Consider an operator (called the *FOE operator*) with many lines passing through its center (centered at some (x, y) position in the image) in many directions in the image (see Fig. 3a), such that each line passes through several image points. For a scene with sufficient depth variation so that randomly sampling along only two lines will not sample at coplanar points, the theoretical minimum needed is two lines (because the FOE will be at the intersection of these two lines). The theoretical minimum number of points needed along a line is three for the same stringent conditions of not sampling at coplanar points. In practice, such as for the implementation presented later, between 50 and 250 points are used along each line, the exact number depending on how many grid points the line intersects. In the implementation to be described later, 16 lines were used. Along such a line, overlapping triplets of image velocity are used and summed (see Fig. 3b). Within the set of points along the line, the *within-triplet spacing, S* (i.e., how far apart the points within each triplet are), and the *between-triplet spacing, s* (i.e., how far apart the centers of adjacent triplets should be), parameters need to be chosen.

The absolute values of all triplet sums (as defined by Eq. (4)) along a line are summed to give a *LineSum*, and then the *LineSums* are summed to get a *Response* at the center (x_0, y_0), given by

$$Response(x_0, y_0) = \frac{\sum_{\theta_k}\sum_{s_j}|Sum_{S,\theta_k}|}{TotalTriplets},$$

where *TotalTriplets* is the total number of triplets used for computing the response at (x_0, y_0), where

$$Sum_{S,\theta_k} = [m \;\; -(m+n) \;\; n]\begin{bmatrix} u_{-1} & v_{-1} \\ u_0 & v_0 \\ u_1 & v_1 \end{bmatrix}\begin{bmatrix} -\sin\theta_k \\ \cos\theta_k \end{bmatrix},$$

and where

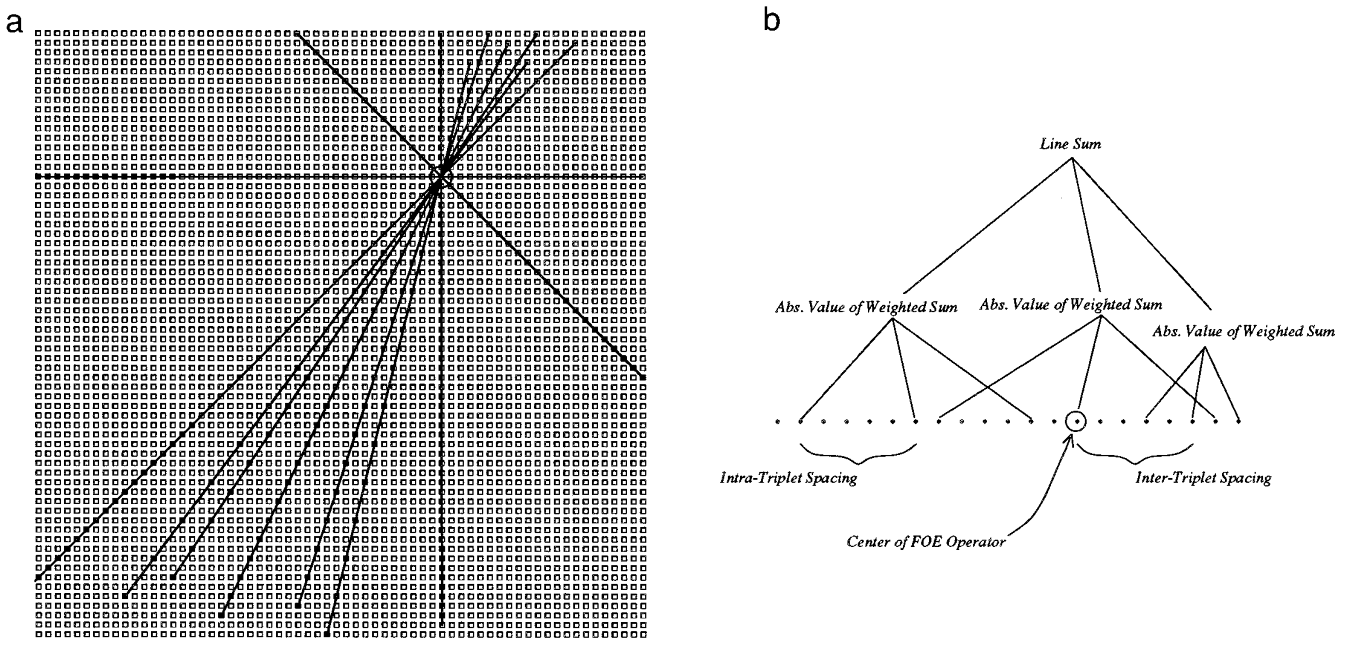

a
b
Line Sum
Abs. Value of Weighted Sum
Abs. Value of Weighted Sum
Abs. Value of Weighted Sum
Intra-Triplet Spacing
Inter-Triplet Spacing
Center of FOE Operator

FIG. 3. (a) The FOE operator. This shows how an operator is made up of intersecting lines of points in the image. On a regular 256 × 256 square grid of points, at a single point it is possible to have about 60–70 intersecting lines, each of which passes through at least several image points. Here, only nine lines are shown. The image is a regular gird of points, shown as hollow squares, each with a flow estimate associated with it. Some of the points used by each line have been blackened to identify them. Along each line a *LineSum* is calculated (see Fig. 3b). The *LineSums* are added to give the response of the operator at position (x, y), which is where the lines intersect. (b) This shows how points on a line are processed. Flow estimates at points are grouped into triplets and summed according to the weighted sum specified in Eq. (4). Then the absolute values of these weighted sums are summed to give the *LineSum* for the line. For now, discussions of the optimal selection of *within-triplet spacing,* i.e., how far apart the points within each triplet should be, and the optimal *between-triplet spacing,* i.e., how far apart the centers of adjacent triplets should be, are postponed.

$$u_i = u(x_0 + (s_j + iS)(\cos\theta_k), y_0 + (s_j + iS)(\sin\theta_k))$$
$$v_i = v(x_0 + (s_j + iS)(\cos\theta_k), y_0 + (s_j + iS)(\sin\theta_k)).$$

For a rigid scene, there are three reasons why the response could be zero. First, each of the triplets summed by this operator could be a collinear triplet in the scene; however, along each line the triplets overlap, thus implying that for this situation to occur the whole scene would have to be a single plane, which for typical scenes is a rare occurrence. Second, there could be no translation. Third, the true FOE could be at the position of the center of the operator. Fortunately, the former two are distinguishable from the third by the fact that if either of the two were to hold, the operator responds with zero everywhere; whereas, in the third case, assuming the first two do not hold, there is a unique zero. The first two cases can be detected easily and eliminated. Thus, the FOE operator can be swept across the whole image (i.e., the whole flow field) to obtain a response map, detecting where it gives a zero response surrounded by nonzero responses that are above some prespecified threshold. Note that if the operator were not designed with overlapping triplets in each line and with triplets that straddle the operator center, the operator would give a zero response at the peak of a single conical shape that spanned the whole scene.

The response map can obviously be computed by operators in parallel, after choosing an appropriate *between-operator spacing*. Then the operator with the zero response must be found. If the between-operator spacing is large (i.e., if the response is coarsely sampled), the responses are interpolated to find a zero (see subsection 4.9.2). In practice, due to noise in the input flow field there may not be a zero in the response map, but there will be a global minimum. In the case where the FOE lies in the image, and where the noise is uncorrelated and not excessive,[1] the global minimum typically corresponds to the FOE.

For the implementation of the FOE algorithm in the upcoming section and in later sections, unless otherwise specified, the following parameters were used for the FOE operator. The FOE operator was centered at each node of a 256 × 256 grid. Each FOE operator consisted of 16 lines passing through it (see Fig. 3). Along each line, a

[1] Empirically, as described later, it was found that the mean of the Gaussian distribution of the percentage of added (for additive) noise should not exceed about 8%.

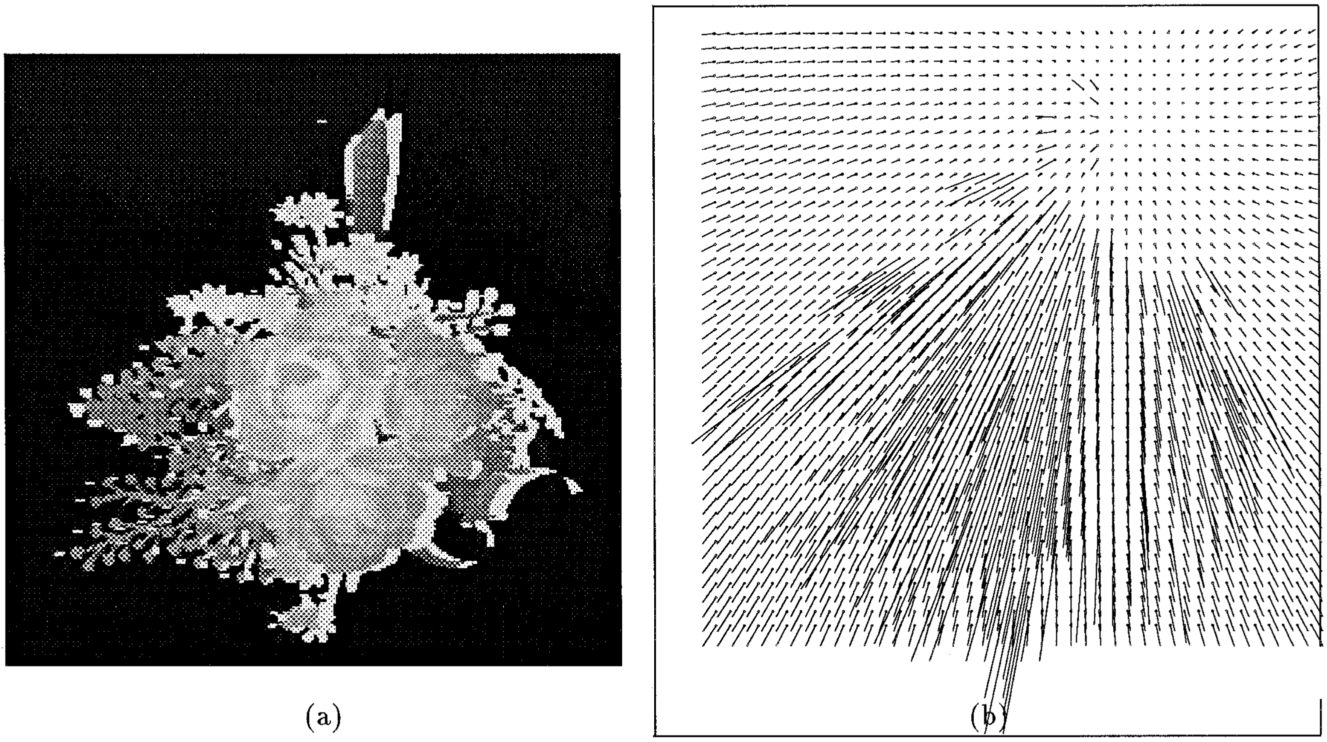


FIG. 4. (a) Range data that were used to generate flow field used in these experiments. Observer moves with some instantaneous 3-D rotation and translation parameters with respect to these depth points and the flow equations are used to give an image velocity vector at each image point. (b) Flow field generated using 3-D motion and the range data shown earlier. The rotation used was (0.004, 0.003, 0.004) and the translation used was (4.5, 8.5, 10.0). Field shown subsampled.

triplet was centered at each point on that line, so that the between-triplet spacing, s, is one unit. Note that the spacing of points on a line depends on the slope of the line. See Fig. 3a and inspect the spacing between the blackened pixels for the different lines. Within each triplet, the within-triplet spacing, S, was such that three alternate points on the line were used (i.e., a triplet spanned five pixels).

4.9. Empirical and Analytical Findings

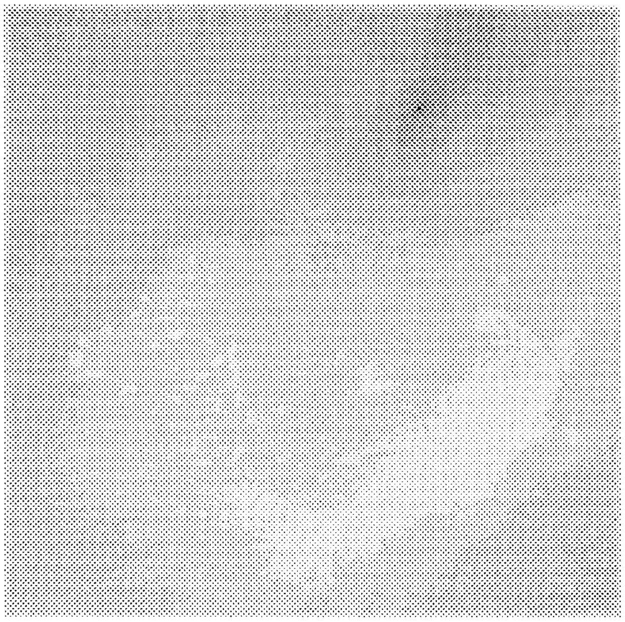
In this section, the FOE algorithm is tested on a variety of synthetic input under controlled conditions. Several factors influencing its robustness are studied.

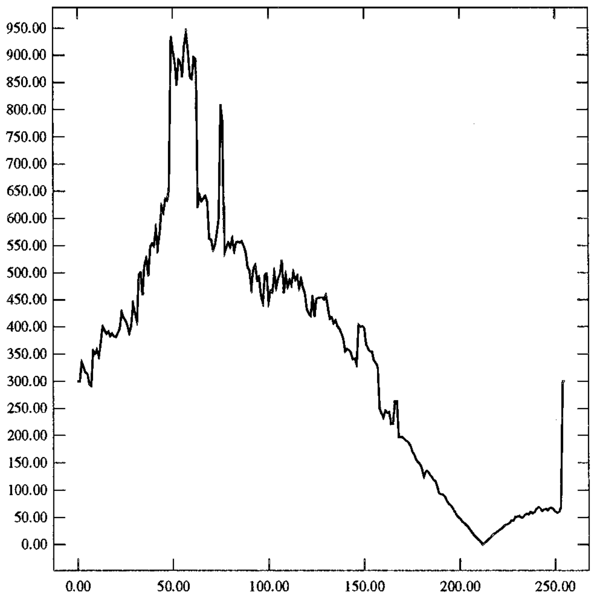
4.9.1. *With Ideal Data*

The range image shown in Fig. 4a was used to generate the synthetic flow field shown in Fig. 4b. This image contains the kind of depth variation that a perceiver may encounter in typical situations of navigation where there are obstacles, or where the scene itself is naturally not flat. The manner in which this data was used to synthesize flow was the following. The field of view spanned horizontally and vertically by this data is simulated to be an angle of 2 Arctan (1.28), which is approximately 105° visual arc, and diagonally (from upper left to bottom right) to be an angle of 2 Arctan (1.28 $\sqrt{2}$), which is approximately 122° visual arc. The image grid used here simulates 1/100 focal length units (FLUs) between each grid point horizontally (and vertically), while diagonally the grid points are separated by $\sqrt{2}/100$ FLUs. Thus near the image origin, the horizontal space between grid points subtends about 0.5° visual arc, while at the border of the image this horizontal space subtends about 0.2° visual arc. Diagonally, these parameters are 0.5 $\sqrt{2}$° visual arc and about $\sqrt{2} \times 0.2$° visual arc.

Scene depth simulated by this data puts the closest point at a Z-value of 60 FLUs and the farthest points (the range image's background plane) at 600 FLUs. This represents the depth variation available when an observer moves relative to an object that is close up against a background, with the background Z-value 10 times away.

With this scenario and the motions used here, if these were to correspond to a real motion sequence, completely dense flows could not be obtained: occlusion/disocclusion


(a)



(b)

FIG. 5. (a) Response map for noise-free flow. In these maps, brightness is shown proportional to *Log*(*Response*). For this input, the global minimum is exactly at the true FOE. (b) 1-D slice passing through the detected minimum of this map.

would leave certain pieces of the scene without sufficient information to compute flow. Fortunately, this behavior is confined to small portions of the scene. Thus, for the purposes of the experiments here, these regions of occlusion/disocclusion are ignored and flow is synthesized everywhere. This shortcoming is compensated for by using a flow model with noise which can emulate the behavior occurring at these regions.

The translation and rotation parameters used to generate this flow field were $(U, V, W) = (4.5, 8.5, 10.0)$ and $(A, B, C) = (0.004, 0.003, 0.004)$. Using Eqs. (1), the field acquired flow values at every pixel position. It was used as input to the FOE algorithm and the FOE was computed.

Figure 5a shows the response map for the operator as a function of image location (x, y). All response maps are shown with brightness proportional to the *Log* (*Response*). The darkest point in the map is the global minimum; this corresponds to the computed FOE and it is not surprising that it is exactly correct.

Figure 5b shows the profile of a one-dimensional vertical slice through the response map passing through the minimum.

In this case, the direction of translation had been simulated to have an FOE positioned at exactly one of the grid points. Given that the between-operator spacing was aligned with that of the grid points, the true FOE was located at an FOE operator sample point, hence the exact position of the minimum is found. When the true FOE lies in between sampling positions, the minimum is found by interpolation as described next.

4.9.2. *Interpolation to find Minimum in Response Map*

A quadratic is fit around the located minimum sample point and its two closest neighbors horizontally (and the same is repeated vertically).

$$r(x) = ax^2 + bx + c,$$

where

$$a = \frac{1}{2}(r_1 + r_{-1}) - r_0 \quad b = \frac{1}{2}(r_1 - r_{-1}) \quad c = r_0.$$

So, taking the derivative and setting it to zero,

$$x_{\min} = -b/2a$$
$$r(x_{\min}) = c - b^2/4a.$$

A similar computation gives the estimated minimum along the y-dimension. The minimum is taken to be the position specified by $(x_{\min}, y_{\min})$. An alternative method of interpolation would be to use a full quadratic $ax^2 + bxy + cy^2 + dx + ey + f$ on six points in a neighborhood.

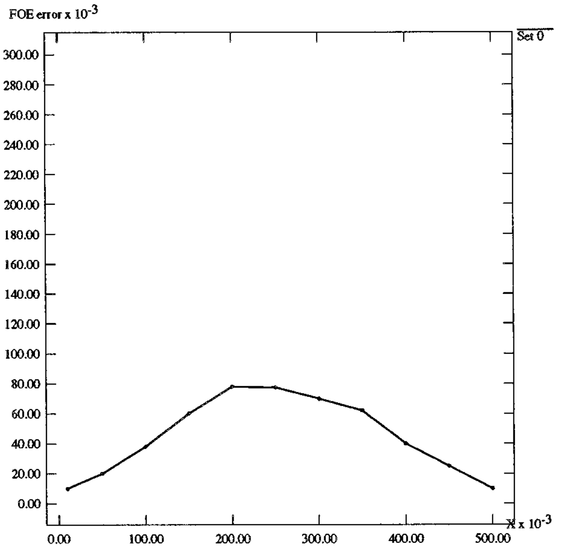


FIG. 6. FOE error (due to interpolation) versus distance of true FOE from closest sample point. FOE error is in units of degree of visual arc.

When the true FOE does not coincide with an operator sample point, there is some error. This is captured in the plot of Fig. 6 showing error versus distance of true FOE from its closest sample point, which for the ideal data used is none other than the grid point identified as the *Response's* minimum sample point. (The FOE error for all upcoming plots are in degrees of visual arc.) The worst FOE error due to interpolation for this range data is 0.08° visual arc, and this is when the true FOE is about 0.2° visual arc (about midway between two grid points). These results are stable for even 1000-fold increases in the magnitude of rotation.

For example, Figs. 7a, 7b, and 7c show the initial range data used for flow synthesis, the synthesized flow, and the response map. The closest point is about 60 FLUs away and the furthest is about 300 FLUs away from the optical center of the observer. The 3-D motion parameters used to synthesize flow were $[U, V, W] = [-5.32, 1.61, 10.0]$ and $[A, B, C] = [5.0, 8.1, 3.6]$, and the recovered FOE is correct to within 0.01° visual arc of the true FOE. Note that the rotation used to synthesize this data is very large. This insensitivity to the magnitude of rotation is not surprising, because for this error-free data the rotation is exactly cancelled.

4.9.3. *Model of Noise Used to Simulate Realistic Data*

To better simulate a realistic flow field, noise is introduced into the ideal data. The model of noise used has a Gaussian probability density function, but its mean is chosen not at zero but rather at some nonzero real number. Thus, if the real number is, say, 8.0, then if the standard deviation of the density function is, say, 2.0, the noisy flow estimate is generated according to

$$\vec{v}_{\text{noisy}} = \begin{bmatrix} u + sign_1 * N(8.0, 2.0) * 0.01 * u \\ v + sign_2 * N(8.0, 2.0) * 0.01 * v \end{bmatrix}, \tag{7}$$

where $sign_i$ is a binary function that is randomly chosen with equal probability and $N(a, b)$ is a Gaussianly chosen random number with mean a and standard deviation b. Throughout the paper, this noise model is referred to with the phrase "Gaussian noise with mean a% and $\sigma = b$%." While an annular error distribution could have been used, instead of selecting four portions of the annular region as our error model does, in practice we found that this choice has insignificant effects on our results. What is far more important is the magnitude of the mean of the error distribution.

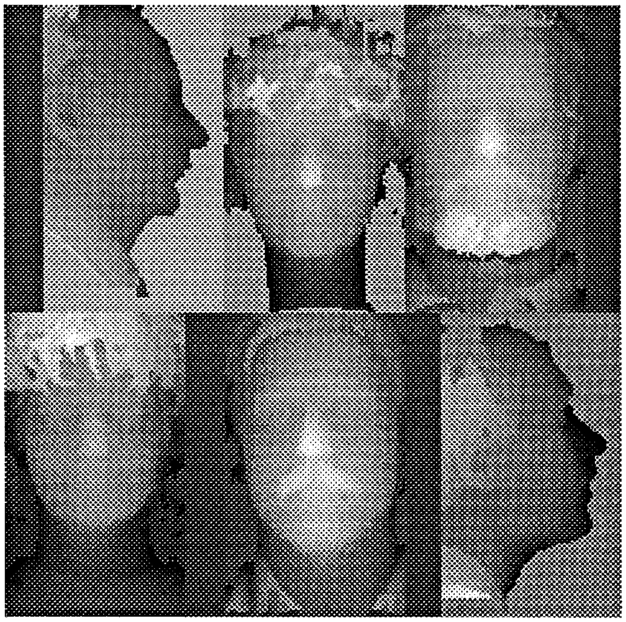
This error model provides the ability to synthesize errors whose range is similar to that produced by an optical flow technique for a dense field. See [3] for a review of current optical flow techniques. Typical optical flow techniques produce results with an error distribution that have a substantial number of outliers. The parameters actually used for the error models adopted in the experiments with synthesized data results in less error than would be practically computed by currently available techniques with real data [3].

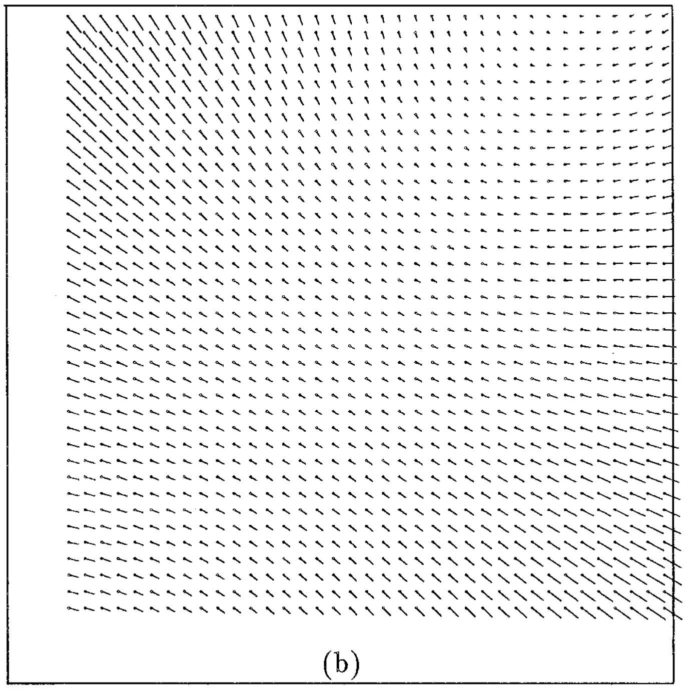
4.9.4. *With 8% Mean Gaussian Noise*

The ideal flow field was made more realistic by adding random Gaussian noise with mean 8% and standard deviation 2% of each of the image velocity estimates (as described in Section 4.9.3). A good approximation to the correct direction of translation was still found. The response map is shown in Fig. 8a. Figure 8b shows a one dimensional vertical slice of the response map through the minimum. The FOE error is 0.227° visual arc.

When noise is included, the error in the response minimum was found to increase with the magnitude of rotation. Figure 9a plots this behavior for flow data where across experiments the mean of the percentage of noise is kept fixed at 8%, and the translation is fixed, while rotation is varied in magnitude. The effects observed here have to do with the fact that noisy rotation introduces residual effects that are not cancelled and that affect the rest of the computation. On the other hand, changes in translational magnitude are more robust to noise. See Fig. 9b, showing the results for experiments in which the mean of the Gaussian noise was kept fixed at 15%.

These patterns of behavior are consistent across a variety


(a)


(b)

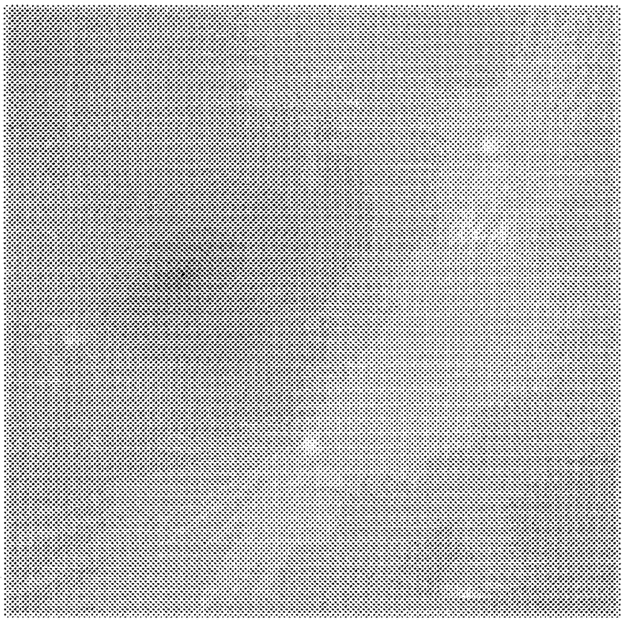

(c)

FIG. 7. (a) Another range image used to synthesize flow for these experiments. (b) Flow synthesized from motion parameters $[U, V, W] = [-5.32, 1.61, 10.0]$ and $[A, B, C] = [5.0, 8.1, 3.6]$, and the range image of faces. Note that since rotation is quite large, it dominates the flow, so that on superficial inspection of the flow, the underlying depth structure is not apparent. (c) Response map for the flow data synthesized from the faces range-image.

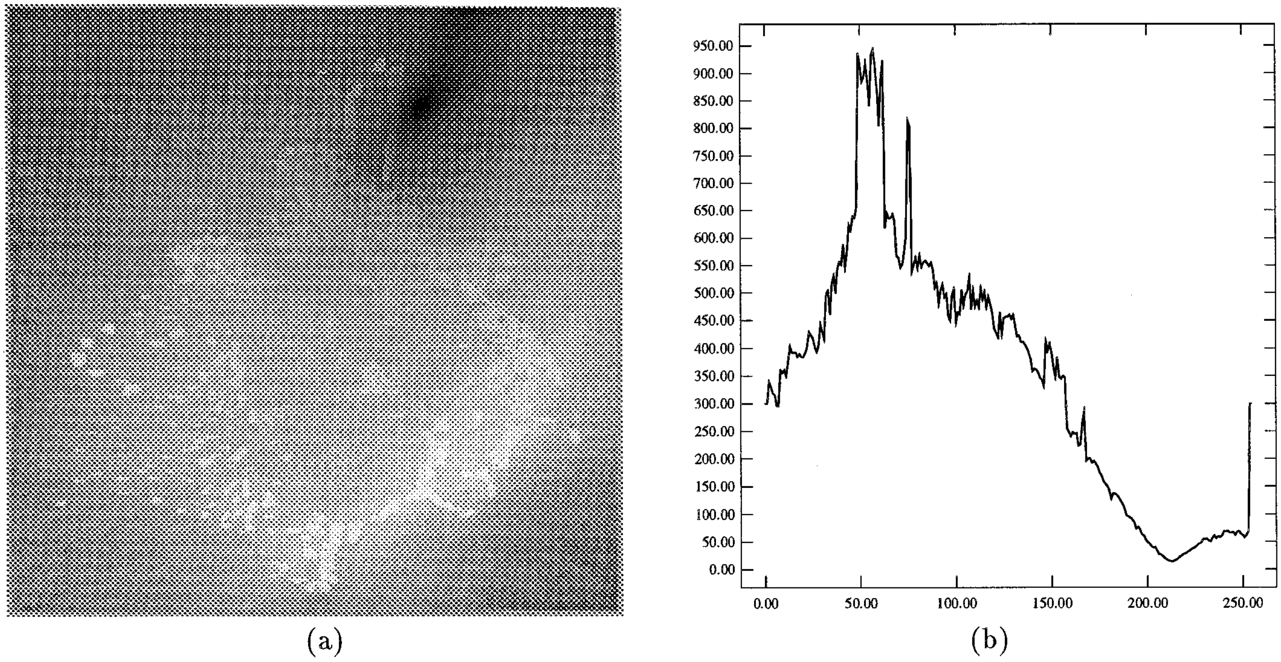


FIG. 8. (a) Response map for flow field with 8% mean Gaussian noise. (b) 1-D vertical slice through it.

of range data profiles[2] with the depth variation kept the same as in the first experiment.

An overview of the behavior with increasing noise is seen from Fig. 10, where the errors are averaged over three range data profiles with the same operator parameters and scene parameters as the experiment with ideal data described above. The mean of the Gaussian noise in the input flow is increased to 12% in steps of 1%.

4.9.5. *On a Planar Scene or when Motion is Purely Rotational*

On a planar scene with 4% noise in flow, the height plot of the response map is presented in Fig. 11a.

The response map is mostly flat, with the response everywhere being near zero. This behavior is robust to slants, tilts, and different combinations of small rotations and arbitrary translations. However, planes of steep slant or tilt (for example, those parallel to or near the ground plane) will typically have a horizon in the scene, thus not providing a complete flow field. Hence the behavior of the algorithm with these cases was not confirmed with experimentation.

For a case where motion is only rotational and noisy (1%), the height plot of the response map is presented in Fig. 11b. The flat response is robust to any depth profile (trivially, since depth has no influence on the flow) and choice of rotation axis. However, as observed earlier, increasing the noise or using larger rotations with moderate noise causes rotation not to be exactly eliminated and the response map is no longer flat and near-zero. Note that for two transparent planar surfaces (such as a picket fence in front of the wall of a house), if the planar surfaces are sufficiently separated in depth, there is enough depth variation to yield a solution to the moving observer's FOE computation.

When observer motion is purely rotational, the response map is indistinguishable from that for a scene composed of a single plane. An algorithm for detecting the behavior in these two cases involves taking FOE operator responses at several points, well-spaced apart, and confirming that some very large percentage of them are below a threshold. The algorithm used in this paper uses 16 FOE operator responses in a well-spaced 4×4 grid. The threshold is chosen arbitrarily, and 75% of the responses must be below this threshold in order to be classified in this category of planar or purely rotational. These numbers were chosen to keep the computation efficient, while still providing enough confidence in the result of the computation.

4.9.6. *Effects of Depth Variation*

The depth factor in Eq. (6) controls whether a triplet has a significant nonzero *Sum* when the image triplet line is unaligned with the line from the triplet center to the FOE (i.e., an FOE field line). This, in turn, influences the ability of the FOE operator to accumulate counterevidence for wrong FOE candidates. As seen, with scenes involving

[2] Other profiles used to synthesize flow were an image of some blocks and an image containing human faces.

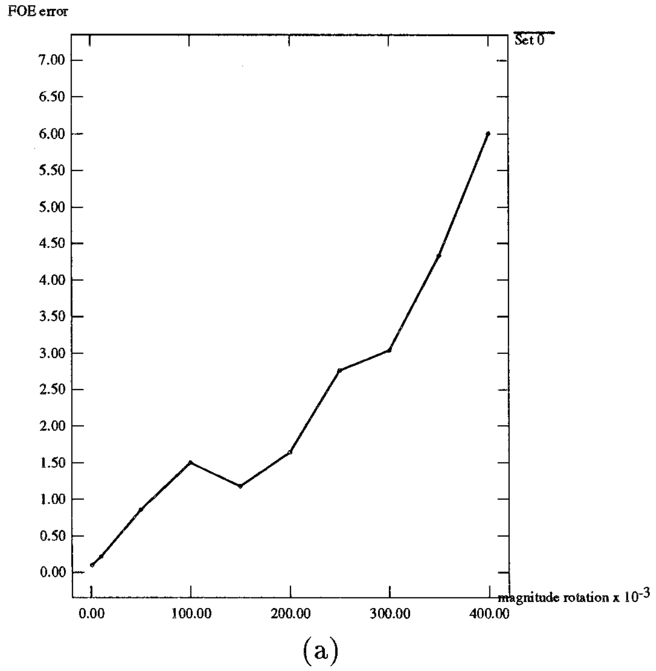


FIG. 9. (a) FOE error versus magnitude of rotation (for noisy data). (b) FOE error versus magnitude of translation. Noise in input was Gaussian with mean 15%. (This is a lot of error.) For lower noise levels, for example, Gaussian with mean 8%, the FOE error has a slight decrease with increasing translational magnitude.

single planes this approach will not provide an FOE solution or even a finite set of candidate solutions (which distinguishes it from an approach such as that of [11]), while with the depth variation of the first experiment described

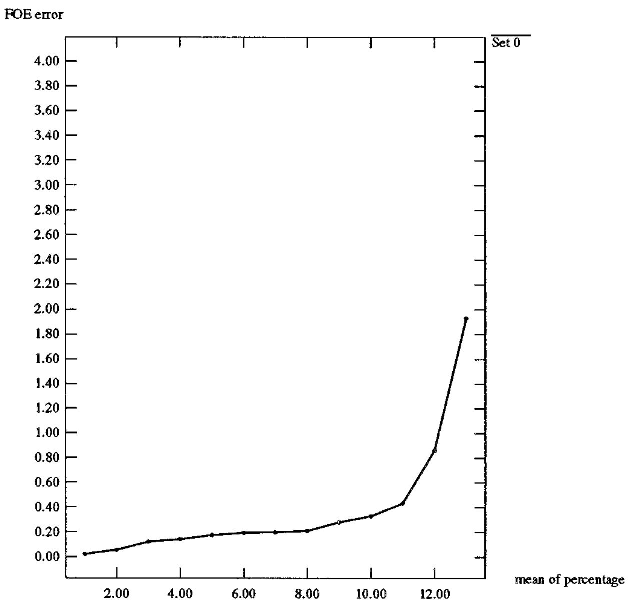


FIG. 10. Averaged results showing FOE error versus mean of the random Gaussian percentage error distribution.

earlier, a unique solution is found. Clearly, it is to be expected that as depth variation decreases, performance declines. This can be understood by examining Eq. (6). Inspecting it shows that the depth factor on the left-hand side is influenced by two relevant factors. First, there is the influence of the difference in gradients, $\vec{\kappa}$, of the two limbs composing the chevron[3]; second, for a fixed chevron, moving it away from the observer increases the product $Z_1Z_2Z_3$ while not changing the numerator $(Y_1 - Y_2)$ $(Y_2 - Y_3)$ and hence decreases the depth factor. The decrease in the depth factor is only partially due to the trivial decrease in m and n that occurs as a result of moving the chevron away. A substantial part of the decrease is due to the relative changes within the three Z_i. Thus, to obtain a high depth factor, it is necessary that the image triplets are sampling scene points in highly curved (away from collinear) chevron-shaped formations and also that these structures not be too distant from the viewer. In the presence of noise, a high depth factor will increase the robustness to noise.

In the seminal paper [12], it has been shown that there exists a unique solution to the egomotion problem if the scene has any more structure than a hyperboloid of one sheet. However, as seen in Fig. 12, a far greater amount

[3] The term chevron is used here to describe the structure formed by the three scene points which project collinearly to the image (see Fig. 2a).

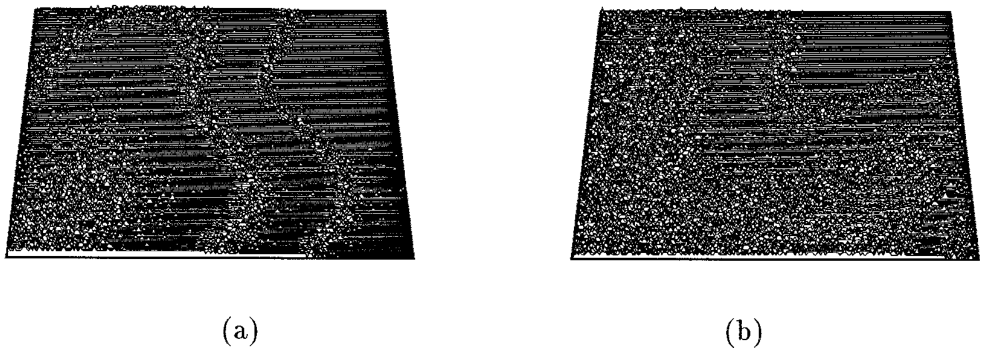


FIG. 11. Data range 1 to 5: (a) Height plot of response map for a planar scene with 4% noise in flow. The response is near zero everywhere. (b) Height plot for the response map for a purely rotational field and 1% noise in flow. The response is near zero everywhere.

of depth variation is needed for robust performance. Figure 12a plots the FOE error against $(Z_{\max} - Z_{\min})/Z_{\max}$ which is a simple way to approximately capture depth variation for a scene that is already known to have curvature in it. Depth variation is introduced into the experiments by moving the block of range data closer to or further from the synthesized viewer, while the block simulates the subtending of the same field of view in all cases. Throughout all experiments 8% mean Gaussian noise was added to the ideal synthesized data. The results confirm that depth variation is an important requirement for the algorithm to maintain robustness in the presence of noise. The effect of this factor is also emphasized in [18].

4.9.7. *Effects of Varying the Within-Triplet-Spacing, S*

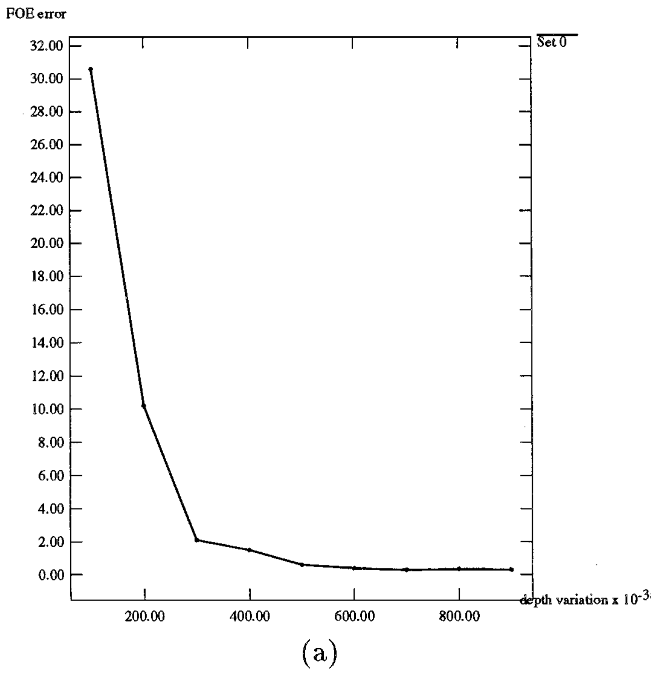
Once again inspecting the depth factor in Eq. (6), it can be observed that it grows with increasing separation of the scene coordinates along the X and Y axes. However, this is reciprocated by a division by the three depth values Z_i. Thus, if when using wider within-triplet spacings, κ_y is kept constant and the numerator (the separation in X and Y) grows relative to the denominator (the Z_i), the depth factor increases. Consider a typical convex triplet in the scene that projects to collinear image points, such that the midpoint is closer in depth to the viewer than the outer two. The outer two lie on limbs that recede from the viewer. Now, if a wider triplet is used but the center point is held constant, κ_y stays constant, while the outer two points are now points that are further away, and at the same time the separation along Y also grows. For shallower structures, the separation along Y grows faster than the points recede in depth, and the depth factor grows. Similarly for all concave structures, the separation along Y grows while the outer points are now more proximal to the viewer, hence the depth factor always grows. This is confirmed empirically with experiments on shallow depth variation for which results are shown in Fig. 12b. Figure 12b shows a plot of FOE error versus the spacing within triplets, for the case where the range image is placed at a distance such that $(Z_{\max} - Z_{\min})/Z_{\max}$ is 0.22 (quite shallow), and 8% mean Gaussian noise is added to the flow. The computation is sensitive to noise for small spacings but robust for spacings corresponding to over 10° visual arc. Similar behavior persists even when the range data is closer (making the scene less shallow). This is because the convex chevron structures in the scene truly have to be impossibly acutely angled before the effect of faster recession dominates for convex triplets to reduce the depth factor (the recession referred to here is that of the convex chevron's endpoints receding in depth away from the viewer as the triplet spacing is widened). Thus, using wider spacing within triplets gives larger depth factors and hence leads to more robust computations.

Several authors have noted that measurements from within a wide field of view are needed to get robust performance, while others have shown that taking a second derivative is very sensitive to noise. The obvious benefit of the framework here is that the robustness due to wide within-triplet spacing, S, eliminates the detrimental potential of the second derivative operation.

Figure 13a plots FOE error versus numbers of triplets used for the FOE operator, for ideal data plus 15% mean Gaussian noise. The plot is low and flat signifying little consequence of using fewer triplets.

The plots in Fig. 13b are profiles of one-dimensional slices through the response maps for experiments using about 1000 triplets and about a quarter of that amount, both including 15% mean Gaussian noise. The point of this figure is to show that the characteristics of the profiles do not change, only the heights vary a bit. Note that both profiles yield very similar (but wrong) FOEs (due to low SNR).

The effect of fewer triplets is most pronounced when there are large outliers in the flow data, possibly due to correlated noise or patches in the image that are moving independently (see Section 5). This is because for a given distribution of outliers, there is the chance that when fewer



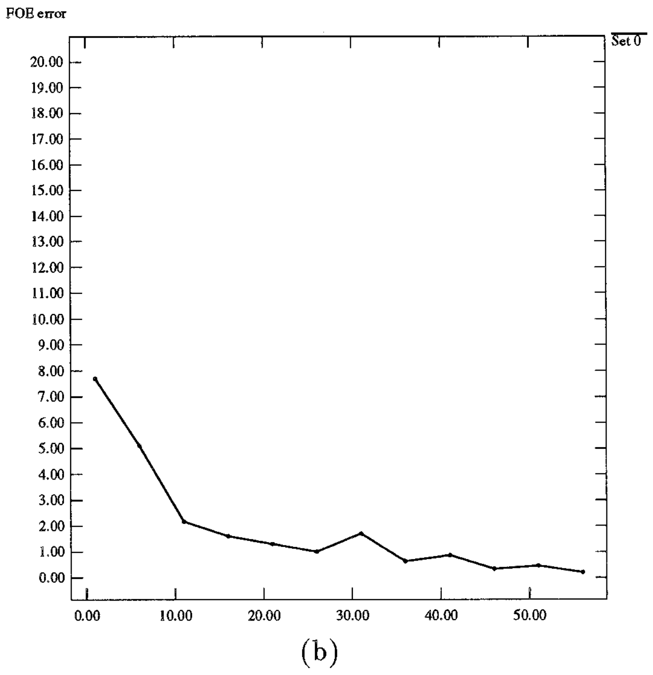


FIG. 12. (a) Plot showing effects of depth variation encoded as $(Z_{\max} - Z_{\min})/Z_{\max}$. Mean of Gaussian noise used was 8%, and a small within-triplet spacing, $S = 2$, was used. (b) Plot showing influence of within-triplet-spacing, S. Depth variation used was 0.22 and mean of Gaussian noise used was 8%.

triplet samples are used in the FOE computation, the outliers are disproportionately highly sampled. For uncorrelated Gaussian noise only the heights of the response profiles change, not the important characteristics of their outline. Thus the main reason to use more triplets is to offset the effects of correlated noise or patches of independent motion.

4.9.8. *Effects of Off-Linear Triplets*

Situations may arise where the points at which flow is available are not exactly collinear, and hence it is important to understand how this may affect the FOE computation. For the experiments reported in Figure 13c the collinearity was violated in the following manner. In the FOE operator, for each triplet, one of the points was randomly perturbed so that the flow measurement used for that point was actually from a neighboring point of the flow field. Here, wide within-triplet spacing of 20 grid-points between points was used. Figure 13c shows the FOE error versus the perturbation of a point within a triplet. The perturbation is measured in number of pixels distant from the correct position of the perturbed point.

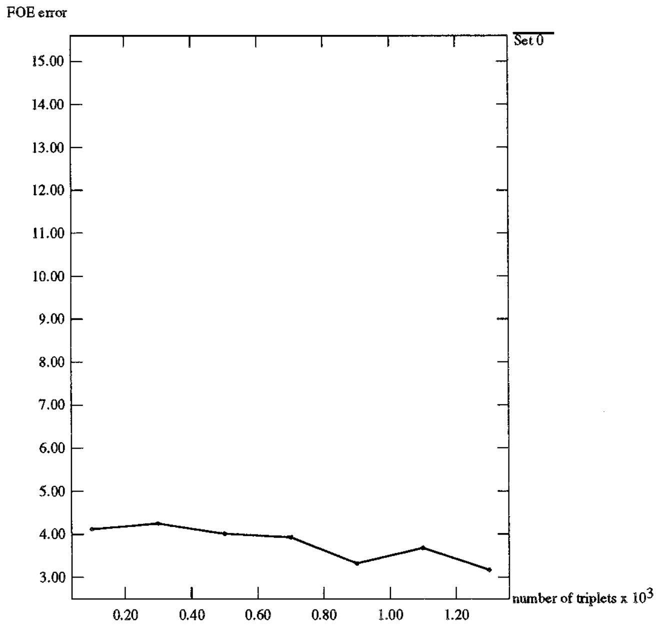
4.9.9. *Run Time of the FOE Computation*

Each FOE operator response takes the *TotalTriplets* number of triplet *Sum* computations, which in turn involves a small constant number of steps, called *StepsSum*. The variable *TotalTriplets* varies with position in the field of view, with peripheral positions allowing fewer lines which compose the operator. On each line an average of about 50 triplets are used, making *TotalTriplets* about 800 (for 16 lines), and the cost about 800 × *StepsSum* steps. Clearly, for cases where the scene was known to not contain independently moving objects, far fewer triplets could be used.

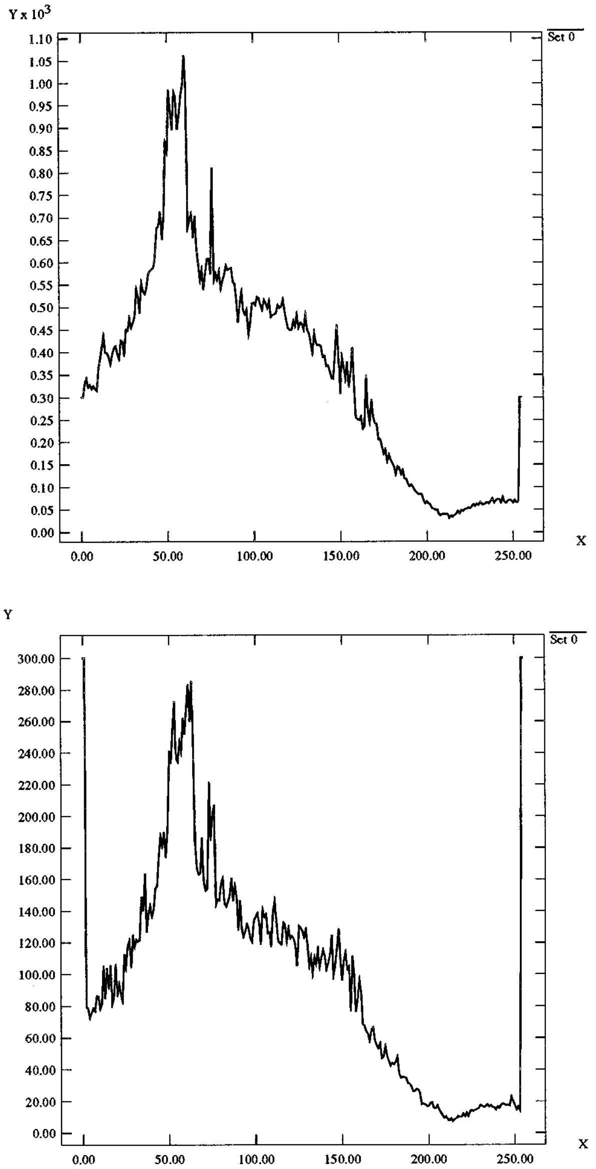
Eliminating the possibilities of pure rotation and of the entire scene being a single plane takes 16 FOE operator responses and 16 comparisons with some prespecified threshold. Thus, the cost for this stage is about 16 × 800 × *StepsSum* steps.

The time to compute the response map on a sequential machine is the number of sample points on the map times the time for computing each response. In the implementation described, 256 × 256 sample points were used, thus needing 256 × 256 × 800 × *StepsSum* steps. For the case of dense flow (about 256 × 256 flow vectors), this FOE algorithm involves linear cost. Finding the minimum is accomplished by keeping track of the minimum thus far, as the computation progresses, hence adding little additional cost. Finally, the interpolation of the map around the minimum point takes an insignificant number of steps.

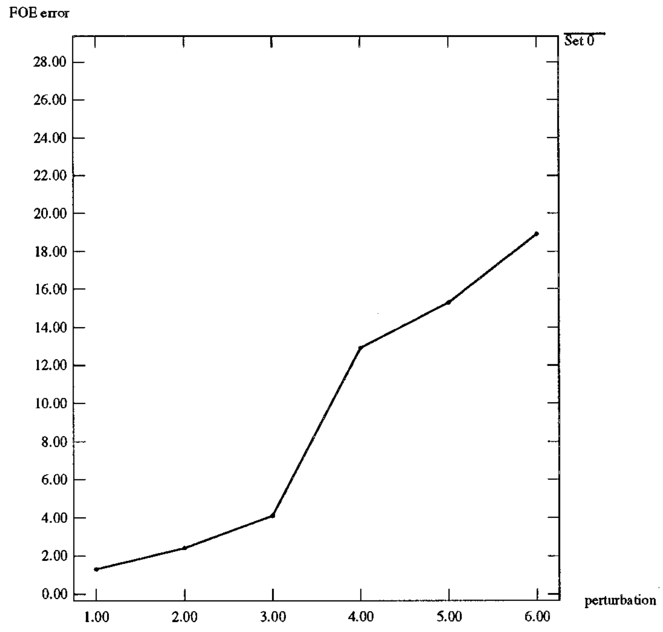
On a SPARC station, for the experiments reported above, the complete computation takes less than 10 min of CPU time. The algorithm may be parallelized effectively.



(a)



(b)



(c)

FIG. 13. (a) Plot showing effects of using fewer triplets on the FOE error. (b) 1-D slices through response maps for experiments using about 1000 triplets (top figure) and about a quarter of that many (bottom figure), both for 15% mean Gaussian error. The point of these plots is to show that the characteristics of the profiles do not change, only the heights vary. Note that both of them give similar wrong FOEs. (c) Plot showing effects of off-linear triplets. Abscissa represents perturbation of triplet's central point, i.e., distance of center point that was actually used, from where it should have been taken, in units of pixel distances.

4.10. Computing Translation's Sign: Backward/Forward

Thus far, the FOE computation has only provided the axis of translation but not the sign; i.e., it has not yet been shown how to tell whether the translation is backwards or forwards. Since this task merely involves computing a sign, it is a qualitative decision. The technique proposed here is an extension of the observation in [32] that rotation can be cancelled for two points on a spherical projection if each point's component along the great circle joining the two points is considered. That observation was also made use of in [30] when computing the directional divergence but employed for different purposes than the one here. The foundation of this observation is that on a spherical projection, the rotational component of flow projected along a great circle is constant. This means that for the spherical projection, for any two points in order to cancel rotation, it suffices to compute their flow measurements' projection along the great circle and to subtract one from the other; this eliminates rotation. In planar projection, the subtraction for any two points' components of flow along the counterpart of a great circle must be appropriately weighted due to the distortion caused by unfolding the sphere and its flow field onto the plane. For any two points in the image plane, this distortion (and, hence, the weights) are known. In Appendix A, these weights are algebraically derived.

Thus given an FOE position the algorithm is the following. Take several (the theoretically required minimum here is one measurement) flow measurements peripheral to the FOE. For each of them, consider the counterpart of the great circle joining the FOE to it. Then subtract the FOE's projected flow along the great circle from the point's projected flow weighted by the term needed to compensate for the distortion caused by using planar projection. Since the rotation will be cancelled out by this operation, what remains is the subtraction of the projected translation components of the two points (the FOE and the point under consideration). Since the FOE's translation component is zero, the result of this subtraction is the translation component of the point, and it should point either towards the FOE or away from the FOE. This exactly indicates whether or not the translation component of the observer's motion is forwards or backwards. See Fig. 14a.

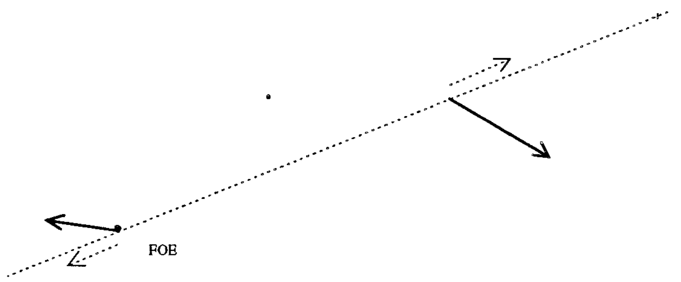
The only difficulty with this scheme is that there may not be a flow measurement at the FOE since the FOE may lie between flow grid sample points. In that case, the closest flow measurement to the FOE is used in lieu of the FOE. In practice, 20 arbitrarily chosen peripheral points were used for this backwards/forwards computation and after the computation with each point, the vote of a two-thirds majority was used as the decision criteria for whether the motion was backwards or forwards. These figures were chosen to keep the computation efficient, while still sampling a sufficient number of points to avoid spurious results. There are very few conditions under which this majority requirement is not satisfied. Such conditions would include cases where many of the points come from independently moving objects or are extremely noisy measurements, hence leading to inconsistent and inconclusive evidence, such as a 50–50 split. Figure 14b shows how the sign computation can tolerate substantial noise in the arbitrary points' measurements even if the FOE estimate is off by a couple of degrees of visual arc. Here, noise tolerance is defined as the point at which the noise in the flow of the arbitrarily chosen points causes the sign computation to compute the wrong sign. The arbitrarily chosen points before noise was added came from the ideal flow of experiment 1.

5. FUNCTIONING WITH INDEPENDENTLY MOVING OBJECTS

Objects in a moving observer's view will often move independently—a typical scene is rarely rigid. It is important that any egomotion algorithm maintain competence in these situations. Figure 15a depicts the flow field generated by combining one of the rigid flow fields used in earlier experiments in Chapter 4 with the flow for an independently moving rectangular patch in the upper middle of the image. The frontoparallel patch translates upward, to the right, with an image flow magnitude comparable to those of the flower petals. It subtends an angle of about 9° visual arc by 6° visual arc. The FOE algorithm was tested on this combined flow field. The output FOE is still within 1° visual arc distance away from the true FOE (operator response map in Fig. 15b.) Tests indicate that even with modestly larger patches the FOE computation is accurate to within a degree of visual arc, demonstrating robustness to patches of independent motion in the scene. However, this breaks down at some point. Next, effects of independently moving objects on the FOE computation are studied systematically and empirically. Then, theoretical results showing how collinear points assist in detecting independent motion are presented. After that, previous work is briefly discussed, followed by an algorithm for detecting independent motion that relies on knowledge of the direction of translation.

5.1. Effects of Independent Motion on FOE Computation

First, it is important to understand that as far as the FOE operator is concerned, when it is centered at the correct FOE, barring small amounts of noise in the flow field, the lines it radiates should typically encounter flow values for which the triplet *Sums* are zero. Any nonzero *Sum* contributions are noise to this computation. The magnitudes of this contribution to noise are dependent on the magnitude of the flow value that is not consistent with the correct FOE. Thus the magnitude of the velocity of the independently moving patch is a contributing factor to the noise component. The size of the independent patch is clearly a contributing factor to the noise, as the larger the patch the more often its flow values will appear in the computation of the operator's response. So, it is intuitively clear that the magnitude (and direction) and the size of the patch, and similarly the number of patches, degrade the performance of the FOE operator.



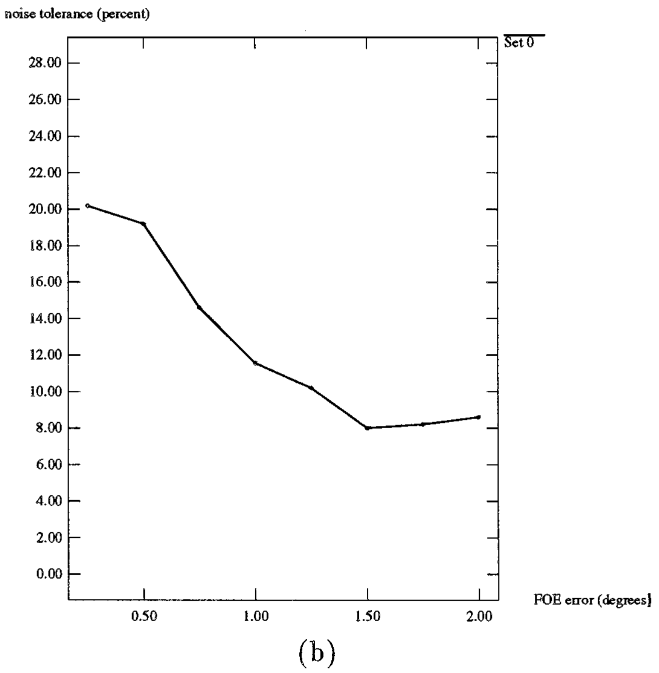


FIG. 14. (a) Figure shows how rotation can be cancelled between the FOE and an arbitrary point, and thus used to tell the sign of translation from the translation of the arbitrary point. The dark arrows are the flow measurements at the FOE and at the point, and the dotted arrows are the projections of flow along the great circle joining the two points. In spherical projection, the rotational component along the great circle (the dotted arrow) is identical for two points along the great circle. (b) Plot showing noise tolerance of sign computation to error in FOE position.

Figures 16a, 16b, and 17 show the range data for the scene containing blocks, the flow associated with this scene in which the upper-middle block (a larger patch than before) moves independently, and the FOE response map for the flow. (To simulate motion from the range data, the closest point in the scene is at 60 FLUs and the furtherst is at 600 FLUs.) Note that in this case to keep the FOE detectable, the flow values of the object had to be smaller than in the patch used earlier. The object subtends about 20° of visual arc in the horizontal and vertical directions.

Figure 18a shows the inverse nature of the size of object and its maximum velocity allowable to permit the FOE computation to still produce the correct result, for the original scene's ideal flow with 3% mean Gaussian noise. Note that in all the experiments for this figure, the direction of flow of the object is kept almost or exactly perpendicular to the set of lines from the FOE traversing the object. In this way, the effects of the patch, as noise, on the FOE computation was maximal. (Later, Fig. 19a presents the effects of patch motion direction relative to scene motion direction, i.e., the FOE field lines.)

Figure 18b shows how for a fixed velocity of a patch, a fixed mean (of 3%) for the backround's Gaussian noise, and a fixed position of the patch's center, increasing the patch's size increases the FOE error.

The overall direction of the patch velocity relative to the direction of translation for the rest of the scene is also relevant to how damaging a patch's effect is on the FOE computation. This effect is shown in Figure 19a. The patch used here subtended 30° visual arc, so that this will cause a substantial FOE error when the direction of patch motion is such that the patch's detrimental effect is maximal. The magnitude of the patch's flow was kept constant across experiments. As the figure shows the effect peaks when the translation direction of the patch is orthogonal to the scene's local direction of translation (i.e., the FOE field lines) and ebbs when the directions coincide. This is to be expected because as the patch's direction of translation approaches that of the rest of the scene, this method interprets it as part of the rigid scene.

Another factor that has potential influence is the proximity, as measured in the image plane, of an independent patch to the FOE. As an independently moving patch is just noise, the frequency of its contribution to the FOE operator is clearly a factor in its overall influence. The FOE operator, as it has been currently designed, is biased to flow values near the center of the operator due to the fact that more lines traverse that area of support than any other area. Thus, when the operator is centered at the FOE, a patch's deleterious effect is inversely related to its distance to the operator center (which is the FOE). Figure 19b shows the effects of proximity of the patch to the FOE on the FOE computation.

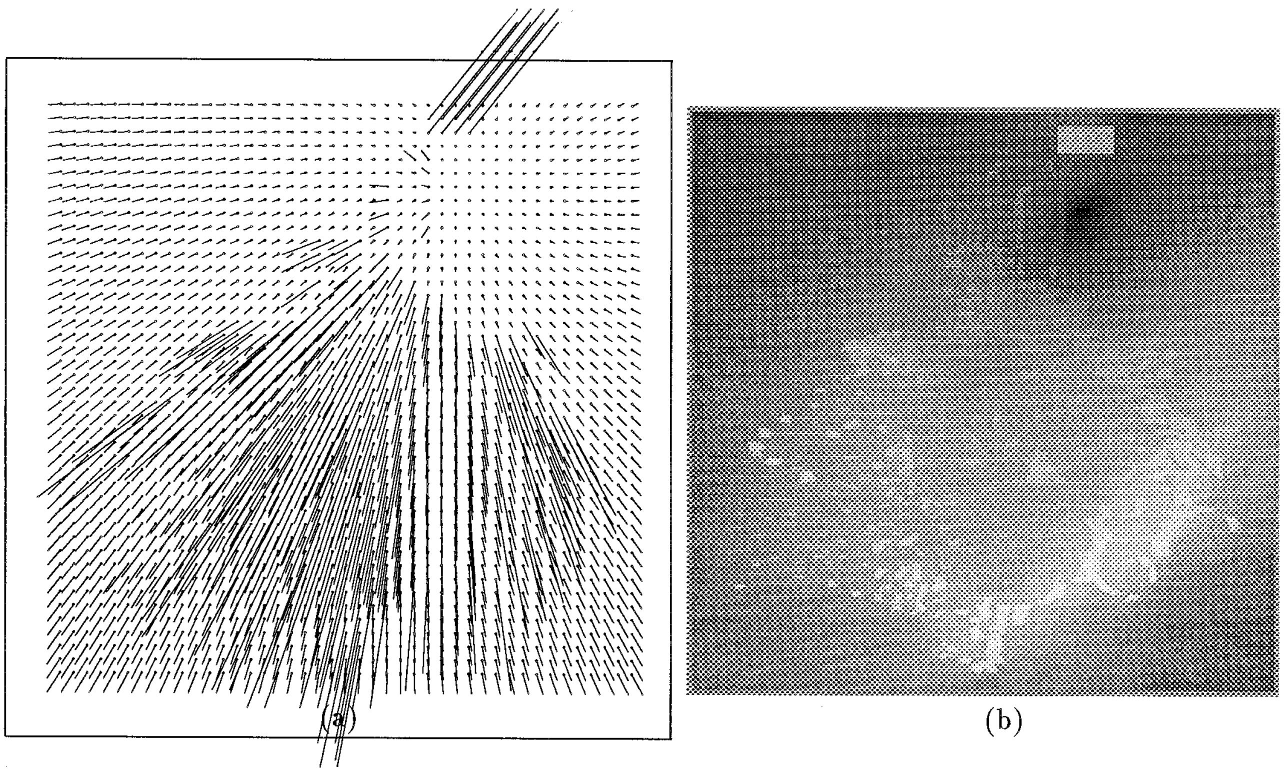


FIG. 15. (a) Flow field combining the original rigid flow field and an independently moving patch in the upper right part of image. The frontoparallel rectangular patch translates upward, to the right, with an image flow magnitude that is comparable to those of the flower petals. (b) Response map for the flow associated with the nonrigid scene in which a patch moves independently. The global minimum is in the same position, indicating that the FOE computation is robust to some nonrigidity in the scene.

5.2. Finding Points that Thwart Rigidity

For a 3-D motion algorithm that maintains competence despite some independently moving scene parts and noise, the next step in its enhancement is to enable it to detect parts of the scene where the overall-rigidity assumption does not hold.

A point may be measured as moving differently from the rest of the scene either because it may be a noisy measurement, in which case when the shape reconstruction step is being carried out this erroneous measurement should be treated with caution, or because it could legitimately belong to an independently moving part of the scene. If this is the case, this moving part will probably need additional attention and possibly some special purpose processing. Such processing might lead to segmenting the moving objects from their background or segmentation into parts (in articulated motion). Analysis of the nonrigidity may also provide data for classifying objects in scene recognition.

As a first step, it is desirable to have the algorithm signal that certain points are moving inconsistently with respect to the rest of the scene. Here, first, some theoretical preliminaries are presented and then other approaches are reviewed. Then it is shown that the egomotion algorithm can be straightforwardly extended to indicate whether or not all three points in a triplet lying on a line through the FOE are moving with the 3-D motion of the rest of the scene. It suffices to test if the triplet *Sum* for the three points is near zero or not. Since it has already been shown in Section 4 that if such a collinear triplet were moving with the same rigid parameters as the scene, the triplet *Sum* must be zero, then, when *Sum* is sufficiently[4] above zero, at least one of the three points is moving inconsistently with the rest of the scene.

5.3. Detecting the Absence of Rigidity: Theoretical Limits

The question addressed here is: given an FOE, how many points should a subset consist of before it is possible to tell whether this subset includes at least one point not moving consistently with that FOE.

PROPOSITION. *Given two flow measurements and an*

[4] This threshold, as noted later, is set so that even noise as high as 15% in the flow field does not cause a noisy triplet to be signalled as independently moving.

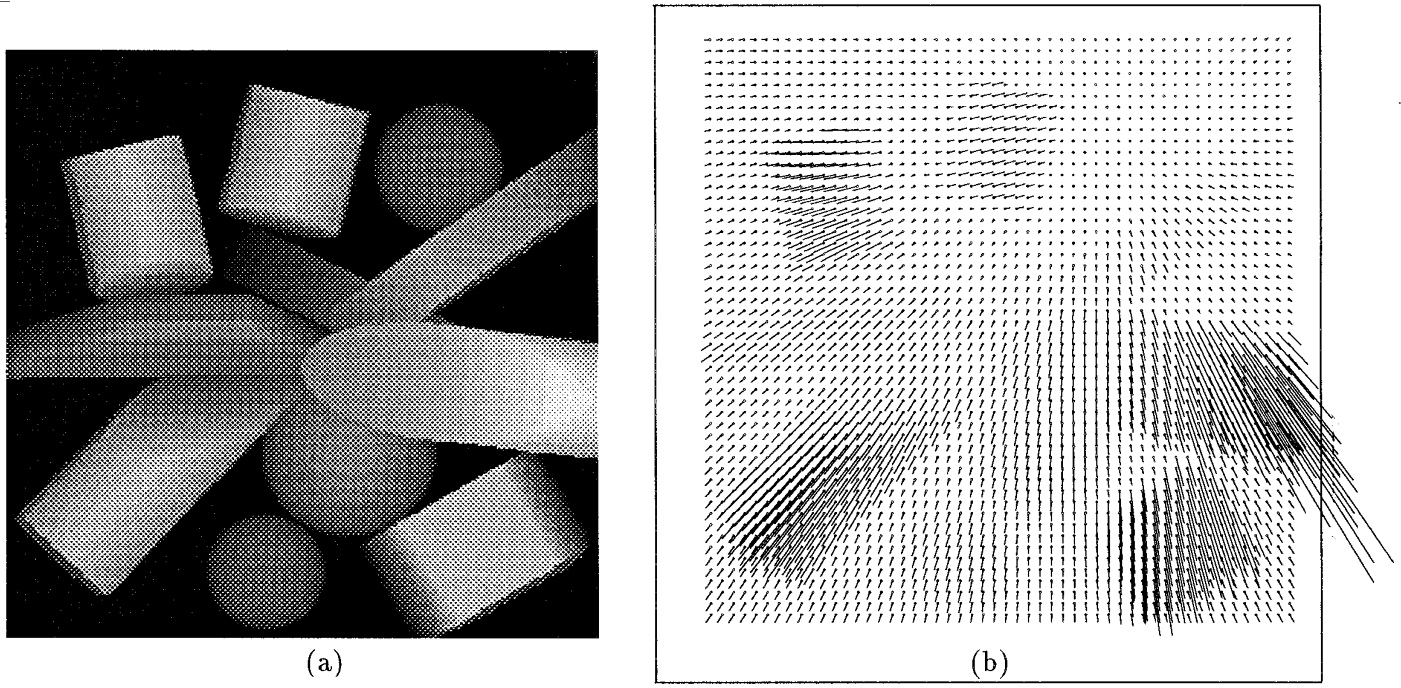


FIG. 16. (a) Range data of scene containing blocks. (b) Flow associated with the blocks scene, in which an object subtending a larger patch moves independently. The object is the block at the upper central portion of the image: it translates independently to the right and slightly upwards.

FOE, it is not possible to determine definitively whether or not they are moving rigidly with the parameters of that FOE.

Proof. What needs to be shown here is that for any two flow measurements, there is always a set of motion parameters $(U_0, V_0, W_0, A_1, B_1, C_1)$ that could have generated both flow measurements, where (U_0, V_0, W_0) is a translation component whose direction (the FOE) is specified in advance. As shown in Appendix A (Eq. (8)), in relation to an observation in [32], there is a way to cancel the hypothesized common rotation (A_1, B_1, C_1) for the two points. Then the calculation *TwoSum* applied to the two measurements can be interpreted as,

$$TwoSum(u_1, v_1, u_2, v_2) = \frac{1}{Z_1}\begin{bmatrix} U \\ V \\ W \end{bmatrix} \cdot \begin{bmatrix} C_{U,1} \\ C_{V,1} \\ C_{W,1} \end{bmatrix} + \frac{1}{Z_2}\begin{bmatrix} U \\ V \\ W \end{bmatrix} \cdot \begin{bmatrix} C_{U,2} \\ C_{V,2} \\ C_{W,2} \end{bmatrix},$$

where the $C_{U,i}$, $C_{V,i}$, $C_{W,i}$ are simple polynomials in the position-coordinates of the two points. *TwoSum* is a linear sum of the flow inputs.

Clearly, to get the equality to work for a specific choice of direction for (U_0, V_0, W_0), the sign of translation and relative values of Z_1 and Z_2 merely need to be adjusted. Thus there is a consistent set of motion parameters (including the translation parameters are constrained by a particular FOE) that could have generated the two flow measurements. Another way to see this is to use the result in [24], that for a given FOE, projecting the flow orthogonally to

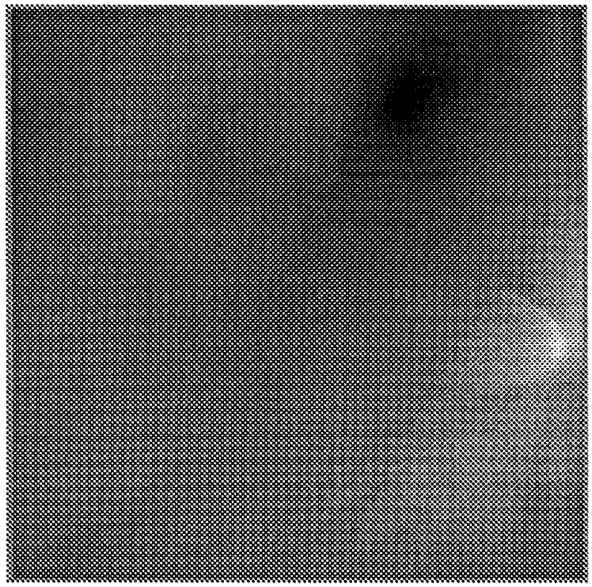
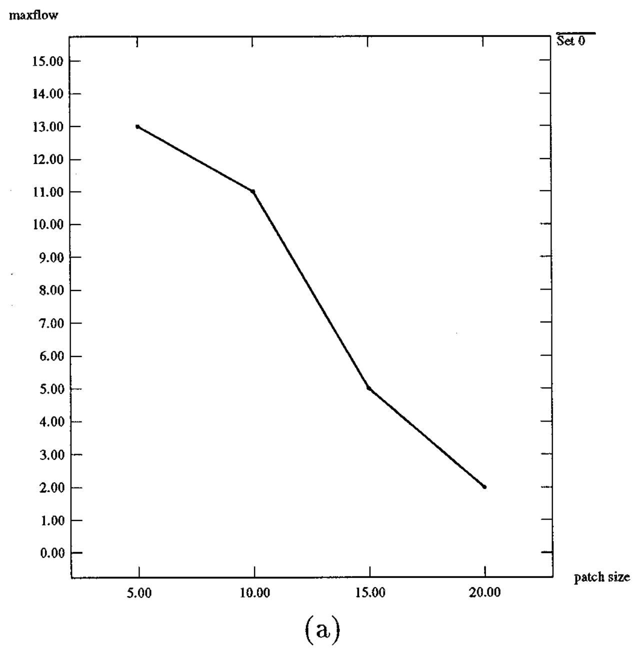


FIG. 17. Response map for the blocks' scene flow containing the larger independent patch.



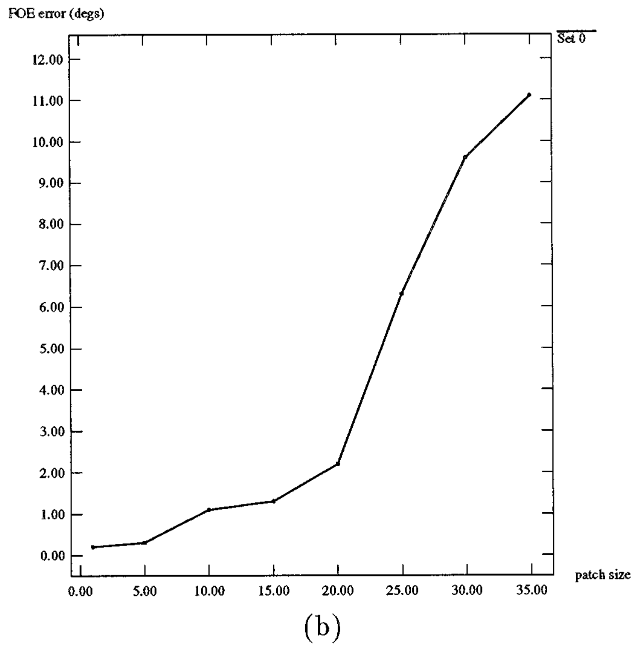
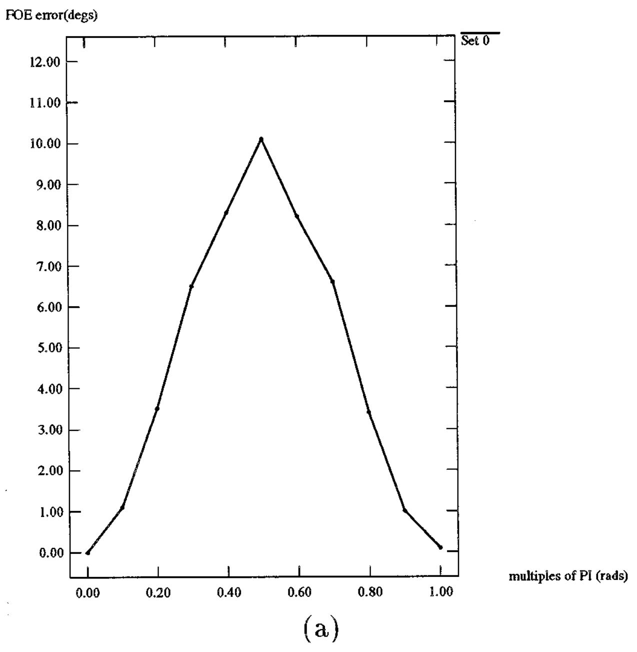


FIG. 18. (a) Plot shows some independent patch sizes and the maximum allowable flow magnitude orthogonal to the FOE radial lines traversing the object. (b) Plot shows how FOE error increases with independent patch size. Here, a fixed velocity of a patch, a fixed level of background noise of mean 3%, and a fixed position of its center were used.



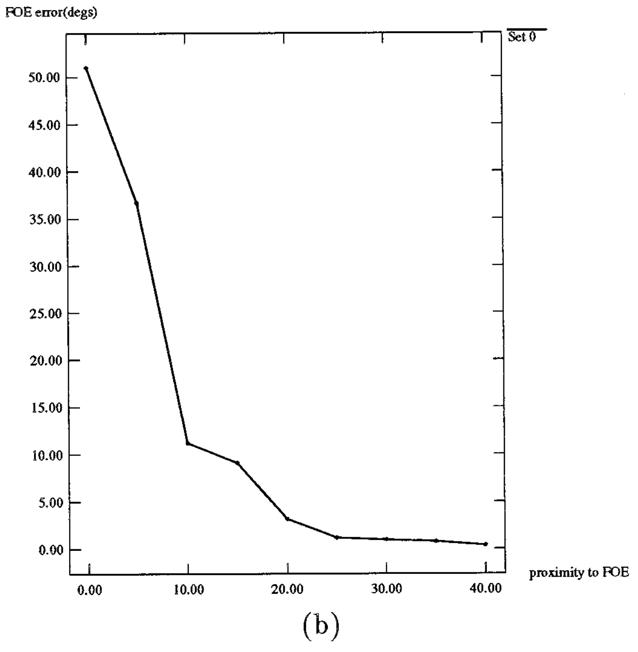


FIG. 19. (a) Plot shows how FOE error is dependent on the relative directions of translation between the patch and the overall scene data. The background used here had noise-free flow, and the directions of the patch relative to the direction of translation of the rest of the scene in this region are varied from 0.0 to π. (b) Graph shows effects of patch's proximity to FOE on FOE computation. Abscissa is in grid-point distance units.

a radial line gives a linear equation in the three rotation parameters. Thus if this is done for the two points, one obtains two linear equations in three rotation unknowns, and so it is clear that there are in fact an infinite number of sets of rotation parameters that are consistent with a particular FOE and two flow measurements. ■

Proposition. *Given an FOE and flow measurements at three points that are not all collinear with the FOE, it is not possible to definitively exclude a rigid explanation consistent with that FOE, as there is a set of rotation parameters and depth values that is consistent with that FOE and the three flow measurements.*

Proof. The simplest way to confirm the existence of a parameter set that is consistent with these three points, not all of which are collinear with the FOE, is to use Longuet-Higgins and Prazdny's rotation projection result again. For the particular choice of FOE, there are three linear equations in the three rotational parameters (these equations are derived from the projections orthogonal to the radial lines). As long as the three points are not all collinear with the FOE, the linear system always[5] has a solution. ■

Next, the significant theoretical contribution of this section is stated in the form of a theorem.

Theorem. *The minimum number of points for which it is possible to detect inconsistency of motion with a particular FOE is three points in a collinear relationship to the FOE.*

Proof. Taking the three orthogonal projections of Longuet-Higgins and Prazdny's rotation projection result again, the three projected rotational components are obtained. However, due to the fact that these are collinear points, according to the Collinear Point Constraint, these orthogonal components must vary linearly along position. Thus, if they do not, the points must be moving inconsistently with that FOE and a single set of rotation parameters. Hence, the triplet *Sum* as defined in Eq. (4) must be zero for rigid points. Otherwise, at least one of the three points is moving independently from that specified FOE. Since it has already been shown that for two points in any configuration, as well as for three points not all collinear with the FOE, there is no way to detect the absence of rigidity, this is the smallest configuration that can indicate this property. ■

[5] The reader may wonder about the case where two of the three are collinear with the FOE. In this case, these two points give projected equations that determine (due to the linear variation of the orthogonal component to the line that a common rotation is required to satisfy) the rotation component orthogonal to the line joining these two points and the FOE. However, the component of rotation *along* this line is still a free parameter, and this allows the third point to have an arbitrary image motion vector.

5.4. Previous Work on Segmenting Independently Moving Objects

Methods of detecting independently moving objects can be classified into 2-D methods and 3-D methods. The 2-D methods use a heuristic or make assumptions about the world or about the motion, so as to permit direct inspection of the flow to achieve the segmentation.

Some authors assumed that the camera is stationary, so that significant image motion indicates independently moving objects. [17] extracted interframe differences to extract images of moving objects, and [16] pursued a related method. This approach does not offer any solutions for a moving camera.

A recent proposal by [15] is representative of the 2-D approach. They assume that projected 3-D motions can be approximated by a 2-D parametric transformation in the image plane such as pure translation, affine transformation, and projective transformation. Once a motion has been determined, the region is identified; the two images are registered using the detected motion and nonstationary regions are segmented. This scheme works well in practice but has difficulty with motion in depth or significant rotation of the independently moving object.

The 3-D methods, on the other hand, explicitly utilize an underlying model of 3-D motion and possibly make simplifying assumptions about the model. The most general method was that of [1], described earlier for computing the parameters of 3-D motion of patches. There, motion segmentation was proposed based on the difference of these 3-D parameters. This is, of course, very desirable in motion vision but, as has been shown repeatedly since then, is difficult for patches subtending small fields of view.

The authors of [42] investigate a variety of combinations of limited motions and known parameters. For known pure translation, the motion epipolar constraint, namely that the flow is restricted to FOE field lines, offers enough constraint to permit a test for independence. They also investigated detecting independent motion assuming that the rotation is known, and, in another experiment, that depth is known. [29] used the analysis of [42], adding assumptions about the ranges and possible direction of observer motion. [41] attempted to apply ideas from robust statistics to compute global motion parameters in the presence of outliers, where the outliers are likely to be associated with independently moving objects. Orthography is used along with a qualitative analysis. Due to the use of orthography, a substantial number of independently moving points are not detected as independent, and, in addition, the LMedS procedure from robust statistics is very computationally intensive.

[9] assumes that motion parameters are known for the case of pure translation and uses a complex logarithmic mapping (CLM) representation of the motion epipolar

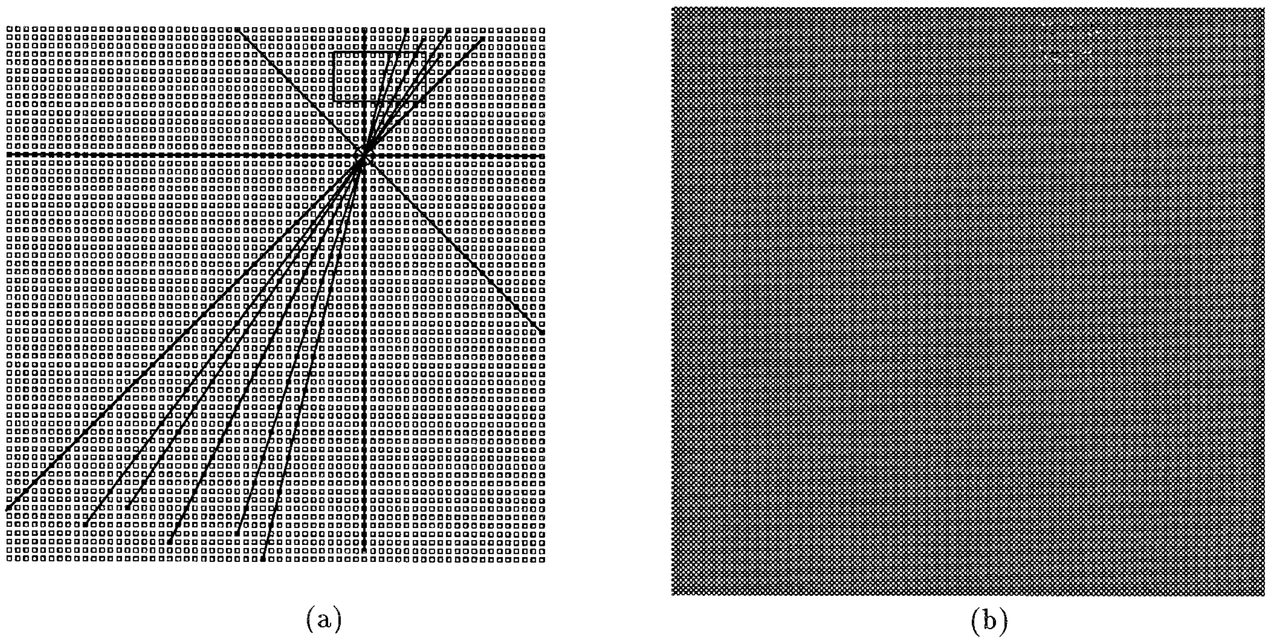


FIG. 20. (a) Points from an independently moving patch as in the upper right corner will be detected by lines passing through it. (b) The program marks points in triplets that overlap the patch in this flow as inconsistent triplets. Because the patch itself is moving in a planar fashion, the triplet sum, when all three points are inside the patch, is zero. So, these internal points are not marked.

constraint. The method needs horizontal or near-horizontal edges in the CLM image. [37] investigates the use of purely normal flow in the detection of independent motion. It is assumed that translation dominates, and hence, normal components found towards the FOE imply independent motion.

Several interesting ideas have been considered in previous work, but none have succeeded in the case of unknown rotation. Here, because the collinear points allow the simple cancellation of rotation, a powerful constraint is obtained on rigid motion, making independent motion detection easy.

5.5. A New Algorithm for Segmenting Independently Moving Objects

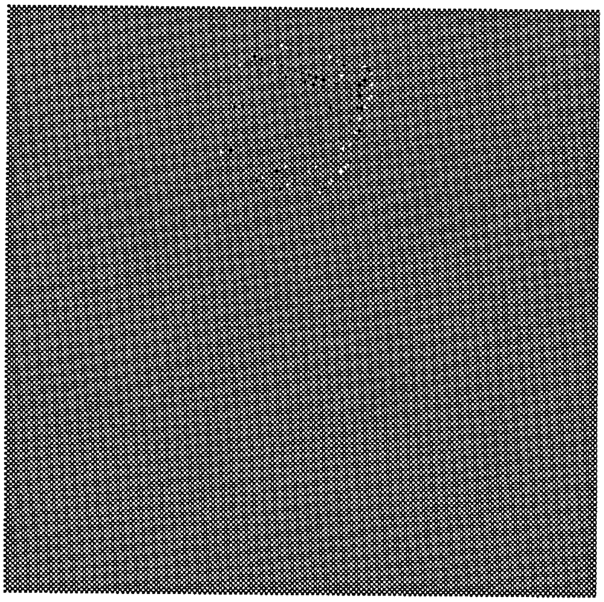
A practical way to detect independently moving objects is to use the operator that gave the minimum response indicating that the FOE is nearest to its center. Each of its various lines (only 16 lines were used in this implementation) (see Fig. 20a) can be scanned, searching for triplets that do not provide near-zero sums. Inspecting from the end of a line, triplets of within-triplet spacing 1 grid unit (a triplet spans about 1.5° visual arc) are examined consecutively. When *Sum* is above a certain threshold (determined by trial and error, so that even noise levels as high as 15% in the flow field would not cause a noisy triplet to be signalled as independently moving), the center of the triplet is marked. The computation can be performed in parallel as a convolution on sets of three inputs, followed by a Loser-Take-All computation; the Winner-Take-All algorithm presented in [45] can be trivially modified to find a minimum value of a set in a parallel, iterative and highly efficient manner. Figure 20b shows the region around the patch in the flow field of Fig. 20a marked by the program as a set of inconsistent triplets. Note that in this case because the image patch itself is moving as a planar scene region would, the triplet sum, when all three points are inside the patch, is zero. So, at an independently moving planar region, only the boundary areas of the region will be detected (this depends on the within-triplet spacing used). For an independently moving nonplanar region, all the triplets, in any manner overlapping the region's points, will be detected. In this case the patch corresponded to an object subtending 10° of visual arc.

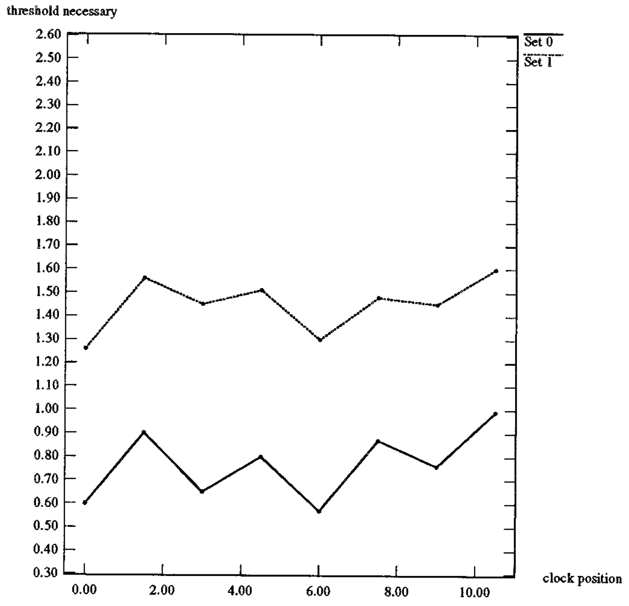
Figure 21a shows the outcome of the above sequence of steps for the larger object used in the flow shown earlier.

The issue of choosing an appropriate threshold to decide when a triplet should be considered to have a nonzero *Sum* is a difficult one that has not yet been adequately addressed.

Isolated triplets that are found to be non-zero probably represent noisy flow measurements rather than independent motion. Grouping these triplets to identify regions is one approach to distinguishing between isolated high noise points and independent motion.

The center of the independent motion detector may not


(a)



(b)

FIG. 21. (a) The larger independent object used in the above figure, being signalled as independent. (b) Threshold necessary to avoid flagging many of the rigid points vs the clock position of the assumed FOE relative to the correct FOE. Data shown for assumed FOE at 1 pixel away (solid lines) from correct FOE and at 2 pixels away (dotted lines).

be positioned at the exact FOE due to the operator's limited sampling, or due to FOE computation error. Figure 21b shows how for exactly the same data, the threshold needed for independence detection varies with an error in the estimated FOE. Here to represent the error in the assumed FOE, clock positions are used to show the relative position of the assumed FOE with respect to the correct one. Data are shown for distances one pixel away and two pixels away.

6. DETECTING INDEPENDENT MOTION BEFORE THE FOE

In the previous section, it was seen that the FOE computation is not robust to arbitrary amounts of independent motion. There were limitations on size and speeds and numbers of patches moving independently to ensure robustness of the FOE operator. Hence, an important direction for research is to try to presegment the independent motion before applying the FOE operator. Here, this direction is pursued.

Consider the operator in Fig. 22a. It is a smaller version of the post-FOE-computation-independent-motion-detector, in the form of a 9×9 grid (the number 9 was chosen to keep the region of support for the operator small, while at the same time permitting a sufficient number of triplets to be chosen along each line to permit the detection of independent motion). Through each of its lines it has 5 or 9 flow measurements, depending on the orientation of the line. No matter where it is centered in the image, one or two of its lines will be oriented in directions close to that of a line joining the operator's center to the FOE. Along each line, the three (or five, depending on the orientation of the line) triplets are examined in sequence. For the one or two lines (which are unknown), the triplet sums will be near zero, so that a traversal along the line should give near-zero responses throughout, unless at one end of the line there is a flow value from an independently moving patch. This will cause a step in the triplet *Sum* as a function of position. If this pattern occurs along a line, the step is marked. If two lines have similar patterns in that the step occurs around proximal ends of the two lines, that region is marked as being a boundary to an independently moving patch.

The only case that needs to be eliminated before applying this operator is the case where there is a depth boundary adjacent to a planar region. This can be dealt with by not applying the operator to regions that may be planar. Regions can be considered possibly planar when the eight triplets that straddle the operator's center flow measurement all give near-zero *Sums*.

The main drawback to this approach is that independently moving objects against planar backgrounds are not signalled. The intuitive reason for this failure is the following. The 3-D motion based segmentation approach outlined in this section needs vague information of the location

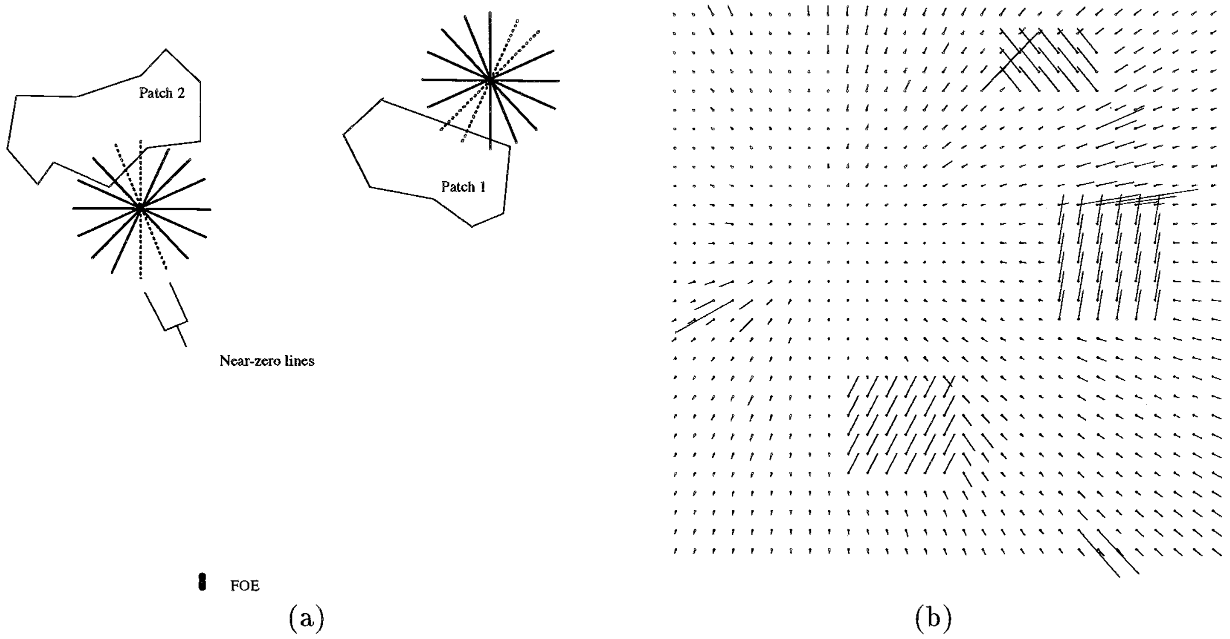


FIG. 22. (a) Operator for segmenting independent motion before the FOE computation. (b) The flow field for the faces range data, with three large independently moving patches in the field of view. Gaussian noise (with mean 3%) is added to the background flow values.

of the FOE. With a planar background, the collinear point approach is completely incapable of giving any information about the FOE. Hence, in the local environment of a planar region, a local approach to independent motion detection fails.

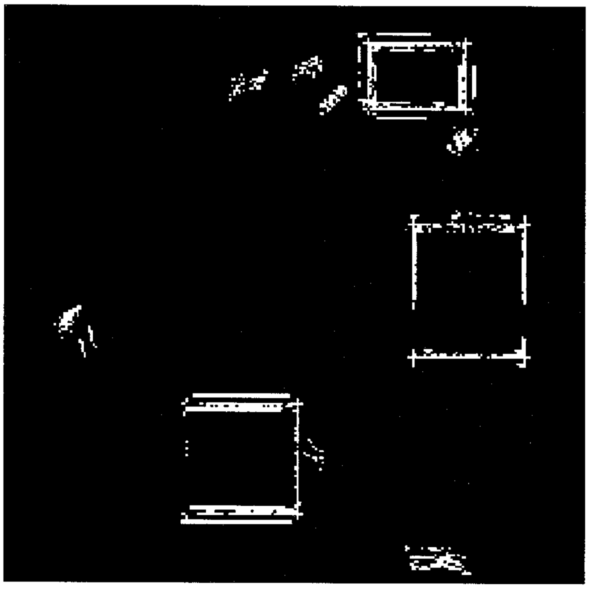


FIG. 23. Multiple and large patches signalled using the early segmentation method. Note that some spurious regions are also signalled; these will need to be processed by some grouping mechanism later.

Figure 22b shows the flow field for the faces data, with three large independently moving patches in the field of view. Figure 23 shows these patches signalled by the new algorithm. In this implementation, only one line was required to have a near-zero *Sum* before it was used to detect a discontinuity. Note that some other spurious regions are also signalled; thus, the output of this algorithm needs further processing to remove false positives.

7. SUMMARY

This paper has presented a theory based on the Collinear Point Constraint for computing the instantaneous direction of translation for an observer moving with unrestricted motion and for computing the locations of independently moving objects. The constraint is based on cancelling linear variation of the flow field, as this eliminates rotation. It also eliminates the translational flow due to motions of planar regions, and hence the detection of the FOE is based on higher order variation in the flow field such as that occurring where there is translation in an environment which looks sufficiently different from a single plane. We showed how collinear points can be used to cancel the rotation and simultaneously constrain the direction of translation's FOE, which can then be localized from the intersection of constraint lines. An FOE operator was constructed that consists of radial lines that sample the flow measurements projected orthogonally to these lines and locates the FOE when the operator is swept across the flow field. The use of collinear points provides an elegant characterization of the depth variation needed to succeed, as well as an analysis of the effects of the spacing between points used in the computation.

The FOE computation was found to be reasonably robust to noise in the input flow, with robustness improving with greater depth variation and wider spacing between points in the unit computation with three collinear points. The method of location of the FOE involves a 2-D search, and it is convenient to limit the search to the nodes on a grid. The effects of interpolation due to the discrete sampling imposed by the grid on the computation, of differing amounts of rotation and translation, of marginally noncollinear image points assumed to be collinear, and of using different numbers of sample points have been empirically studied and reported. A technique for obtaining the qualitative sign bit of the direction of translation was presented and empirically tested on synthetic data.

Next, we examined how the FOE computation performs in the presence of independent motion and deals with the issue of detecting these motions. The FOE computation was found to be susceptible to independent motion that has very large magnitude, that occupies a large percentage of the field of view in either whole or disjoint chunks, and that occupies a patch very close to the true FOE. The exact points at which the algorithm would break down are dependent on the interplay among these and other factors, such as noise, depth variation, and the spacing of triplets.

To understand the possibilities for independent motion detection, theory was presented that showed that three collinear points is the minimum structure needed to be able to accomplish this task. Then, a review of previous work on independent motion detection was done, and a new algorithm was presented and tested empirically. Lastly, another avenue was explored, in which independently moving objects are detected before the FOE is computed. Preliminary results were presented in [8], and further details appear in [7].

This framework based on the Collinear Point Constraint has shown promise in the domain of egomotion and independent motion. It needs to be further studied in the area of the development of a control strategy for the detection of large numbers of independently moving objects, the extension to benefitting from multiple image frames, and the use of purely normal optical flow. Other current work includes the integration of the representations of egomotion direction and independently moving objects into the attentional framework of [45, 5]. At the moment, the system integrates and uses representations of luminance, edges, abrupt onset and offset, motion categories, and peripheral cues for directing attention and provides attentive control signals which form part of the control strategy for the TRISH stereo head [25]. The head is mounted on a mobile platform, and thus the addition of the egomotion and independently moving objects representations will form critical inputs for the successful navigation behavior of the robot.

APPENDIX A

For the two image points p_i, the x- and y-components of the image velocities will be (from Eqs. 1)

$$u_1 = \frac{-U + x_1 W}{Z_1} - Ax_1y_1 + B(x_1^2 + 1) - Cy_1$$

$$u_2 = \frac{-U + x_2 W}{Z_2} - Ax_2y_2 + B(x_2^2 + 1) - Cy_2$$

$$v_1 = \frac{-V + y_1 W}{Z_1} - A(y_1^2 + 1) + Bx_1y_1 + Cx_1$$

$$v_2 = \frac{-V + y_2 W}{Z_2} - A(y_2^2 + 1) + Bx_2y_2 + Cx_2.$$

Now noting that there are four equations that are linear in three of the unknowns A, B, C, these three will be eliminated from the system, thus reducing the system to one equation in the other unknowns. This can be summarized as

$$TwoSum = \frac{1}{Z_1}\begin{bmatrix} U \\ V \\ W \end{bmatrix} \cdot \begin{bmatrix} C_{U,1} \\ C_{V,1} \\ C_{W,1} \end{bmatrix} + \frac{1}{Z_2}\begin{bmatrix} U \\ V \\ W \end{bmatrix} \cdot \begin{bmatrix} C_{U,2} \\ C_{V,2} \\ C_{W,2} \end{bmatrix},$$

where *TwoSum* is a linear sum of the flow values involving coefficients that are polynomials in the image position coordinates and the RHS is its interpretation, where the symbol $\cdot$ represents the inner product and where the $C_{tj,i}$ are polynomials in the image position coordinates. The $C_{tj,i}$ are either obtained by simple Gaussian elimination or by using the observation of Prazdny [32] that the component of flow in the direction of a line joining two image points has constant rotational contribution on a spherical retina. Note that Prazdny's observation was also used in obstacle avoidance work by Nelson and Aloimonos [30] employing directional divergence. The directional divergence projects the flow for two points onto the great circle arc joining the two and takes their difference, thus cancelling the rotation component.

The inner products above can be expressed as

$$\begin{bmatrix} U \\ V \\ W \end{bmatrix} \cdot \begin{bmatrix} C_{U,i} \\ C_{V,i} \\ C_{W,i} \end{bmatrix} = |T|\, Pr_i$$

where $|T|$ is translation's magnitude and Pr_i is the projection of the vector $[C_{U,ki}, C_{V,i}, C_{W,i}]^T$ onto the translation vector $[U, V, W]^T$. So,

$$TwoSum = |T|\, Pr_1/Z_1 + |T|\, Pr_2/Z_2. \quad (8)$$

APPENDIX B: SYMBOLS

$\vec{X}$	scene (3-D) vector consisting of 3 components, X, Y, Z
X	component of scene (3-D) vector
Y	component of scene (3-D) vector
Z	component of scene (3-D) vector
x	component of position in image
y	component of position in image
p	label for point in image
P	label for point in scene
$\vec{T}$	translation (3-D) vector; components are U, V, W
$\vec{\Omega}$	rotation (3-D) vector; angular velocity; components are A, B, C
U	component of translation (3-D) vector
V	component of translation (3-D) vector
W	component of translation (3-D) vector
A	component of rotation (3-D) vector
B	component of rotation (3-D) vector
C	component of rotation (3-D) vector
u	component of image motion vector
v	component of image motion vector
d/dt	derivative with respect to time
X'	coordinate of 3-D point after motion
Y'	coordinate of 3-D point after motion
Z'	coordinate of 3-D point after motion
x'	coordinate of image point after motion
y'	coordinate of image point after motion
R	rotation matrix
$r_1, \ldots, r_9$	components of rotation matrix
ω	generic component of angular velocity vector; stands for A, B, C
∂	partial derivative
x_f	component of location of FOE
y_f	component of location of FOE
α	the depth factor
$\overset{\text{def}}{=\!=}$	is computationally defined as
Sum	collinear triplet sum
m	distance between first and second point in collinear triplet
n	distance between second and third point in collinear triplet
θ	orientation of line in image plane
π	the irrational number
Δ	discrete difference
$\overline{P_1P_2}$	the line segment joining 3-D points P_1 and P_2
κ_x	difference between gradients of two limbs along Y
κ_y	difference between gradients of two limbs along X
$(\ )^T$	transpose
$\vec{s}$	direction vector
$\vec{n}$	vector perpendicular to $\vec{s}$
∇	gradient
$\cdot$	dot (or scalar) product
S	spacing within triplet
s	spacing between triplets
$LineSum$	the sum along a line
$Response$	the response of the FOE operator
$TotalTriplets$	total triplets used to compute a single response
Σ	summation
$\lvert\ \rvert$	absolute value
$r(x)$	curve interpolating response map
a	coefficient for interpolation
b	coefficient for interpolation
c	coefficient for interpolation
$x_{\min}$	coordinate of grid point with minimum response
$y_{\min}$	coordinate of grid point with minimum response
σ	standard deviation
$\vec{\kappa}$	difference vector of gradients of limbs; components κ_x, κ_y
$Z_{\max}$	furthest visible point in view
$Z_{\min}$	closest visible point in view
$C_{U,i}$, $C_{V,i}$, $C_{W,i}$	polynomials for computing *TwoSum*
Pr_i	projections for computing *TwoSum*

ACKNOWLEDGMENTS

The authors wish to thank Allan Jepson for his comments and suggestions during the course of this work. The second author is the CP-Unitel Fellow of the Canadian Institute for Advanced Research. This research was funded by the Information Technology Research Center, one of the Province of Ontario Centers of Excellence, the Institute for Robotics and Intelligent Systems, a Network of Centers of Excellence of the Government of Canada, and the Natural Sciences and Engineering Research Council of Canada.

REFERENCES

1. G. Adiv, Determining 3-D motion and structure from optical flow generated by several moving objects, *IEEE PAMI* **7,** 1985, 384–401.
2. A. Bandyopadhay, B. Chandra, and D. H. Ballard, Active navigation: Tracking an environmental point considered beneficial, *Proceedings, IEEE Workshop on Motion, Charleston, NC, 1986,* pp. 23–29.

3. J. Barron, D. J. Fleet and S. S. Beauchemin, Performance of optical flow techniques, RPL-TR-9107, Dept. of CIS, Queen's University, Canada, 1992.
4. A. R. Bruss and B. K. P. Horn, Passive navigation, *CVGIP* **21,** 1983, 3–20.
5. S. Culhane and J. K. Tsotsos, An attentional prototype for early vision, *Proceedings, Second European Conference on Computer Vision, Santa Margherita Ligure, Italy, May 1992* (G. Sandini, Ed.), LNCS-Series Vol. 588, pp. 551–560, Springer Verlag, Berlin/New York, 1992.
6. J. E. Cutting, *Perception with an Eye for Motion,* MIT Press, Cambridge, MA, 1986.
7. N. da Vitoria Lobo, Computing egomotion, shape, and detecting independent motion from image motion, Ph.D. dissertation, Dept. of Computer Science, University of Toronto, Dec. 1992.
8. N. da Vitoria Lobo and J. K. Tsotsos, Using collinear points to compute motion and detect non-rigidity, in *Proceedings, CVPR, Maui, HI, 1991,* pp. 344–350.
9. J. Frazier R. Nevatia, Detecting moving objects from a moving platform, in *Proceedings, DARPA-IU, Pittsburgh, PA, 1990,* pp. 349–355.
10. B. Hallert, *Photogrammetry,* McGraw Hill, New York, 1960.
11. D. Heeger, and A. Jepson, Simple method for computing 3-D motion and depth, in *Proceedings, ICCV, Osaka, Japan, 1990,* pp. 96–100.
12. B. K. P. Horn, Motion fields are hardly ever ambiguous, *IJCV* **1,** 1987, 259–274.
13. T. S. Huang and R. Y. Tsai, Image sequence analysis: Motion estimation, in *Image Sequence Analysis* (T. S. Huang, Ed.), Springer-Verlag, Berlin/New York, 1981.
14. R. Hummel and V. Sundereswaran, Motion parameter estimation from global flow field data, *IEEE PAMI,* to appear.
15. M. Irani and S. Peleg, Image sequence enhancement using multiple motions analysis, in *Proceedings, CVPR, Champaign, IL, 1992,* pp. 216–221.
16. R. Jain, W. N. Martin, and J. K. Aggarwal, Extraction of moving object images through change detection, in *Proceedings, IJCAI, Tokyo, Japan, 1979,* pp. 425–428.
17. R. Jain, D. Militzer, and H. H. Nagel, Separating non-stationary from stationary scene components in a sequence of real world images, in *Proceedings, IJCAI, Cambridge, MA, 1977,* pp. 616–618.
18. A. Jepson and D. Heeger, Subspace methods for recovering rigid motion II: Theory, *IJCV,* 1992.
19. A. Jepson and D. Heeger, Linear methods for recovering translational direction, RBCV-TR-92-40, Dept. of Computer Science, University of Toronto, 1992.
20. C. Jerian and R. Jain, Polynomial methods for structure from motion, in *Proceedings, ICCV, Tampa, FL, 1988.*
21. A. Källdahl, Simultaneous estimation of motion and shape from a sequence of feature points projections, Thesis No. 194, Department of Electrical Engineering, Linköping University, Sweden.
22. J. J. Koenderink and van Doorn, Local structure of movement parallax of the plane, *J. Opt. Soc. Am.* **66,** 1976, 717–723.
23. H. C. Longuet-Higgins, A computer program for reconstructing a scene from two projections, *Nature* **293,** 1981, 133–135.
24. H. C. Longuet-Higgins and K. Prazdny, The interpretation of a moving retinal image, *Proc. Roy. Soc. London B* **208,** 1980, 385–397.
25. E. Milios, M. Jenkin, and J. K. Tsotsos, Design and performance of TRISH, a binocular robot head with torsional eye movements, *Int. J. Pattern Recognit. Artif. Intell.* **7.1,** 1993, 51–68.
26. A. Mitiche, On kineopsis and computation of structure and motion, *IEEE PAMI* **8**(1) 1986, 109–112.
27. K. Nakayama and J. M. Loomis, Optical velocity patterns, velocity sensitive neurons and space perception: A hypothesis, *Perception* **3,** 1974, 63–80.
28. S. Negahdaripour and B. K. P. Horn, A direct method for locating the focus of expansion, *CVGIP* **46,** 1989, 303–326.
29. R. Nelson, Qualitative detection of motion by a moving observer, in *Proceedings, CVPR, Maui, HI, 1991,* pp. 173–178.
30. R. Nelson and J. Aloimonos, Using flow field divergence for obstacle avoidance: Towards qualitative vision, in *Proceedings, ICCV, FL, 1988,* pp. 188–196.
31. K. Prazdny, Determining the instantaneous direction of motion from optical flow generated by a curvilinearly moving observer, *CVGIP* **17,** 1987, 238–248.
32. K. Prazdny, On the information in optical flows, *CVGIP* **22**(2), 1983, 239–259.
33. D. M. Regan and Beverley, How do we avoid confounding the direction we are looking and the direction we are moving?, *Science* **215,** 1985, 194–196.
34. J. H. Reiger and P. T. Lawton, Processing differential image motion, *J. Opt. Soc. Am.* **A2,** 1985, 354–359.
35. K. Rektorys, *Survey of Applicable Math,* MIT Press, Cambridge, MA, 1969.
36. J. Roach and J. K. Aggarwal, Determining the movement of objects from a sequence of images, *IEEE PAMI* **2**(6), 1980, 554–562.
37. R. Sharma and J. Aloimonos, Visual motion analysis under interceptive behaviour, in *Proceedings, CVPR, Champaign, IL, 1992,* pp. 148–153.
38. M. Spetsakis and J. Aloimonos, Closed form solutions to the structure from motion problem from line correspondences, in *Proceedings, AAAI, Seattle, WA, 1987,* pp. 738–743.
39. V. Sundereswaran, A fast method to estimate sensor translation, in *Proceedings, ECCV-92, France, 1992,* pp. 253–257.
40. M. A. Taalebinezhaad, Direct recovery of motion and shape in the general case by fixation, *IEEE PAMI* **14**(8), 1992, 847–853.
41. W. B. Thompson, P. Lechleider, E. R. Stuck, Detecting moving objects using the rigidity constraint, *IEEE PAMI* **15**(2), 1993, 162–164.
42. W. B. Thompson and T.-C. Pong, Detecting moving objects, *IJCV* **4**(1), 1990, 39–57.
43. C. Tomasi and T. Kanade, Shape and motion without depth, in *ICCV, Osaka, Japan, 1990,* pp. 91–95.
44. R. Y. Tsai and T. S. Huang, Uniqueness and estimation of three-dimensional motion parameters of rigid objects with curved surfaces, *IEEE PAMI* **6**(1), 1984, pp. 13–27.
45. J. K. Tsotsos, An Inhibitory Beam for Attentional Selection, in *Spatial Vision in Humans and Robots* (L. Harris and M. Jenkin, Eds.), pp. 313–331, Cambridge Univ. Press, Cambridge, UK, 1993.
46. S. Ullman, *The Interpretation of Visual Motion,* MIT Press, Cambridge and London, 1979.
47. A. M. Waxman, An image flow paradigm, in *Proceedings, IEEE Workshop on Computer Vision: Representation and Control, Bellaire, MI, 1985,* pp. 49–57.
48. A. M. Waxman and K. Wohn, Image flow theory: A framework for 3-D inference from time-varying imagery, in *Advances in Computer Vision* (C. Brown, Ed.), Erlbaum, Hillsdale, NJ, 1987.
49. D. Weinshall, Direct computation of qualitative 3-D shape and motion invariants, in *ICCV, Osaka, Japan, 1990,* pp. 230–237.

50. J. Weng, T. S. Huang, and N. Ahuja, Motion and structure from two views: Algorithms, error analysis, and error estimation, *IEEE PAMI* **11**(5), 1989, 451–476.

51. J. Weng, Y. Liu, T. S. Huang, and N. Ahuja, A linear algorithm for motion and structure estimation using straight line correspondences, in *Proceedings, ICPR, Rome, Italy, 1988.*

52. Y. Yang and A. Yuille, Egomotion recovery using first order optical flow information, TR 91-19, Harvard Robotics Lab, Cambridge, MA, 1991.

53. Y. Yasumoto and G. Medioni, Robust estimation of 3-D motion parameters from a sequence of image frames using regularization, *IEEE PAMI* **8**(4), 1986.