The International Journal of Robotics Research
http://ijr.sagepub.com/

3D Motion and Shape Representations in Visual Servo Control

Cornelia Fermüller, LoongFah Cheong and Yiannis Aloimonos

The International Journal of Robotics Research 1998 17: 4
DOI: 10.1177/027836499801700103

The online version of this article can be found at:
http://ijr.sagepub.com/content/17/1/4

Published by:
SAGE
http://www.sagepublications.com

On behalf of:

Cornelia Fermüller
LoongFah Cheong
Yiannis Aloimonos
Computer Vision Laboratory
Center for Automated Research
Department of Computer Science
Institute for Advanced Computer Studies
University of Maryland
College Park, Maryland 20742-3275 USA

3D Motion and Shape Representations in Visual Servo Control

Abstract

The study of visual navigation problems requires the integration of visual processes with motor control. Most essential in approaching this integration is the study of appropriate spatiotemporal representations that the system computes from the imagery and that serve as interfaces to all motor activities. Since representations resulting from exact metric reconstruction of the environment have turned out to be very hard to obtain in real time, the authors argue for the necessity of representations that can be computed easily, reliably, and in real time and that recover only the information about the 3D world that is really needed to solve the navigational problems at hand. In this paper, the authors introduce a number of such representations capturing aspects of 3D motion and scene structure that are used to solve navigational problems implemented in visual servo systems.

1. Introduction

A continuous interplay between visual processing and motor activity is a characteristic of most existing systems that interact with their environments. Initial attempts at these difficult problems followed a modular approach. The goal of computational vision was defined as the reconstruction of an accurate description of the system's spatiotemporal environment. Assuming that this information can be acquired exactly, sensory feedback robotics was concerned with the planning and execution of the robot's activities. The problem with such separation of perception from action was that both computational goals turned out to be intractable (Aloimonos 1990; Aloimonos, Weiss, and Bandopadhay 1988; Bajcsy 1988; Ballard and Brown 1992). Recently, a number of studies have been published that argue for a closer coupling by means of achieving solutions to a number of specialized visuomotor control problems (Raviv and Herman 1993; Santos-Victor et al. 1993). We also encounter the so-called approach of image-based control (Arkin et al. 1989; Espiau, Chaumette, and Rives 1992; Skaar, Brockman, and Hanson 1987; Weiss and Sanderson 1987). The principle behind this approach is that instead of computing intermediate representations, directly available image measurements are used as feedback for the control loop. Most commonly, a number of feature points extracted from the image are tracked over time. The idea behind this is that the chosen image features can be directly related to the parameters of the robot's joints through the so-called kinematic map. However, image features of this kind are neither easily extractable nor easily tracked, although recently some interesting techniques for this problem have appeared, and they work well in restricted domains (Murray et al. 1995; Reid and Murray 1993; Hollinghurst and Cipolla 1993; Hager, Chang, and Morse 1994; Beardsley et al. 1995). In addition, even simple kinematic maps are no longer simple when it comes to inverting them.

Here we argue that from the viewpoint of computational perception, the essence of understanding the coupling of perception and action (Brady 1985; Fermüller and Aloimonos 1995e), will come from understanding the appropriate spatiotemporal representations that the system computes from the imagery. By considering two visuoservo motor control tasks, we present representations for 3D rigid motion and shape. In both tasks, we consider a robot system consisting of a body and a camera that can move independently of the body. The first task consists of changing the robot's direction of motion toward a fixed direction using visual information. For this task, we need only partial egomotion information about the robot. In the second task, the robot has to follow a perimeter. This task requires the system to obtain partial depth information about the perimeter. The depth representation employed is less complex than the classical ones of (scaled) distance that are usually used. It is a function of scaled shape that can be derived without first computing 3D

The International Journal of Robotics Research,
Vol. 17, No. 1, January 1998, pp. 4–18,

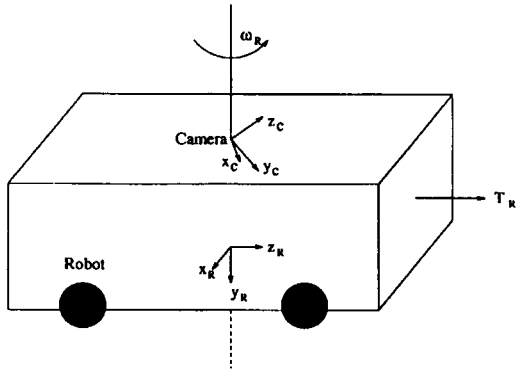


Fig. 1. Camera and robot local coordinate systems.

motion. In particular, we employ an "ordinal depth" representation (Fermüller and Aloimonos 1995d, 1996) that lies in the spectrum between projective and Euclidian representations (Faugeras 1995; Koenderink and van Doorn 1995).

2. Task 1: Moving Toward a Fixed Direction

The robotic system considered in both tasks consists, as illustrated in Figure 1, of a body on wheels with a camera on top of the body. To describe the system's degrees of freedom, we define two coordinate systems attached to the camera and the robot. The robot moves on a surface and is constrained in its movement to a forward translation T_R (along z_R) and a rotation ω_R around the vertical axis; that is, the y_R-axis. The camera is positioned along a vertical axis that passes through the center of rotation of the robot, and it has two independent rotational degrees of freedom. Its orientation measured with respect to the coordinate frame of the robot is given by its tilt, θ_x, and its pan, θ_y, which are unknown; the roll denoted by θ_z is zero.

We first consider the simple task of moving toward a new direction. Referring to the mobile robot illustrated in Figure 2, the problem can be stated as follows. A robot, moving forward with speed S, is required to head toward a new direction, along which some visual feature **p** lies; **p** is selected beforehand by some higher level process. The robot first directs the camera at **p** so that the line of sight is now positioned along **p**. We assume such gaze shifts are accomplished by fast saccadic movements.

The robot must now make a series of steering decisions so that eventually its heading direction is aligned with where the camera is pointing. To be more accurate, since the robot moves on a surface and the feature in view can be at any height, we only want the direction of the forward motion and the direction of the heading to have the same projection on the xz-plane of the camera coordinate system; that is, θ_y has to become zero. These steering movements are controlled by

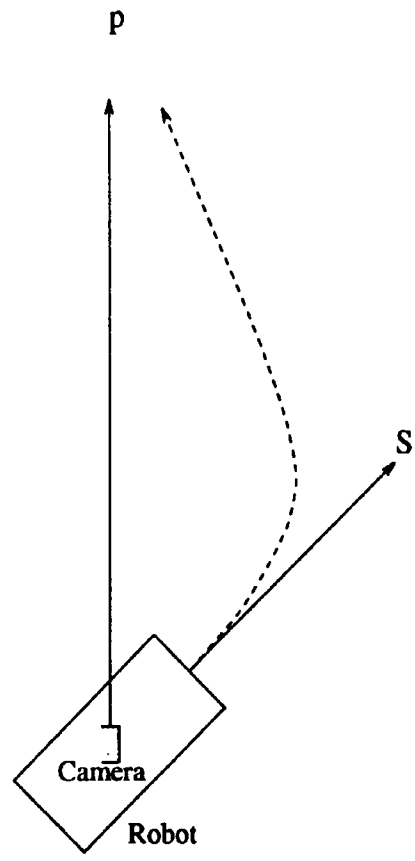


Fig. 2. The robot, currently moving forward with speed S, aims to veer toward **p**. The dotted path represents the trajectory generated in flight by the servo system.

a servomechanism, which derives information from images captured by the camera. A similar problem was addressed by Barth and Tsuji (1993), except that Barth and Tsuji aligned the camera with the direction of motion and used point matches.

The next section will be devoted to a discussion of the visual features, or more exactly, the visual patterns, employed by the servomechanism and the manner in which they are used. It will be followed by an analysis of the servomechanism itself.

2.1. *Global Motion Patterns*

The visual input that has been used to describe the computational analysis of visual motion is the optic flow field, which stems from the movement of light patterns in the image plane. Optic flow fields constitute a good approximation to the projection of the real 3D motion at scene points on the image (Singh 1990; Verri and Poggio 1987). In general, however, accurate values of optic flow are not computable. On the basis of local information, only the component of optic flow perpendicular to edges, the so-called normal flow, is well defined (the aperture problem; see Fig. 3). In many

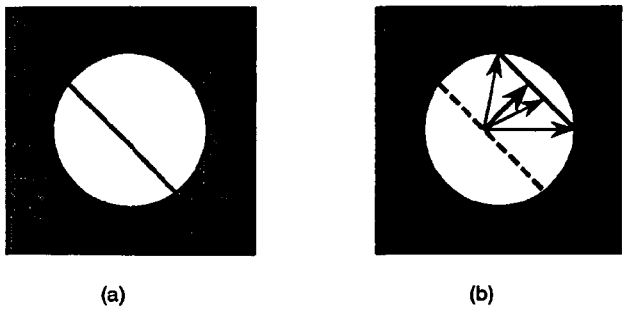


Fig. 3. (a) Line feature observed through a small aperture at time t. (b) At time $t + \delta t$, the feature has moved to a new position. It is not possible to determine exactly where each point has moved to. From local measurements, only the flow component perpendicular to the line feature can be computed.

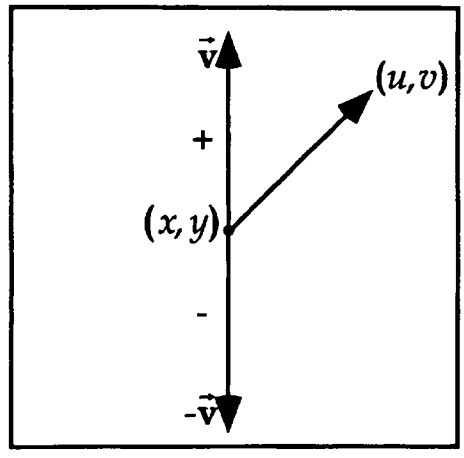


Fig. 4. At an image point (x, y), measurement of image motion along a vector $\vec{v}$ amounts to finding the projection of the local motion vector (u, v) on $\vec{v}$. Measurement of the sign of the image motion along $\vec{v}$ amounts to finding whether the angle between (u, v) and $\vec{v}$ is greater or less than 90°.

cases, it is possible to obtain additional flow information for areas (patches) in the image. Thus, the input that can be used by perceptual systems for further motion processing is some partial optic flow information.

The constraints that have been used in earlier work are mostly local ones. However, the use of image measurements from only small image regions is extremely error prone; for example, image measurements are very hard to compute accurately. Even if we just compute the normal flow, the projection of the retinal motion on the local image gradients, we need to use infinitesimal computations and have to approximate derivatives by difference quotients, and thus, our computations can only be approximations. Much more difficult, however, is the computation of optic flow or disparity measurements, which requires us to employ some additional assumptions, usually smoothness assumptions, and thus, we run into problems at motion and depth boundaries. On the other hand, even if we had reasonably accurate flow, it would not be the case that a small local change in flow implies a small change in 3D motion. Completely different camera geometries produce locally similar disparity fields. For example, in an area near the vertical axis in the image plane, 3D rotation around the horizontal axis in the 3D world produces a flow field similar to the one produced by translation along the vertical axis in the 3D world.

Also, if we seek to provide solutions for real-time systems, we have to consider time constraints imposed on the actions the system performs. Computations that could not possibly be performed fast, such as linear, nonparallelizable algorithms that require a large number of steps or optimization techniques involving a large number of iterations, pose some problems. In this category falls the computation of exact image displacements. Both the estimation of discrete disparities and the computation of optic flow require optimization techniques to be invoked; and the more effort we put into deriving accurate measurements, by considering realistic situations dealing with motion boundaries and thus modeling discontinuities, the more computational steps we have to perform.

We seek a representation of image motion that is global, robust, and achievable in real time. The most robust local measurement of image motion that we can make use of is the sign of flow along some directions. Consider a point (x, y) on the moving image, as in Figure 4, and pick any orientation $\vec{v}$ on the image plane. Then, the sign of image motion along $\vec{v}$ is either positive, negative, or zero. This means that the angle between the actual motion vector at point (x, y) and the vector $\vec{v}$ is either less than 90°, greater than 90°, or equal to 90°. The estimation of the sign of image motion along a direction can be easily achieved using simple differencing methods, Reichardt-like correlators, or equivalent energy models (Reichardt 1961, 1987; Poggio and Reichardt 1973; van Santen and Sperling 1984). Computational considerations tell us that robust results can be obtained for at least one direction; namely, the direction of the image spatial gradient (i.e., we are guaranteed to find the sign of the image motion at least along the direction of gradient, or the sign of the normal flow). In many cases, it is possible to compute the sign along many more directions.

Thus, since the sign of local motion along a direction is a robust measurement and since we seek global representations, it is natural to work with global representations of these local quantities. Hence, a global representation we can work with is a pairing of an *orientation field* and a map of $+$, $-$, or 0s associated with each vector in the field (Fig. 5), denoting the sign of the local motion vector along the corresponding vector of the orientation field. In the following, we use such data structures for 3D motion estimation. As has been shown (Fermüller 1995; Fermüller and Aloimonos

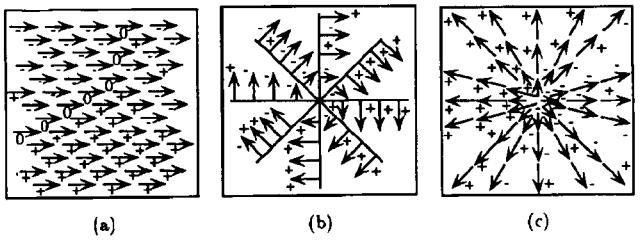


Fig. 5. Global spatiotemporal representations. We consider a vector field (in (a), a parallel horizontal field; in (b), a field such that, at every point, the vectors are normal to the line connecting the point with some center; in (c), the vectors are radial) and we label each vector of the field with the sign of the image motion along that vector. The sign depends on the angle between the local image motion vector and the vector of the orientation field. The choice of an orientation field like (a), (b), or (c) is vital to addressing particular tasks.

1995a, 1995b, 1995c), if the orientation fields are chosen in particular ways, we obtain patterns of motion vectors that are invariant to the 3D environment. In the following sections, we will use some of these patterns; namely, the copoint and coaxis patterns defined on the coaxis and copoint vector fields. A description of the corresponding constraints is given below.

2.2. *Copoint and Coaxis Vectors*

The 2D motion field on an imaging surface is the projection of the 3D motion field of the scene points moving relative to that surface. If this motion is rigid, it is composed of a translation $\mathbf{t} = (U, V, W)$ and a rotation $\boldsymbol{\omega} = (\alpha, \beta, \gamma)$. We consider the case of a moving camera in a stationary environment with the coordinate system fixed to the nodal point of the camera and the image projection on a plane perpendicular to the Z-axis at distance f (focal length) from the center. Introducing the coordinates $(x_0, y_0) = (\frac{Uf}{W}, \frac{Vf}{W})$ for the direction of translation (i.e., the projection of the translation axis onto the image), the so-called epipole, we obtain the following well-known equations relating the velocity $\mathbf{u} = (u, v)$ of an image point to the 3D velocity and the depth Z of the corresponding scene point (Longuet-Higgins and Prazdny 1980):

$$\begin{aligned} u &= u_{\text{trans}} + u_{\text{rot}} \\ &= (-x_0 + x)\frac{W}{Z} + \alpha\frac{xy}{f} - \beta(\frac{x^2}{f} + f) + \gamma y \qquad (1) \\ v &= v_{\text{trans}} + v_{\text{rot}} \\ &= (-y_0 + y)\frac{W}{Z} + \alpha(\frac{y^2}{f} + f) - \beta\frac{xy}{f} - \gamma x. \qquad (2) \end{aligned}$$

For the case of pure translation, the epipole is also known as the focus of expansion (FOE), but the term FOE is widely used to refer to the projection of the translation axis onto the image. In the sequel, we use the term FOE.

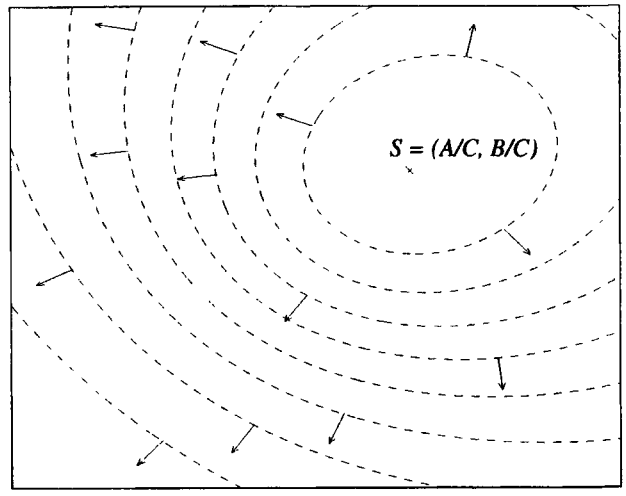


Fig. 6. Field lines corresponding to an axis (A, B, C) and positive (A, B, C)-coaxis vectors.

We consider the flow along certain directions, that is, the projection $\mathbf{u_n}$ of the flow $\mathbf{u}$ on direction $\mathbf{n} = (n_x, n_y)$ (unit vector), which is given as

$$\mathbf{u_n} = (\mathbf{u} \cdot \mathbf{n})\mathbf{n}.$$

Thus, we obtain u_n for the value of the vector $\mathbf{u_n}$ along the direction $\mathbf{n}$:

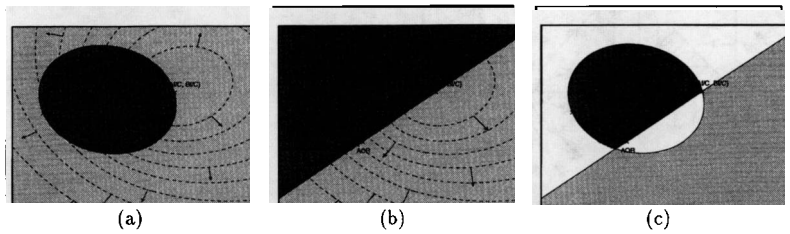
$$\begin{aligned} u_n &= \tfrac{W}{Z}((x - x_0)n_x + (y - y_0)n_y) \\ &\quad (\alpha\tfrac{xy}{f} - \beta(\tfrac{x^2}{f} + f) + \gamma y)n_x \qquad (3) \\ &- (\alpha(\tfrac{y^2}{f} + f) - \beta\tfrac{xy}{f} - \gamma x)n_y. \end{aligned}$$

By choosing particular directions, we define classes of vectors, such as the ones in Figure 5. In particular, we consider motion vectors in the direction of two classes of vectors: the coaxis vectors and the copoint vectors.

The coaxis vectors, defined with respect to a direction in space, are described as follows: A line through the image formation center defined by the direction cosines (A, B, C) defines a family of cones with axis (A, B, C) and apex at the origin. The intersections of the cones with the image plane give rise to a set of conic sections, called field lines of the axis (A, B, C), and the vectors perpendicular to the conic sections are called the (A, B, C)-coaxis vectors. Their direction is parallel to the vector (m_x, m_y), which is defined as

$$\begin{aligned} (m_x, m_y) &= ((-A(y^2 + f^2) + Bxy + Cxf) \\ &\quad (Axy - B(x^2 + f^2) + Cyf)). \end{aligned} \qquad (4)$$

To establish conventions about the vector's orientation, a vector will be said to be of positive orientation if it is pointing in direction (m_x, m_y). Otherwise, if it is pointing in direction $(-m_x, -m_y)$, its orientation will be said to be negative (see Fig. 6).



(a) (b) (c)

Fig. 7. (a) The (A, B, C)-coaxis vectors due to translation are negative if they lie within a second-order curve defined by the FOE and are positive at all other locations (dark gray areas are negative, light gray areas are positive). (b) The coaxis vectors due to rotation separate the image plane into a half-plane of positive values and a half-plane of negative values. (c) A general rigid motion defines an area of positive coaxis vectors and an area of negative coaxis vectors. The coaxis vectors can take the value zero only in the remaining area (white).

If we consider for a class of (A, B, C)-coaxis vectors the sign of the translational flow along these vectors, we find that a second-order curve $h_A(A, B, C, x_0, y_0; x, y) = 0$ (Fig. 7a) separates the positive from the negative components, where

$$\begin{aligned} h_A(A, B, C, x_0, y_0; x, y) &= (x - x_0, y - y_0) \cdot (n_x, n_y) \\ &= x^2(Cf + By_0) + y^2(Cf + Ax_0) - xy(Ay_0 + Bx_0) \\ &-xf(Af + Cx_0) - yf(Bf + Cy_0) + f^2(Ax_0 + By_0). \end{aligned} \quad (5)$$

Curve $h_A = 0$ passes through the FOE and is uniquely defined by the FOE's two image coordinates (x_0, y_0). Similarly, the positive and negative components of the (A, B, C)-coaxis vectors due to rotation are separated by a straight line $g_A(A, B, C, \alpha, \beta, \gamma; x, y) = 0$, where

$$\begin{aligned} g_A(A, B, C, \alpha, \beta, \gamma; x, y) = y(\alpha C - \gamma A) - x(\beta C - \gamma B) \\ + \beta Af - \alpha Bf. \end{aligned} \quad (6)$$

The line passes through the point where the rotation axis pierces the image plane (Fig. 7b). The point, whose coordinates are $(\frac{\alpha f}{\gamma}, \frac{\beta f}{\gamma})$, is called the axis of rotation point (AOR). Combining the constraints due to translation and rotation, we obtain the following geometrical result: A second-order curve separating the plane into positive and negative values and a line separating the plane into two half-planes of opposite sign intersect. This splits the plane into areas of only positive coaxis vectors, areas of only negative coaxis vectors, and areas in which the rotational and translational flow have opposite signs. In these last areas, there exist both positive and negative vectors, depending on the depth of the scene in view (Fig. 7c). Since only in these areas do g_A and h_A have opposite sign (i.e., $g_A \cdot h_A < 0$), these are the only areas where the copoint vectors can take the value zero. The structure defined on the coaxis vectors is called the coaxis pattern.

For a second kind of classification—the copoint vectors, which are defined with respect to a point—similar patterns are obtained. The (r, s) copoint vectors are the vectors perpendicular to straight lines passing through the point (r, s) (see Fig. 8). At point (x, y), an (r, s) copoint vector (o_x, o_y) of unit length in the positive direction is defined as

$$(o_x, o_y) = \frac{(-y + s, x - r)}{\sqrt{(x - r)^2 + (y - s)^2}}. \quad (7)$$

For the copoint vectors, the rotational components are separated by a second-order curve into positive and negative values and the translational components are separated by a straight line. The structure defined on the copoint vectors is called the copoint pattern.

Of particular interest for this application are the (r, s) copoint vectors for which the copoint (r, s) lies in infinity. The corresponding copoint vectors are all parallel to each other with gradient $\frac{n_x}{n_y} = -\frac{s}{r}$ (see Figs. 9a, 9b). For these cases, the line separating the translational components is perpendicular to the gradient vector (n_x, n_y) and has the following form:

$$g_P(n_x, n_y, x_0, y_0; x, y) = (x - x_0)n_x + (y - y_0)n_y. \quad (8)$$

The curve separating the rotational components is a hyperbola given by

$$\begin{aligned} h_P(n_x, n_y, \alpha, \beta, \gamma; x, y) = (\alpha\frac{xy}{f} - \beta(\frac{x^2}{f} + f) + \gamma y)n_x \\ +(\alpha(\frac{y^2}{f} + f) - \beta(\frac{y^2}{f} + f) - \gamma x)n_y. \end{aligned} \quad (9)$$

The positions of the coaxis and copoint vectors in the image plane encode the parameters describing the axis of translation and the direction of the rotation axis. Thus, these constraints lead to formulating the problem of 3D motion estimation as a pattern recognition problem. If the system has the capability of estimating the sign of the flow along the directions defined by various families of coaxis (or copoint) vectors, then by localizing a number of patterns or, more precisely, the boundaries of the regions separating positive and negative vectors, the system can find the axes of translation and rotation. The intersection of the coaxes' second-order curves

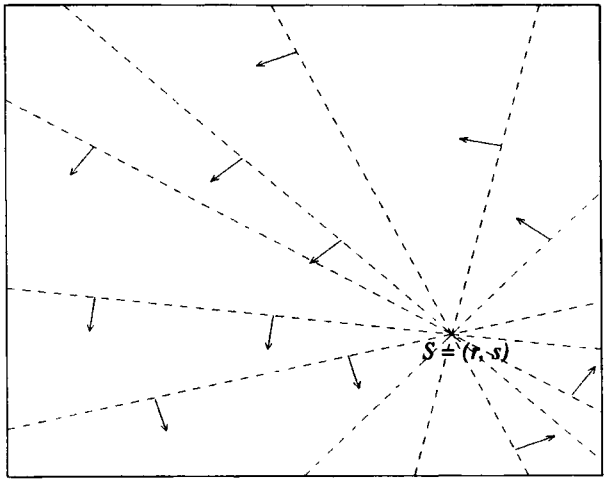
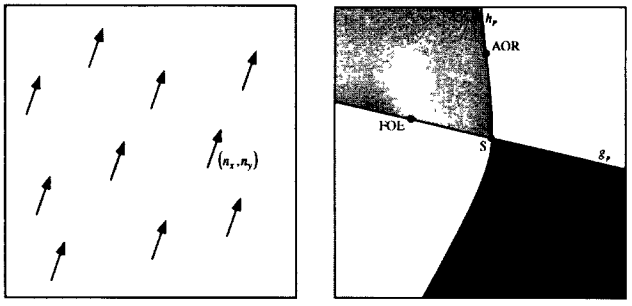


Fig. 8. Positive copoint vectors (r, s).



(a) (b)

Fig. 9. (a) Parallel copoint vectors. (b) Corresponding pattern. Dark gray areas are positive, light gray areas are negative. In the white area positive, negative, and zero value vectors are found.

and the copoints' lines provides the FOE, and the intersection of the coaxes' lines and the copoints' curves provides the AOR.

How much information will be available for pattern fitting and, thus, how accurately the FOE and AOR can be localized, depends on the computability of flow information. If the system is able to derive optic flow then it is able to estimate the sign of the projection of flow along any direction, and thus, for every pattern at every point, information is available. If, however, the system is less powerful and can compute the sign of the flow in only one or a few directions, then patterns are matched as before. The difference is that information is not available for every point, and consequently, the uncertainty in pattern matching may be larger, and the FOE and AOR can only be located within bounds. In the simplest case, which requires the least amount of computational effort, the flow in only one direction, namely, the one perpendicular to the local edge, is computed (the normal flow). But even this

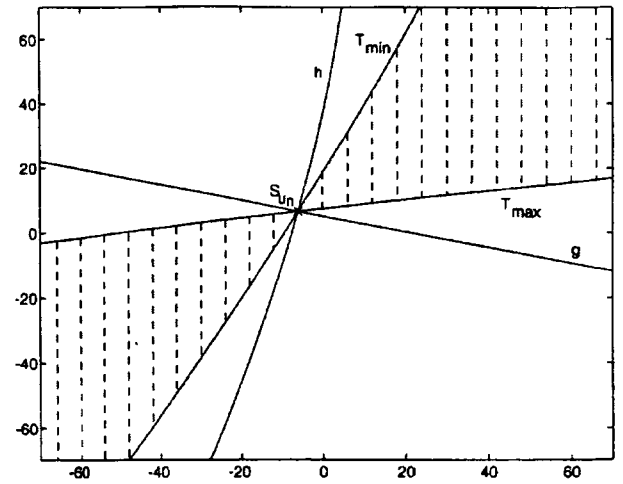


Fig. 10. Bounds on the value W/Z constrain the motion vectors that are zero along a direction to an area defined by two hyperbolas, the so-called motion band.

minimal amount of information can lead to rather small uncertainty in the motion estimation (Fermüller and Aloimonos 1995a, 1995c).

If the rigid motion estimation problem is considered for a passive system, we could address the localization of the patterns as a simple search problem, which means that for every possible motion we would check whether the constraints are satisfied for the available motion measurement. Every single pattern is defined by three unknown parameters (a second-order curve of two unknowns and a line of one unknown). A general rigid motion for which no information is available that could reduce the space of possible solutions thus requires a search in various 3D subspaces. However, it has been shown in our previous work (Fermüller and Aloimonos 1994, 1995b) that by exploiting the particular structure of the positive and negative vectors in each pattern, the search can be decreased considerably. Furthermore, if the system is active, and if it has the capability to control its motor apparatus, the constraints described above may be used in much more efficient ways.

The work most closely related to ours is that by the Oxford group who have demonstrated real-time robotic response to visual stimuli, including saccades, smooth pursuit, OKR, and time-to-contact, all from normal components of optic flow (Murray et al. 1993; Murray et al. 1995; Bradshaw et al. 1994).

2.3. *Using Visual Patterns in the Servomechanism of a Moving System*

In the most general case where the robot is moving with an unrestricted 3D motion, to steer the robot it is necessary to derive the angle between the robot's translation and its gaze

direction. Thus, the input to the servomechanism computed from the imagery is the position of the focus of expansion—or a small uncertainty area containing the FOE. As mentioned before, such an area can be found using simple search techniques. A possible area for the FOE can be determined by exploiting the particular structure of the areas that contain both positive and negative values, as well as vectors of value zero, in the copoint pattern. To simplify matters, let us concentrate on the parallel copoint vectors only; that is, vectors of constant direction (n_x, n_y). In most situations, the depth of the scene in view is not completely unconstrained but lies within a range, and thus, W/Z is constrained. If, in eq. (3), we substitute $T_{\min}$ for the minimum value and $T_{\max}$ for the maximum value of W/Z, and we set u_n to zero, we obtain two equations of hyperbolas. These equations define the boundaries of the area in which motion vectors in direction (n_x, n_y) of length zero can be found. We refer to these areas as motion bands (see Fig. 10). The motion bands consist of two areas meeting at the point $\mathbf{S}_{\mathbf{u}_n}$. Through $\mathbf{S}_{\mathbf{u}_n}$ and the FOE passes the line g_P, which also separates the translational motion components (Fig. 9b). The slope of the line is known; it is perpendicular to the direction of the motion vector (n_x, n_y). The exact position of the line is defined by the exact position where the band is thinnest. We can approximate these bands by localizing the vectors of value zero or, similarly, patches, where both positive and negative vectors occur. It may not be possible to locate the exact line, only a bounded area that contains the line. The intersection of at least two such bounded areas corresponding to motion vectors in different directions gives an area in which the FOE lies. This is demonstrated in Figure 11 using a synthetic normal flow field. The localization of the FOE using these constraints could be implemented in a very simple way with a Hough transform-like scheme; we partition the image into a large number of cells. For every flow direction (n_x, n_y) considered, we check whether a cell contains vectors of value zero (or equally both positive and negative values). If not, the cell votes for a line passing through the cell perpendicular to (n_x, n_y); that is, we increase a counter in all cells containing that line. After voting in all cells for all directions, the FOE is found in the cells with highest counters.

If instead of using only vectors of values greater than, less than, or equal to zero, we also consider vectors greater or smaller than some threshold v_n, we obtain similar structures and patterns, and we can use much more data and additional constraints for the estimation of the motion (Fermüller and Aloimonos 1994).

But a situation in which the robot moves with a general 3D motion rarely arises in specific systems. In the sequel, we consider a situation in which the motion of the robot is not totally unconstrained, and we present a solution and an actual implementation of the problem of Task 1 using the global representations introduced before.

The retinal motion field perceived by the robot's camera is due to translation and rotation. The direction of translational motion is defined by the angle between the direction in which the robot is moving and the direction in which the camera is pointing. The rotation originates from body motion and is mainly due to the robot's turning around the y-axis. There could also be some rotation around the x-axis because the surface on which the robot is moving might be uneven, but there will be no or only very small rotation around the z-axis (cyclotorsion).

Recall that the goal of the visual task is to change the robot's motion such that the direction of the forward motion and the direction of the heading have the same projection on the xz-plane of the camera coordinate system. Stated in terms of motion parameters, this means that we want the x-coordinate of the FOE to be zero, but we do not care about the y-coordinate.

Let us now investigate the patterns of positive and negative flow vectors that correspond to such motion. We first consider the copoint patterns with parallel motion vectors. If the FOE is on the y-axis (i.e., $x_0 = 0$), (8) describing the translational vectors becomes

$$k(n_x, n_y, 0, y_0; x, y) : xn_x + (y - y_0)n_y = 0$$

or

$$y = -x\frac{n_x}{ny} + y_0, \tag{10}$$

which constitutes a line perpendicular to the gradient (n_x, n_y) with intercept y_0. In particular for the horizontal gradient direction $(n_x = 1, n_y = 0)$, we obtain the simplified equation

$$k(1, 0, 0, y_0; x, y) : x = 0.$$

During the process of steering while the FOE is not aligned with the y-axis, the flow field due to translation is separated into positive and negative vectors through

$$k(n_x, n_y, x_0, y_0; x, y) : y = -x\frac{n_x}{ny} + y_0 + x_0\frac{n_x}{n_y},$$

and the horizontal flow vectors are separated by

$$k(1, 0, x_0, y_0; x, y) : x = x_0.$$

Tracking is achieved by an independent servo loop implementing (Pahlavan 1993). The rotation around the y-axis is controlled by the robot's steering mechanism. Therefore, the robot is knowledgeable of this rotation and the tracking rotation, and can compensate for the resulting flow component perceived on the image by subtracting it from the visual motion field. Of course, we cannot assume the exact amount of rotation around the y-axis, but we can assume that we know a good approximation to it. This additional knowledge makes one of the patterns,

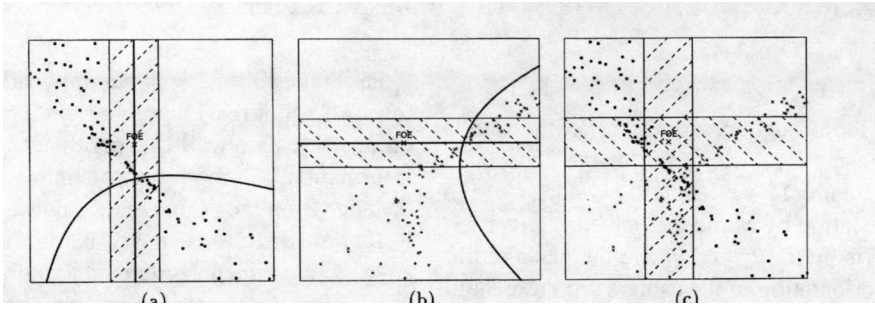


Fig. 11. Localization of FOE: (a) Motion band due to normal motion vectors of length zero parallel to the x-axis and corresponding curves $g(x, y)$ and $h(x, y)$, defining the boundaries of the positive and negative areas. By localizing where this motion band is thinnest, a bounded area for the line separating the translational normal motion components is found (marked by diagonal lines). (b) Motion band due to normal motion vectors of length zero parallel to the y-axis with overlaid curves $g(x, y)$ and $h(x, y)$, and localization of bounded area for the line separating the translational normal motion components (marked by diagonal lines). (c) The intersection of these areas gives a bounded area for the FOE.

namely, the copoint pattern with gradient (1, 0), particularly suitable for fast estimation of the parameters to be controlled.

Let us consider the rotational components of the copoint pattern with gradient $(1, 0)$. Within the visual field of view, the rotation around the x-axis gives rise to flow vectors that are mostly parallel to the y-axis and thus are perpendicular to the chosen gradients. As a result, the components along the gradient direction $(1, 0)$ are close to zero. The remaining (not derotated) rotation around the y-axis is nearly parallel to the gradients and thus causes a small, nearly constant component to be added to every flow vector. In summary, the contribution of the rotation to the pattern can be described as follows: The line in the translational pattern will be shifted by a small amount in the direction defined by the sign of the (not derotated) rotation around the y-axis.

For the purpose of the servoing task, it will be sufficient to approximate the copoint pattern with gradient $(1, 0)$ by its translational flow field components. Using this approximation gives us the advantage of deriving the x-component of the FOE with very little effort; we just fit a line perpendicular to the gradient direction separating positive from negative vectors. This approximation will not affect the successful accomplishment of the task. As the robot approaches its goal, the steering motion it has to apply becomes smaller and smaller, and thus, the additional rotational flow field component also decreases, which in turn allows the FOE to be estimated more accurately.

If it is certain that the rotation around the x-axis is also very small, then any other copoint pattern with some gradient (n_x, n_y) could be used in addition to estimate the FOE's coordinates using the approximation of considering the pattern to be translational.

Instead of using copoint vectors, we could equally well employ a class of coaxis vectors, namely, those that correspond to axes in the XY-plane, the $(A, B, 0)$ coaxis patterns. For these patterns, the hyperbola separating the positive from the negative translational vectors becomes

$$\begin{aligned} h(A, B, 0, x_0, y_0; x, y) :& \\ & By_0x^2 + Ax_0y^2 - (Ay_0 + Bx_0)xy - Af^2x \\ & - Bf^2y + f^2Ax_0 + Bf^2y_0 = 0. \end{aligned} \tag{11}$$

Since within the field of view, f^2 is much larger than the quadratic terms in the image coordinates (x^2, y^2, xy), (11) can be approximated by $\hat{h}$ as

$$\hat{h}(A, B, 0, x_0, y_0; x, y) : f^2(-Ax - By + Ax_0 + By_0) = 0$$

or

$$y = \frac{A}{B}x_0 + y_0 - \frac{A}{B}x,$$

which describes a line with slope $\frac{-A}{B}$ and intercept $\frac{A}{B}x_0 + y_0$.

If $x_0 = 0$, the intercept is y_0. Again, one of these patterns, which allows us to directly derive x_0, is of particular interest. This is the pattern corresponding to the axis $(1, 0, 0)$. We call this pattern the α-pattern and the corresponding vectors the α-vectors, since they do not contain any rotation around the x-axis (denoted in the equations by α):

$$h(1, 0, 0, x_0, y_0; x, y) : x_0y^2 - y_0xy - xf^2 + x_0f^2 = 0,$$

which simplifies to

$$x = x_0.$$

The α-vectors do not contain any rotation around the x-axis, and the components due to rotation around the y-axis are nearly constant. As in the previous case, we can approximate the α-pattern by its translational component, which is of a particular simple form. And again, if we know that rotation around the x-axis is small, to obtain more data we can employ many $(A, B, 0)$ coaxis patterns in the estimation of the FOE.

2.4. Servo System

To set up the control loop equation, we must relate the robot's motion to the image motion. Referring to the coordinate systems defined in Figure 1, we denote the velocity of the robot's forward translation by S and the velocity of its rotation around the y-axis by β. θ_x and θ_y are the pan and tilt, respectively, of the orientation of the camera with respect to the coordinate frame of the robot. Using the subscripts R and C to denote the robot and the camera, respectively, we can express the motion of the robot in the camera's coordinate system as follows. First, let $\vec{P}$ be the position vector that relates the origins of the two coordinate frames. The rotation matrix ${}^{C}R_R$ that relates the orientations of the frames is of the following form:

$$\begin{pmatrix} \cos\theta_y & 0 & \sin\theta_y \\ -\sin\theta_y \sin\theta_x & \cos\theta_x & \cos\theta_y \sin\theta_x \\ -\sin\theta_y \cos\theta_x & -\sin\theta_x & \cos\theta_y \cos\theta_x \end{pmatrix}.$$

Using T to denote translation and ω to denote rotation, we can express the motions of the robot and the camera in their respective coordinate frames as follows:

$$\vec{T_R} = (0, 0, S)^T \qquad \vec{\omega_R} = (0, \beta, 0)^T$$

$$\begin{aligned} \vec{T_C} &= {}^{C}R_R(\vec{T_R} + \vec{\omega_R} \times \vec{P}) \\ &= (S\sin\theta_y, S\cos\theta_y \sin\theta_x, S\cos\theta_y \cos\theta_x)^T \\ \vec{\omega_C} &= {}^{C}R_R \vec{\omega_R} = (0, -\beta\cos\theta_x, \beta\sin\theta_x)^T. \end{aligned}$$

Thus, the coordinates of the FOE (x_0, y_0) that we computed for the camera motion are related to pan and tilt as follows:

$$x_0 = \frac{\tan\theta_y}{\cos\theta_x} f \qquad y_0 = \tan\theta_x f. \tag{12}$$

The position of x_0 is used as the input to the servo system to control the amount of steering the robot has to perform. If the servo system is operated with a proportional controller, the rotational speed β of the robot will be given by $\beta = Kx_0$. Writing β as $\frac{d\theta_y}{dt}$ and substituting (12) for x_0, we obtain $\frac{d\theta_y}{dt} = \frac{\tan\theta_y}{\cos\theta_x} f$. Approximating $\tan\theta_y$ by θ_y and $\cos\theta_x$ by 1, we obtain a linear control equation $\beta = \frac{d\theta_y}{dt} = K\theta_y$.

3. Task 2: Perimeter Following

3.1. Estimating Functions of Depth

For many visual tasks requiring some depth or shape information, instead of computing exact depth measurements, it may be sufficient to compute less informative descriptions of shape and depth, such as functions of depth and shape where the functions are such that they can be computed easily from well-defined image information. This idea is demonstrated here by means of the task of perimeter or wall following.

Perimeter following in our application is described as follows: A robot (car) is moving on a road which is bounded on one side by a wall-like perimeter. On the basis of visual information, the robot has to control its steering to keep its distance from the perimeter at a constant value and maintain its forward direction as nearly parallel to the perimeter as possible. The perimeter is defined as a planar textured structure in the scene (connected or not) perpendicular to the plane of the road.

Usually, perimeter following is addressed either through general motion and depth reconstruction or by computing the slopes of lines parallel to the road (boundary lines on highways), which means that the boundary first has to be detected, and thus, the segmentation problem has to be solved.

The strategy applied here to the perimeter following task is as follows. While the robot is moving forward, it has its camera directed at some point on the perimeter. As it continues moving, it maintains the relative orientation of the camera with regard to its forward translation. It compares distance information derived from flow fields obtained during its motion with distance information computed from a flow field obtained when it was moving parallel to the road. This distance information will reveal what the robot's steering direction is with respect to the perimeter.

The distance information we use is the scaled directional derivative of inverse depth along (imaginary) lines on the perimeter. From the observed flow field, normal flow measurements along (imaginary) lines through the image center are selected and compared to normal flow measurements along (imaginary) lines of equal slope in the reference flow field (Fig. 12). The details of the computations are outlined below.

3.2. Direct Visual Depth Cue

As the robot is moving along its path, the motion parameters perceived in the images change. For comparison reasons, we assume that the angle θ_y and the angle θ_x between the forward direction and the camera direction (determining x_0 and y_0) remain constant. The robot's rotational velocity (around the x-axis and y-axis) can change in any way. The technique, however, is independent of these parameters. Next, we investigate the motion fields perceived during motion and how depth is encoded in the flow values.

The motion perceived in the images is due to a translation (U, V, W) and a rotation (α, β). Thus, from (3), if we divide u_n by n_x (if $n_x \neq 0$), we obtain a function $f_n(\mathbf{x}, \mathbf{n}) = \frac{u_n}{n_x}$ of the image coordinates $\mathbf{x} = (x, y)$ and the normal direction $\mathbf{n} = (n_x, n_y)$:

$$\begin{aligned} f_n(\mathbf{x}, \mathbf{n}) &= \frac{u_n}{n_x} \\ &= \frac{(xW - U)}{Z} + \frac{(yW - V)}{Z}\frac{n_y}{n_x} \end{aligned}$$

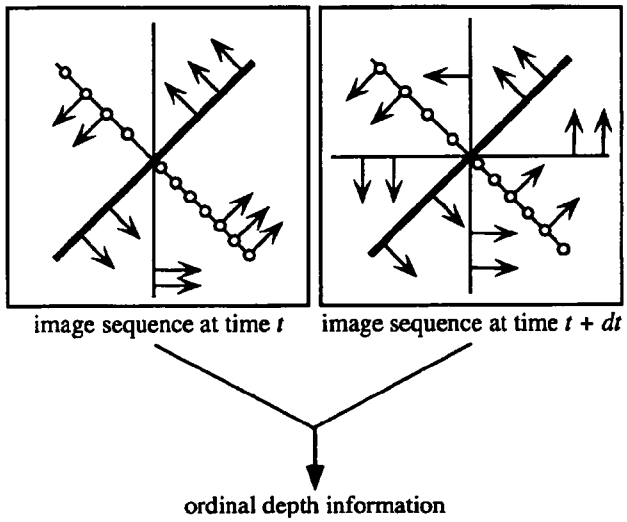


Fig. 12. By comparing measurements of directional flow—wherever available—along lines of equal slope passing through the center, we derive ordinal depth information adequate to accomplish the perimeter following task.

$$+ \alpha\left(\frac{xy}{f}\left(\frac{y^2}{f} + f\right)\frac{n_y}{n_x}\right) - \beta\left(\left(\frac{x^2}{f} + f\right) + \frac{xy}{f}\frac{n_y}{n_x}\right). \quad (13)$$

Let us choose $\mathbf{n} = (n_x, n_y)$ in the direction of the $(0,0)$ copoint vectors, that is, perpendicular to lines through the image center. For these directions, we obtain

$$f_n(\mathbf{x}, \mathbf{n}) = \frac{u_n}{n_x} = \frac{(-U - V\frac{n_y}{n_x})}{Z} + \alpha f\frac{n_y}{n_x} - \beta f. \quad (14)$$

Along each of these lines, $\frac{n_y}{n_x}$ is constant, and thus, $(-U - V\frac{n_y}{n_x})$ and $(\alpha f\frac{n_y}{n_x} - \beta f)$ are also constant, and (14) describes $f_n(\mathbf{x}, \mathbf{n})$ as a function that is linear in the inverse depth. For any two points P_1 and P_2 with coordinates $\mathbf{x}_1$ and $\mathbf{x}_2$ along such a line, the difference $(f_n(\mathbf{x}_1, \mathbf{n}) - f_n(\mathbf{x}_2, \mathbf{n}))$ is independent of the rotation. We thus compute the directional derivative $D(f_n(\mathbf{x}, \mathbf{n}))_{\mathbf{n}^\perp}$ of $f_n(\mathbf{x}, \mathbf{n})$ at points on lines with slope $k = -\frac{n_x}{n_y}$ in the direction of a unit vector $\mathbf{n}^\perp = (-n_y, n_x)$ parallel to the image lines. Dropping, in the notation of the directional derivative, the dependence of f_n on $\mathbf{x}$ and $\mathbf{n}$, we obtain

$$D(f_n)_{\mathbf{n}^\perp} = (-U - V\frac{n_y}{n_x})D(\frac{1}{Z})_{\mathbf{n}^\perp}. \quad (15)$$

We next derive $D(\frac{1}{Z})_{\mathbf{n}^\perp}$. Referring to Figure 13, the camera is mounted on the robot and the robot is moving forward with velocity S. The camera is directed toward a point F on the perimeter. We fix a coordinate system XYZ to the robot such that the Z-axis is aligned with the optical axis of the camera and the XY-plane is perpendicular to it. Let m

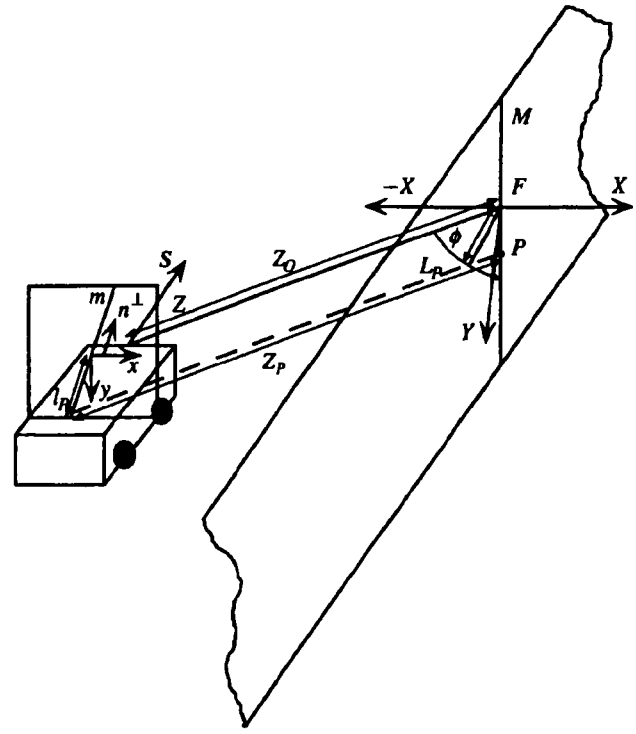


Fig. 13. Geometric configuration during perimeter following.

be the line in the image parallel to $\mathbf{n}^\perp$ along which we compute depth, and let M be the corresponding line in 3D on the perimeter. Let Z_0 be the depth at the fixation point, and let Φ be the angle between the Z-axis and M. The depth Z_P of any point P on M is

$$Z_p = Z_0 + \frac{L_p}{\tan\Phi},$$

where L_p is the value of the parallel projection of $\bar{FP}$ on the XY-plane. L_P projects perspectively onto l_P (along $\mathbf{n}^\perp$) in the image plane, where $l_P = \frac{L_P f}{Z_P}$; thus, dropping subscript P, for any depth value Z, we obtain

$$Z = \frac{Z_0}{1 - \frac{l}{f\tan\Phi}}.$$

For points with coordinates $l\mathbf{n}^\perp$ and gradient direction $\mathbf{n}$, we obtain

$$f_n(l\mathbf{n}^\perp, \mathbf{n}) = (-U - V\frac{n_y}{n_x})\frac{Z_0}{1 - \frac{l}{f\tan\Phi}} + \alpha(f\frac{n_y}{n_x} + \frac{Ky}{f}) - \beta(f + \frac{Kx}{f}),$$

and (15) for any point along the line m becomes

$$D(f_n)_{\mathbf{n}^\perp} = (U + V\frac{n_y}{n_x})\frac{1}{\tan\Phi Z_0 f}. \quad (16)$$

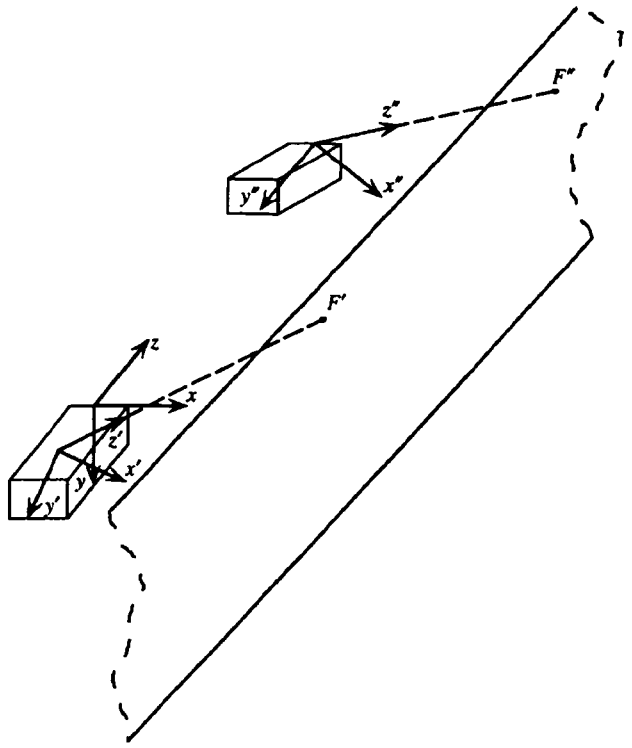


Fig. 14. Three reference coordinate systems.

In the remainder of this section, we demonstrate the dependence of $D(f_n)_{\mathbf{n}^\perp}$ on the robot's steering direction. In particular, we show that $|D(f_n)_{\mathbf{n}^\perp}|$ along certain directions $\mathbf{n}^\perp$ decreases as the robot steers toward the perimeter or that for any two flow fields corresponding to motion configurations C_1 and C_2 depending on parameters (Φ_1, Z_{0_1}) and (Φ_2, Z_{0_2}), $|D(f_n)_{\mathbf{n}^\perp}|_1 > |(D(f_n)_{\mathbf{n}^\perp}|_2$ if $|\Phi_1| < |\Phi_2|$. Here, using the absolute value allows us to provide a general notation that describes fixations of the robot to its left and to its right.

3.3. *Comparing Ordinal Depth Information*

We describe vectors with respect to three orthogonal coordinate systems XYZ, $X'Y'Z'$ and $X''Y''Z''$ that are being rotated into each other (see Fig. 14). The orientations of these coordinate systems are such that the Z-axis is parallel to the direction of translation when the robot is moving parallel to the perimeter, the Z'-axis is parallel to the robot's viewing direction in configuration C_1, and the Z''-axis is parallel to the robot's viewing direction in configuration C_2. The orientations of the frame $X'Y'Z'$ and the frame $X''Y''Z''$ are related to the orientation of the frame XYZ through rotation matrices R' and R'', as described below, which are dependent on parameters ϕ'_x,ϕ'_y and ϕ'_x,ϕ''_y (rotation around the x-axis is the same), where $|\phi''_y| > |\phi'_y|$:

$$R' = \begin{pmatrix} r'_{11} & r'_{12} & r'_{13} \\ r'_{21} & r'_{22} & r'_{23} \\ r'_{31} & r'_{32} & r'_{33} \end{pmatrix}$$
$$= \begin{pmatrix} \cos\phi'_y & 0 & \sin\phi'_y \\ -\sin\phi'_y \sin\phi'_x & \cos\phi'_x & \cos\phi'_y \sin\phi'_x \\ -\sin\phi'_y \cos\phi'_x & -\sin\phi'_x & \cos\phi'_y \cos\phi'_x \end{pmatrix}$$
$$R'' = \begin{pmatrix} r''_{11} & r''_{12} & r''_{13} \\ r''_{21} & r''_{22} & r''_{23} \\ r''_{31} & r''_{32} & r''_{33} \end{pmatrix}$$
$$= \begin{pmatrix} \cos\phi''_y & 0 & \sin\phi''_y \\ -\sin\phi''_y \sin\phi'_x & \cos\phi'_x & \cos\phi''_y \sin\phi'_x \\ -\sin\phi''_y \cos\phi'_x & -\sin\phi'_x & \cos\phi''_y \cos\phi'_x \end{pmatrix}.$$

Any vector V in XYZ corresponds to V' in $X'Y'Z'$ and V'' in $X''Y''Z''$, where

$$V' = R'V \quad V'' = R''V \quad \text{and} \quad V = R'^T V' \quad V = R''^T V'',$$

with R'^T being the transpose of R' and R''^T being the transpose of R''.

Let (in C_1) m' be the line in the image with slope $k = -\frac{n_x}{n_y}$ along which measurements are taken, which is described by the following line equation:

$$m' : y' = kx'.$$

To obtain a vector in 3D on the corresponding line M' on the perimeter, we intersect the plane $Y' = kX'$ with the plane of the perimeter, which is at a distance d from the robot, and thus is described through equation $X = d$ in the coordinate system XYZ or $r'_{11}X' + r'_{21}Y' + r'_{31}Z' = d$ in the coordinate system $X'Y'Z'$. Thus, a unit vector $\mathbf{a}$ along M' in $X'Y'Z'$ is computed as

$$\begin{pmatrix} 1 \\ k \\ \frac{-r'_{11} - kr'_{21}}{r'_{31}} \end{pmatrix} \frac{1}{\sqrt{1 + k^2 + \left(\frac{r'_{11} + kr'_{21}}{r'_{31}}\right)^2}},$$

and a unit vector $\mathbf{b}$ along the Z'-axis in $X'Y'Z'$ is

$$\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}.$$

The cosine of the angle Φ_1 between $\mathbf{a}$ and $\mathbf{b}$ is thus

$$\cos\Phi_1 = \mathbf{a} \cdot \mathbf{b} = -\frac{r'_{11} + kr'_{21}}{r'_{31}\sqrt{1 + k^2 + \left(\frac{r'_{11} + kr'_{21}}{r'_{31}}\right)^2}}$$

and

$$\tan\Phi_1 = -\frac{r'_{31}\sqrt{k^2 + 1}}{r'_{11} + kr'_{21}}.$$

Let d be the distance from the camera to the perimeter. d is measured along the X-axis in XYZ. Any vector V in XYZ that is parallel to the Z'-axis is described as $\lambda(r'_{31}, r'_{32}, r'_{33})^T$, with λ being a scalar. Thus, for a vector V of length Z_{O_1}, $d = \lambda r'_{31}$, and the value of Z_{O_1} amounts to

$$Z_{O_1} = \frac{d}{r'_{31}},$$

and thus

$$\frac{1}{\tan \Phi_1 Z_{O_1}} = -\frac{r'_{11} + kr'_{21}}{d\sqrt{k^2+1}}. \tag{17}$$

Assuming that the time between measurements is small, and thus the horizontal distance d in C_2 is the same as in C_1, we obtain for C_2 (see Fig. 15)

$$\frac{1}{\tan \Phi_2 Z_{O_2}} = -\frac{r''_{11} + kr''_{21}}{d\sqrt{k^2+1}}. \tag{18}$$

Comparing (17) and (18), we are thus interested in values k for which

$$|r'_{11} + kr'_{21}| > |r''_{11} + kr''_{21}| \tag{19}$$

or

$$|\cos\phi'_y - k\sin\phi'_x \sin\phi'_y| > |\cos\phi''_y - k\sin\phi'_x \sin\phi''_y|.$$

This inequality is true for all ($k\sin\phi'_x\sin\phi'_y > 0$), assuming that $|\ \phi'_y\ | < 45°$ and $|\ \phi''_y\ | < 45°$. (This assumption is used only to allow for a general notation and is not needed if different sign cases are listed separately.) On the basis of this result, the input to the control system for deriving the steering direction can be generated. At any point in time, $D(f_n)_{\mathbf{n}^\perp}$ for directions $\mathbf{n}^\perp$ such that ($-\frac{n_x}{n_y}\sin\phi'_x \sin\phi'_y > 0$) are computed from the visual flow field and compared to prestored values $D(f_n)_{\mathbf{n}^\perp}$ due to a forward motion parallel to the perimeter, the computed sign of the change in $|D(f_n)_{\mathbf{n}^\perp}|$ defines the sign of the change in the steering angle.

In the analysis above, the assumption was made that the horizontal distance d does not change. When this assumption does not hold, the same principle can still be applied to a smaller number of values if enough image gradients are available. If the distance d decreased between C_1 and C_2, then $|\ r'_{11} + kr'_{21}\ |$ may be smaller than $|\ r''_{11} + kr''_{21}\ |$ for values of $|\ k\ |$ smaller than some threshold T, but must be larger for values of $|\ k\ |$ greater than T, and thus, if gradients on a line with $|\ k\ | > T$ are available, these measurements can still be used for comparison.

Finally, we want to characterize the lines for which ($k\sin\phi'_x\sin\phi'_y > 0$). Comparing such a line to the image of a line parallel to the road on the perimeter passing through the image center, we find that the slopes of the two lines are of the opposite sign. For example, if the camera's optical axis is pointing down and to the right (as in Fig. 13), then ϕ'_x is negative and ϕ'_y is positive and the slope of the parallel line is positive, whereas the slope of the line we use for comparison has to be negative.

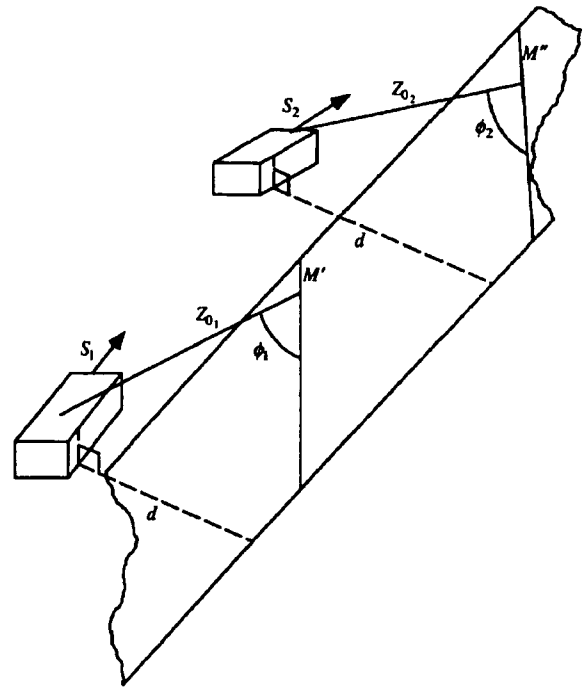


Fig. 15. Comparing the values from two configurations.

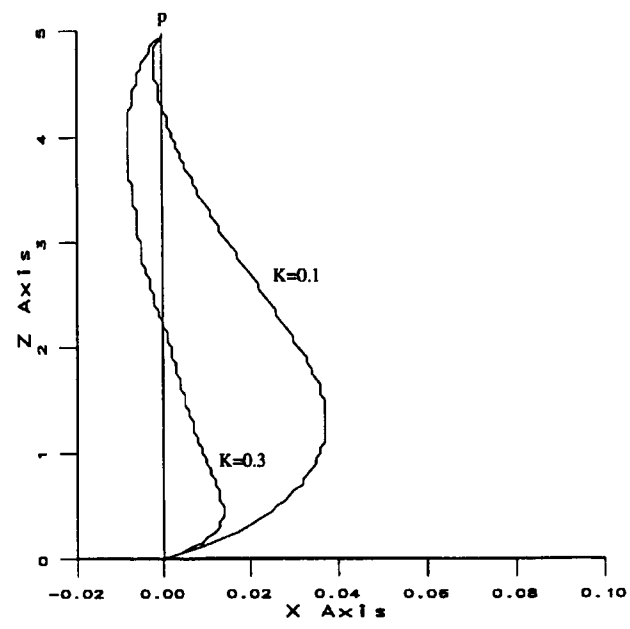


Fig. 16. Trajectory generated by the servo system.

4. Experiments

4.1. Task 1

The first experiments used simulations of the robot's trajectory. The robot is initially moving at an angle relative to the z-axis. It is then required to steer itself so that it is heading along the z-axis, where the feature **p** is located at a distance

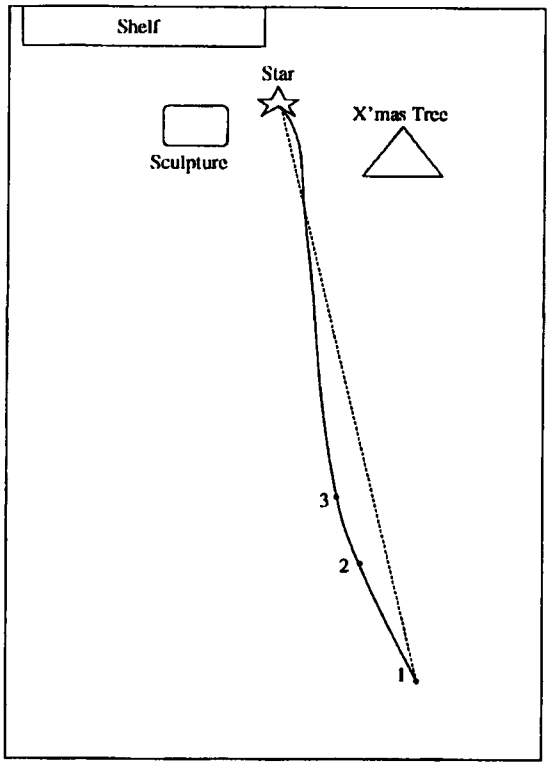


Fig. 17. 3D configuration as studied in Task 1.

Fig. 18. Task 1: Some scenes along the trajectory.

of 5 m away. The robot is moving with a constant forward speed of 1.5 m/s.

The mobile platform is a conventionally steered vehicle; the instantaneous radius of curvature r of its trajectory is related to the steering angle of its wheel θ_R as follows:

$$r = \frac{0.5L}{\sin \theta_R},$$

where L is the body length of the vehicle. The field of view of the camera used in the simulation is 50°. We created the synthetic normal flow field from a scene with random depth, and we added zero-mean Gaussian noise with a standard deviation of 1 pixel to the normal flow measurements. For the estimation of x_0, we employed the α-vectors, where (as described in Section 2.3) we approximated the α-hyperbola by a straight line, which was estimated using a linear classifier.

Figure 16 shows the trajectories for two different values of the proportionality constant K generated by the servo system. The two curves correspond to values of $K = 0.1$ and $K = 0.3$. As can be seen, the system has a poor rise time and insufficient damping—typical of a proportional controller.

In our experiments with real images, the mobile platform on which the camera was mounted was a conventionally steered vehicle. The camera had a focal length of 1136 pixels, and the image dimensions were at 720×576; thus, the field of view was approximately 30°. Considering that the steering movements must be small so as to reduce interframe disparity and allowing for the computation of image flow, especially since the focal length of the camera was very large, we chose to operate the servo system with a proportionality constant of $K = 0.1$. Figure 18 shows some images taken by a camera mounted on a mobile platform as the latter is making steering movements. The feature **p** corresponds to the star in the center of the image, mounted on a tripod and initially located at a distance of 5 m from the camera. Tracking was accomplished by implementing the algorithm of Pahlavan (1993), with occasional fast saccadic movements compensating for drifting; during these saccadic movements, body control was disabled. Figure 17 displays the configuration of this setting including the robot's trajectory. Figures 18a, 18c, and 18e show images taken by the system at three time instants (as marked in Fig. 17) with the normal flow fields superimposed. Figures 18b, 18d, and 18f show the positive and negative α-vectors as computed from the normal flow fields in black and white and the line approximating the α-hyperbola which has been fitted, at frame rate, to the data. The normal flow as shown in the figures was derived only if the spatial gradient was above some threshold, and the sign of the α-vector at a point was taken to be the sign of the normal flow there if the spatial gradient was within 5° of the orientation of the α-vector.

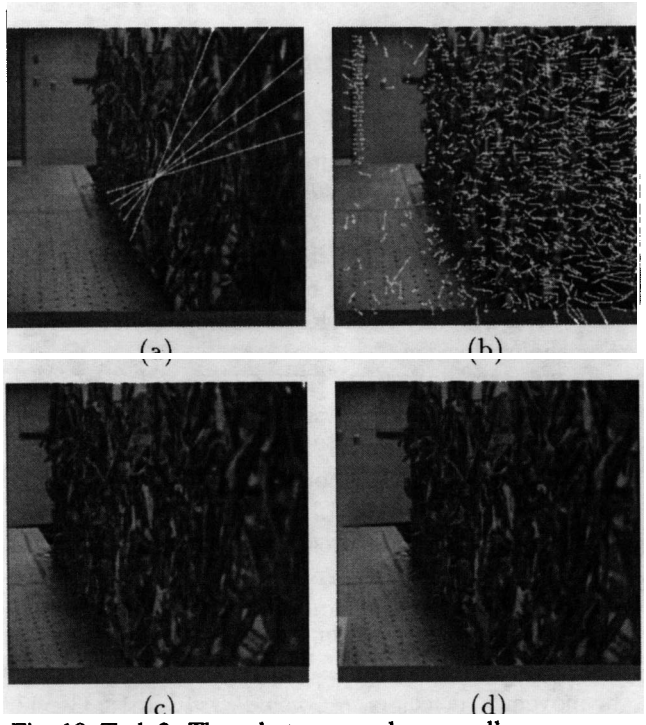


Fig. 19. Task 2: The robot moves along an alley.

4.2. *Task 2*

In the experiments in Task 2, a mobile platform with a camera mounted on it moved along an alleylike perimeter. The camera had a focal length of about 1000 pixels, and the image dimensions were 512×512. The servomechanism was implemented as a simple proportional control relating the robot's rotational speed around the y-axis to the directional derivative of f_n (as defined in Section 3.2). The scene contained a highly textured perimeter. We thus derived image measurements along a number of lines and used the mean of the computed estimates as input to the servo system. Figure 19a shows one of the reference images, which was taken when the robot was moving parallel to the perimeter. The lines along which image measurements were taken are overlaid on the image in white. Figure 19b shows the normal flow field computed for this same image. The success of the technique was tested by studying whether and how the system corrected its path when we moved it either closer to or farther away from the perimeter. In the experiment shown, the system always recovered to a movement parallel to the perimeter. Figure 19c displays an image when the robot steered toward the perimeter, and Figure 19d shows the image when it moved away from the perimeter again.

5. Conclusion

A new way of making use of visual information for autonomous behavior has been presented. Visual representations that are manifested through geometrical constraints defined on the flow in various directions and on the normal flow were used as input to the servomechanism. Specifically, the constraints described are global patterns in the sign of the flow in different directions whose position and forms are related to 3D motion and patterns of normal flow along lines in the image that encode ordinal 3D distance information. 3D motion and structure representations derived from these constraints were applied to the solution of navigational problems involving the control of a system's 3D motion with respect to its environment. Some of the constraints described are of a general nature, however, and thus might be used in various modified forms for the solution of other navigational problems.

Acknowledgments

The support of the Office of Naval Research under Contract N00014-93-1-0257, National Science Foundation under Grant IRI-90-57934, and the Austrian Fonds zur Förderung der wissenschaftlichen Forschung Project No. S7003, and a postgraduate scholarship from Tan Kah Khee Foundation, are gratefully acknowledged. We thank Kourosh Pahlavan for assistance with the experiments.

References

Aloimonos, J., Weiss, I., and Bandopadhay, A. 1988. Active vision. *Int. J. Computer Vision* 2:333–356.

Aloimonos, J. Y. 1990. Purposive and qualitative active vision. *Proc. DARPA Image Understanding Workshop*, pp. 816–828.

Arkin, R., Murphy, R., Pearson, M., and Vaughn, D. 1989. Mobile robot docking operations in a manufacturing environment: Progress in visual perceptual strategies. *Proc. IEEE Int. Workshop on Intelligent Robots and Systems*, pp. 147–154.

Bajcsy, R. 1988. Active perception. *Proc. IEEE* 76:996–1005.

Ballard, D., and Brown, C. 1992. Principles of animate vision. *CVGIP: Image Understanding: Special Issue on Purposive, Qualitative, Active Vision* (edited by Y. Aloimonos) 56:3–21.

Barth, M., and Tsuji, S. 1993. Egomotion determination through an intelligent gaze control strategy. *IEEE Trans. on Systems, Man and Cybernetics* 23(5):1424–1432.

Beardsley, P., Reid, I. D., Zisserman, A., and Murray, D. W. 1995. Active visual navigation using non-metric structure. In *Proc. 5th Int. Conf. on Computer Vision*. Boston: IEEE Computer Society Press, pp. 58–65.

Bradshaw, K., McLauchlan, P. F., Reid, I. D., and Murray, D. W. 1994. Saccade and pursuit on an active head/eye platform. *Image and Vision Computing* 12(3):155–163.

Brady, M. 1985. Artificial intelligence and robotics. *Artificial Intelligence* 26:79–121.

Espiau, B., Chaumette, F., and Rives, P. 1992. A new approach to visual servoing in robotics. *IEEE Trans. on Robotics and Automation* 8:313–326.

Faugeras, O. D. 1995. Stratification of 3-D vision: Projective, affine, and metric representations. *J. Opt. Soc. Am. A* 12(3):465–484.

Fermüller, C. 1995. Passive navigation as a pattern recognition problem. *Int. J. Computer Vision* 14:147–158.

Fermüller, C., and Aloimonos, Y. 1997. On the geometry of visual correspondence. *Int. J. Computer Vision* 21:223–247.

Fermüller, C., and Aloimonos, Y. 1995a. Direct perception of three-dimensional motion from patterns of visual motion. *Science* 270:1973–1976.

Fermüller, C., and Aloimonos, Y. 1995b. Global rigidity constraints in image displacement fields. *Proc. Fifth Int. Conf. on Computer Vision*, pp. 245–250.

Fermüller, C., and Aloimonos, Y. 1995c. Qualitative egomotion. *Int. J. Computer Vision* 15:7–29.

Fermüller, C., and Aloimonos, Y. 1995d. Representations for active vision. *Proc. Int. Joint Conf. on Artificial Intelligence*.

Fermüller, C., and Aloimonos, Y. 1995e. Vision and action. *Image and Vision Computing* 13:725–744.

Fermüller, C., and Aloimonos, Y. 1996. Ordinal representations of visual space. *Proc. ARPA Image Understanding Workshop*, pp. 897–903.

Hager, G., Chang, W.-C., and Morse, A. 1994 (San Diego, CA). Robot feedback control based on stereo vision: Towards calibration-free hand-eye coordination. *Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 2850–2856.

Hollinghurst, N., and Cipolla, R. 1993. Uncalibrated stereo hand/eye coordination. In J. Illingworth (ed.): *Proc. 4th British Machine Vision Conf.* Guildford: BMVA Press, pp. 389–398.

Koenderink, J., and van Doorn, A. 1995. Relief: Pictorial and otherwise. *Image and Vision Computing* 13:321–334.

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

Murray, D., Bradshaw, K. J., McLauchlin, P. M., Reid, I. D., and Sharkey, P. M. 1995. Driving saccade to pursuit using image motion. *Int. J. Computer Vision* 16(3):205–338.

Murray, D., McLauchlin, P. F., Reid, I. D., and Sharkey, P. M. 1993. Reactions to peripheral image motion using a head/eye platform. *Proc. 4th Int. Conf. on Computer Vision, Berlin*. Los Alamitos, CA: IEEE Computer Society Press, pp. 403–411.

Pahlavan, K. 1993. Active robot vision and primary ocular processes. Ph.D. thesis, Royal Institute of Technology, Stockholm, Sweden.

Poggio, T., and Reichardt, W. 1973. Considerations on models of movement detection. *Kybernetik* 13:223–227.

Raviv, D., and Herman, M. 1993. Visual servoing from 2-D image cues. In Y. Aloimonos (ed.): *Active Perception*. Hillsdale, NJ: Lawrence Erlbaum, pp. 191–226.

Reichardt, W. 1961. Autocorrelation, a principle for evaluation of sensory information by the central nervous system. In W. Rosenblith (ed.): *Principles of Sensory Communication*. New York: John Wiley & Sons, pp. 303–317.

Reichardt, W. 1987. Evaluation of optical motion information by movement detectors. *J. Comp. Physiol.* 161:533–547.

Reid, I., and Murray, D. 1993. Tracking foveated corner clusters using affine structure. *Proc. 4th Int. Conf. on Computer Vision, Berlin*. Los Alamitos, CA: IEEE Computer Society Press, pp. 76–83.

Santos-Victor, J., Sandini, G., Curotto, F., and Garibaldi, S. 1993. Divergent stereo for robot navigation: Learning from bees. *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 434–439.

Singh, A. 1990. Optic flow computation: A unified perspective. Ph.D. thesis, Department of Computer Science, Columbia University.

Skaar, S., Brockman, W., and Hanson, R. 1987. Camera-space manipulation. *Int. J. Robotics Research* 6:20–32.

van Santen, J., and Sperling, G. 1984. Temporal covariance model of human motion perception. *J. Opt. Soc. Am. A* 1:451–473.

Verri, A., and Poggio, T. 1987. Against quantitative optic flow. *Proc. IEEE Int. Conf. on Computer Vision*.

Weiss, L., and Sanderson, A. 1987. Dynamic sensor-based control of robots with visual feedback. *IEEE Trans. on Robotics and Automation* 3:404–417.