Visual servoing using dynamic image parameters

François Chaumette, Venkataraman Sundareswaran, Patrick Bouthemy

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Visual servoing using dynamic image parameters

Venkataraman Sundareswaran,

Patrick Bouthemy, François Chaumette

N° 2336

Août 1994

PROGRAMME 4

ISSN 0249-6399

INRIA
RENNES

Visual servoing using dynamic image parameters

Venkataraman Sundareswaran*,
Patrick Bouthemy**, François Chaumette***

Programme 4 — Robotique, image et vision
Projet Temis

Rapport de recherche n° 2336 — Août 1994 — 32 pages

Abstract: Visual servoing is a framework for achieving the tight coupling of camera movements and information from images. We consider a typical visual servoing approach that uses geometric information about image features for controlling the position and attitude of a camera. We extend the applicability of this approach by using image motion information. To this end, we present two different approaches to visual tasks that use motion information. The first one uses the focus of expansion. The second one incorporates the parameters of the 2D affine motion model in control equations. We illustrate both these approaches by means of a task to align the optical axis of the camera with the unknown direction of translational motion of the system on which it is mounted. We present results of simulation experiments, and real experiments done with a six DOF robot with a camera on its end-effector. The contributions of this work are in particular extending the visual servoing formalism beyond using just image features, and in general showing that a tight coupling between camera behavior and image motion is possible.

Key-words: Active vision, visual servoing, focus of expansion, 2D affine motion parameters

(Résumé : tsvp)

*e-mail sundar@bu.edu
**e-mail bouthemy@irisa.fr
***e-mail chaumett@irisa.fr

Unité de recherche INRIA Rennes
IRISA, Campus universitaire de Beaulieu, 35042 RENNES Cedex (France)
Téléphone : (33) 99 84 71 00 – Télécopie : (33) 99 84 71 71

Utilisation d'informations visuelles dynamiques en asservissement visuel

Résumé : Les techniques d'asservissement visuel sont basées sur l'utilisation d'information visuelles de type géométrique afin de contrôler en boucle fermée la position et le mouvement d'une caméra mobile. Dans ce rapport, nous généralisons cette approche à l'utilisation d'informations visuelles de type dynamique, c'est-à-dire basées sur le mouvement mesuré dans la séquence d'images acquises par la caméra. La tâche choisie pour valider cette approche consiste à aligner l'axe optique de la caméra dans la direction inconnue de son déplacement. Deux types d'informations visuelles sont modélisés pour réaliser cette tâche : d'une part, les coordonnées du foyer d'expansion, et, d'autre part, les paramètres affines du mouvement 2D. Après l'élaboration des lois de commande associées, nous présentons des résultats de simulation et des résultats expérimentaux obtenus à l'aide d'une caméra embarquée sur l'effecteur d'un robot à six degrés de liberté. Finalement, nous montrons que cette tâche d'alignement permet de faciliter l'interprétation du mouvement des objets dans la scène, confirmant ainsi l'intérêt de coupler la vision active et la vision qualitative.

Mots-clé : Vision active, asservissement visuel, foyer d'expansion, paramètres affines du mouvement 2D

1 Introduction

In recent years, many new approaches have been taken to perform tasks based on responses to visual information. They are motivated by a need to *react* to visual stimuli in appropriate ways, resulting in a coupling between behavior and perception. These approaches clearly fall in the realm of active vision because of the nature of the generic mechanism involved: visual inputs are processed, and based on this information, camera behavior is controlled, usually with the goal of attaining a configuration that simplifies further action [4, 5, 18]. The approaches span a broad spectrum ranging from methods that perform very specific tasks, to methods that provide general frameworks. An extensive survey of all the methods is outside the scope of this paper. We mention just a few in each of the broad categories.

There are many examples for approaches that perform specific tasks. Nelson and Aloimonos proposed a scheme to detect obstacles by using flow field divergence [29]; Santos-Victor et. al. [23], and Coombs and Roberts [14] presented methods to steer a camera between two walls, and to veer around obstacles, both methods being based on a simple analysis of the computed optic flow fields. Performing saccades in real time to moving regions of interest has been demonstrated in [27]. Krotkov [25] presented a method to perform focusing, based on a special search technique. Yuille and Geiger [39] proposed a method to control movements of stereo cameras to help solve the correspondence problem. Coombs and Brown [13] outlined a method to control camera movements to keep stereo cameras locked on the target.

In the middle of the spectrum, there are methods that perform more general tasks such as gaze control and fixation; these are more general because fixation or gaze control can be used as a means to accomplish other tasks. In [6], the uses of gaze control, and fixation in particular, in obtaining an object-centered reference-frame and in figure-ground discrimination have been outlined. A system performing several active visual tasks, including closed-loop gaze-control (based on a differential analysis) for fixating on an object has been presented in [19]. A fixation method running in real-time on a head-eye system can be found in [32]. Olson and Coombs presented a real-time vergence control for stereo cameras by estimating vergence error with a cepstral disparity filter [31]. Target-tracking and grasping methods by cameras mounted on robots have been demonstrated [1].

On the farther side of the spectrum, there are methods which provide more general solutions. Visual servoing methods [16, 17, 21, 33, 38] present control-theoretic approaches for controlling the position and attitude of the camera by using visual information. These methods are general, and can be used to perform a variety of tasks with specific goals. For instance, the general formalism in [16] has been used in a wide range of tasks such as absolute positioning [16], target tracking [12], and structure estimation [11]. For instance, in [11], visual servoing is used to choose the camera motion that leads to a robust and accurate estimation of 3D structure.

Our present work is in extending the power of this class of approaches, by showing how motion information can be incorporated. We start from one of the visual servoing formalisms [16]. The procedure outlined in [16] for determining control equations is suited for the use of geometric information about features such as points, lines and circles. We show how this formalism can make use of motion-based information. In particular, the methods described in this paper use dynamic image parameters such as focus of expansion or coefficients in the 2D affine motion

model. Such a use of dynamic image parameters for the purpose of visual servoing is new, and it provides a starting point for other interesting closed-loop methods using these parameters.

Certain remarks are in order here. Using motion information is of relevance only when the camera is undergoing certain unknown motion, and/or when mobile objects are in the field of view. This is the case when the camera is acting as a sensor for the host vehicle whose primary mission is not mere visual exploration. This is not an unusual scenario. Consider Fig. 1 which depicts navigational situations where a vehicle carrying a camera moves about to perform tasks (such as driving on a highway, or performing a landing sequence), and the camera acts as a sensory device for avoiding obstacles, locating targets etc. Contrast this to a situation where the camera is *otherwise* stationary except when performing a vision-based task (eg., positioning). The important thing to note is that in the former case, *usually* there are less degrees of freedom available for control than in the latter case (however, when the motion is only external, as with moving objects in the scene, all the degrees of freedom may be available for control). In the situations in Fig. 1, only the rotational motion of the camera is controllable.

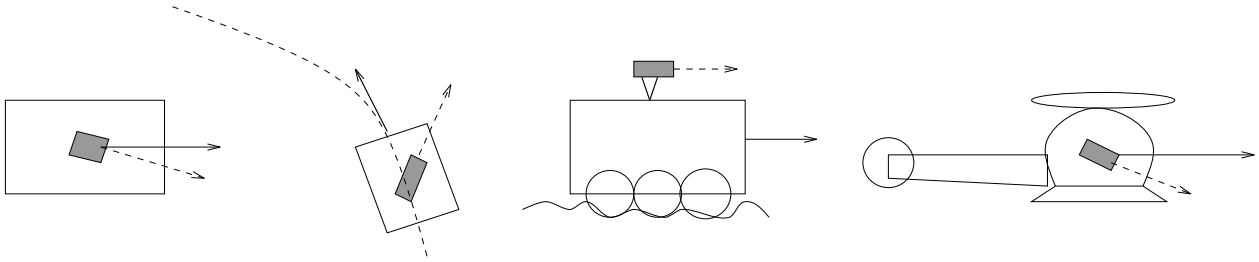


Figure 1: Navigational situations

In this paper, we show that the focus of expansion can be treated just like geometric information. Thus, a straightforward application of the visual servoing principle is possible. We illustrate this by considering the task to align the optical axis of the camera with the direction of translation–henceforth referred to as the *alignment task*. We also show how the visual servoing can be used in completely new ways, by incorporating the parameters of the 2D affine motion model in the control equations. We demonstrate the approach by presenting control equations for the alignment task. Barth and Tsuji [7] have proposed a scheme for the alignment task, based on fixation. We will discuss their scheme after presenting ours.

Towards the end of this paper, we suggest an application for the kind of control we are proposing. A coupling of qualitative vision methods and active vision methods is an interesting direction that needs to be explored. We outline this idea by considering a qualitative method that can detect moving objects seen by a camera that itself is under motion [9]. The qualitative method under consideration works well if the motion of the camera is, for example, along its optical axis. To improve this solution, the main idea is not to find a general solution which extends the previous one but to use system capabilities to circumvent the restrictions in order to be able to still use the initial method which is simple enough and known to be robust and efficient. A promising approach consists in designing active schemes in order to dynamically place the camera in some known tractable configuration and then to use simple interpretation methods. This is precisely a goal of the active visual task described here. The proposed coupling of a qualitative method and an active visual method is validated by a real example in which

a mobile camera detects a moving object. In part, it is also a demonstration to show that the servoing methods described here work not only in the case of a static environment, but also when there are small moving objects in the scene.

In summary, what we present is an approach to the tight coupling of behavior and motion perception. We note that our innovative use of the dynamic image parameters in control is not limited to the task described here. It has general applicability; it is possible to use such dynamic parameters for various active vision tasks to obtain fast and robust control methods. Our intention here is to provide a starting point for such approaches. Parts of this work have appeared in [10, 35, 36, 37].

2 Visual Servoing

In this section, we review the basic principles of visual servoing. Detailed descriptions can be found in [16, 17, 21, 38].

The principle of visual servoing is to use visual information as observation in closed-loop control when the desired configuration can be described as a particular visual observation. The control is effected on the camera position and orientation or on an external object such as a robot arm. The only condition is that the instantaneous change in the visual information (in the sense of temporal derivative) as a function of the controllable parameters be known analytically. Intuitively, if the effect of the control parameters on the observation is known, we could provide the appropriate control in order to result in obtaining the desired observation. The visual servoing theory provides a framework for determining the *control law* which is simply a set of equations to calculate the control parameters.

More precisely, for a given vision-based task, we have to choose a set $\mathbf{s}$ of visual features suited for achieving the task (for example, the coordinates of an image point, the parameters of a selected line, etc). In order to perform a control law based on $\mathbf{s}$, we need to know the equations for the variation of $\mathbf{s}$ with respect to camera translational and rotational motion (T, Ω). In other words, we have to determine the matrix L described by the following equation:

$$\dot{\mathbf{s}} = L \begin{pmatrix} T \\ \Omega \end{pmatrix} \tag{1}$$

In the terminology of [16], L is called the *interaction matrix* related to $\mathbf{s}$. For example, if $\mathbf{s}$ is the location (x_p, y_p) of the object in the image (say, its center of gravity), this matrix can be obtained from the well-known equations of optical flow [22] for the considered point:

$$\begin{array}{rcl} \dot{x_p} & = & \frac{1}{Z(x_p,y_p)}\left[-U + x_p W\right] + A x_p y_p - B\left[1 + x_p^2\right] + C y_p, \\ \dot{y_p} & = & \frac{1}{Z(x_p,y_p)}\left[-V + y_p W\right] + A\left[1 + y_p^2\right] - B x_p y_p - C x_p. \end{array} \tag{2}$$

We assume that we have planar images obtained by the pin-hole perspective approximation, with a focal length of unity. We utilize the usual coordinate systems seen in Fig. 2. The translational velocity T has components U, V, and W. The components of the rotational velocity Ω are A, B and C.

Results for more complex visual features are given in [17] (for image segments) and in [16] where a general method for computing the interaction matrix of any visual feature defined

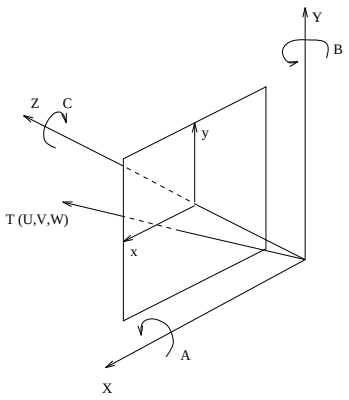


Figure 2: The coordinate systems.

upon geometrical primitives is proposed and where explicit results are given for the parameters describing the projection in the image of lines, circles, spheres and cylinders.

A *task function* $\mathbf{e}$ can be defined as

$$\mathbf{e} = M\ (\mathbf{s} - \mathbf{s}^*), \tag{3}$$

where

- $\mathbf{s}$ is the measured visual features currently observed by the camera,
- $\mathbf{s}^*$ is the desired final configuration for $\mathbf{s}$ in the image, and
- M is a constant matrix which allows, for robustness issues, to take into account more visual features than necessary. Let us note that M can simply be chosen as the identity matrix when the number of the selected visual features is equal to the number of the camera d.o.f. controlled by the task.

The control problem thus appears as the regulation of the task function $\mathbf{e}$ to zero or, equivalently, as the minimization of $\|\mathbf{e}\|$ in the image by appropriate camera motion. We would like the task function to decay exponentially towards zero. For such a requirement,

$$\dot{\mathbf{e}} = -\lambda \mathbf{e},$$

where $\lambda(> 0)$ is the exponent that controls the speed of the decay. Noting that M and $\mathbf{s}^*$ are constant and assuming for simplicity that the scene is static, we obtain:

$$ML \begin{pmatrix} T \\ \Omega \end{pmatrix} = -\lambda \mathbf{e}. \tag{4}$$

Inverting Eqn. 4, we get the control law

$$\begin{pmatrix} T \\ \Omega \end{pmatrix} = -\lambda L^+ M^+ \mathbf{e}, \tag{5}$$

where L^+ and M^+ are the pseudo-inverses of L and M.

Generally, the interaction matrix L cannot be exactly computed. For example, in the case of a point, described by Eqn. 2, L depends on the depth Z which is generally unknown. In such cases, a model of L (say $\hat{L}$) has to be chosen and we finally obtain:

$$\begin{pmatrix} T \\ \Omega \end{pmatrix} = -\lambda \hat{L}^{+} M^{+} \mathbf{e}, \tag{6}$$

An exponential decay of $\mathbf{e}$ will be ensured under the sufficient condition [16]:

$$M L \hat{L}^{+} M^{+} > 0 \tag{7}$$

in the sense that a $n \times n$ matrix A is positive-definite if $x^T A x > 0$ for any nonzero $x \in \mathcal{R}^n$. Usually, $\hat{L}$ is chosen as L^*, the value of L at convergence. Indeed, in that case, the positivity condition is valid around the desired configuration [15].

Thus the principle of visual servoing is to use visual information to perform a closed-loop control to reduce an "error" in the visual information. Various tasks have been performed within the framework of the visual servoing outlined here and described in detail in [16].

We propose to utilize this visual servoing framework, but making use of non-traditional visual features such as the focus of expansion or the affine motion parameters.

3 Image motion information

The traditional visual servoing approach described in the previous section provides a method to determine the camera control in such a way as to have a chosen geometric property (eg., the location of an image feature reach a desired value). For instance, we could bring the camera to position itself at a distance of, say, 30 cm from a square of known size, by requiring that the image projections of the corners of the squares reach predetermined locations on the image [16].

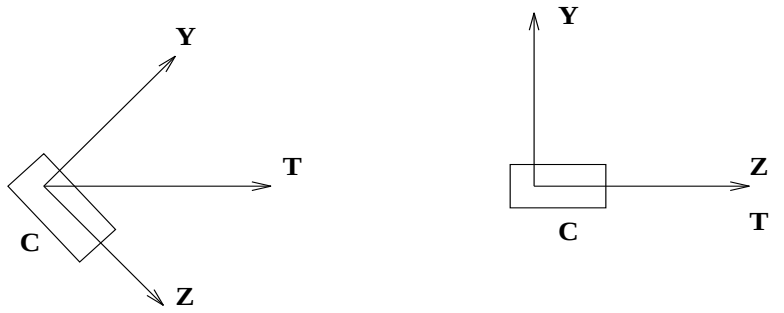
However, the visual servoing approach is more general. Indeed, that is the claim of this paper. We can control the camera to attain the target value for *any* parameter (i.e., not necessarily just a geometric parameter), as long as this parameter can be expressed in terms of the camera motion parameters that can be controlled. We confirm this claim by using parameters obtained from image motion.

First, we show that we can use the focus of expansion. This is only a modest step forward because the focus of expansion (FOE) is a geometric property in some sense; this is so because the FOE is a location on the image plane, and can be treated as such. Nevertheless, this viewpoint brings out the notion of using motion information available as a simple geometric feature.

Secondly, we show how to use the parameters of the affine model (henceforth, *affine motion parameters*) of the optic flow. Here, we truly deviate from the tradition of using geometric information for visual servoing. The affine motion parameters have been well-studied theoretically [24] as well as empirically [28], and have been used in many applications. But they have not been exploited so far as entities useful in directly controlling camera behavior.

We will illustrate the approaches in a task to align the optical axis of the camera with the direction of translation. The initial and final configurations of a camera (C) undergoing alignment is shown in Fig. 3. For simplicity, only the projection on the YZ plane is shown.

The translation vector is T. For the purposes of this paper, we restrict ourselves to the pure translation situation (the first one in Fig. 1), and hope that this will provide an initiative to solve the more general cases.



(a) Before Control (b) After Control

Figure 3: Effect of the alignment control

The alignment task can be accomplished by rotations of the camera with respect to the X and Y axes (i.e., pan and tilt). This is evident if we note that rotations about two axes are sufficient to orient the coordinate system in any desired way. Thus, while the rotation about the Z axis (roll) could also be used, it is not necessary. According to control theory (see [16]), this additional degree of freedom can be used to perfom auxiliary tasks, but we will not explore that possibility here. We will restrict our attention to the computation of the control rotational velocities in the pan and tilt directions.

The various possibilities in Fig. 4 have been investigated. The angular acceleration control has been rejected for reasons explained in Sections 5 and 6.1. The two different possibilities for angular velocity control (i.e., using a_1 and a_4 or using U/Z and V/Z) result in closely related control laws; both have been implemented on our experimental system. In Fig. 4 the box corresponding to the angular acceleration control has been rendered gray to indicate that it has not been implemented in the real system.

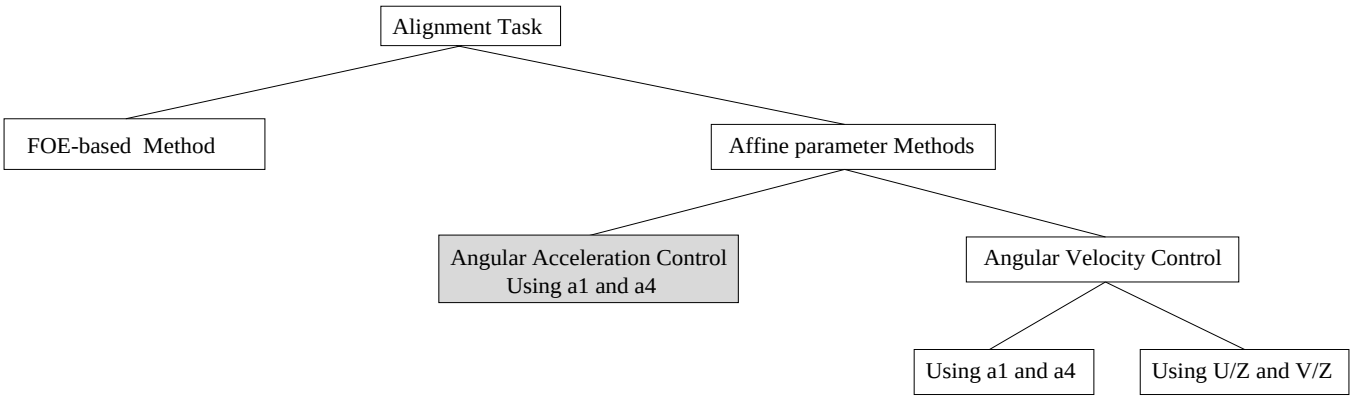


Figure 4: Methods for alignment.

We show the theoretical development first, and then present experimental results.

4 Control using the FOE

The focus of expansion (x_f, y_f) is the projection of the 3D translational vector on the image plane: $x_f = \frac{U}{W}$, $y_f = \frac{V}{W}$. The location (x_f, y_f) may be treated as a geometric feature and used to obtain control laws. For instance, we could apply control to bring this *feature* to the center of the image, thus accomplishing the specified alignment task.

We only control camera pan A and tilt B, and we know, from the optic flow equations 2,

$$\begin{pmatrix} \dot{x_f} \\ \dot{y_f} \end{pmatrix} = L \begin{pmatrix} A \\ B \end{pmatrix}, \quad L = \begin{pmatrix} x_f y_f & -(1+x_f^2) \\ 1+y_f^2 & -x_f y_f \end{pmatrix}.$$

In this case, the matrix L can be estimated on-line since it only depends on the position (x_f, y_f) of the FOE measured in the image. In Eqn. 5, we can set M as the identity matrix and we finally obtain:

$$\begin{pmatrix} A \\ B \end{pmatrix} = -\lambda L^{-1} \begin{pmatrix} x_f - x_f^* \\ y_f - y_f^* \end{pmatrix}, \tag{8}$$

where

$$L^{-1} = \frac{1}{1+x_f^2+y_f^2} \begin{pmatrix} -x_f y_f & 1+x_f^2 \\ -(1+y_f^2) & x_f y_f \end{pmatrix}.$$

For the alignment task, $x_f^* = y_f^* = 0$. One could also position the FOE at any desired location (x_f^*, y_f^*) using exactly the same method.

4.1 Computing the FOE

Here we describe how we can quickly determine an estimate of the location of the FOE, given a translational motion (i.e., no rotational motion). Consider the optical flow equations in Eqn. 2. Since there is no rotation, the equations are simplified:

$$\begin{array}{rcl} u(x,y) & = & \frac{1}{Z(x,y)}[-U + xW], \\ v(x,y) & = & \frac{1}{Z(x,y)}[-V + yW], \end{array} \tag{9}$$

where $Z(x,y)$ is the depth of the point projected at (x,y). It is easy to see that the point of intersection of these image plane vectors gives the Focus of Expansion (FOE). Indeed, at this point $(x = \frac{U}{W}, y = \frac{V}{W})$, $u = v = 0$, and the flow vectors *spread* out in a radial direction from this point (see Fig. 5-left). The vector field (u,v), termed the optical flow field, has to be computed in order to determine the FOE as the intersection of the vectors. Due to the lack of reliable methods to compute the optic flow, it is desirable to estimate the FOE from the *normal flow field* vectors; the normal flow vector is given by [22]

$$\vec{V_n} = \frac{-I_t}{\| \vec{\nabla} I \|^2} \vec{\nabla} I,$$

where I is the spatio-temporal image intensity function, and I_t and $\vec{\nabla} I$ are its temporal and spatial derivatives. This is obtained from the image motion constraint equation

$$\frac{dI}{dt} = 0,$$

or, equivalently

$$I_x u + I_y v + I_t = 0,$$

from which only the projection of (u, v) in the local intensity gradient (I_x, I_y) direction (in other words, on the normal to the local *edge*–hence the name *normal flow*) can be computed, and the magnitude of this projection is determined by the partial temporal derivative I_t. This constraint is a manifestation of the *aperture problem* that complicates the computation of the exact optical flow field, and in most cases, the optical flow field can only be approximately computed based upon certain assumptions about the regularity of the field.

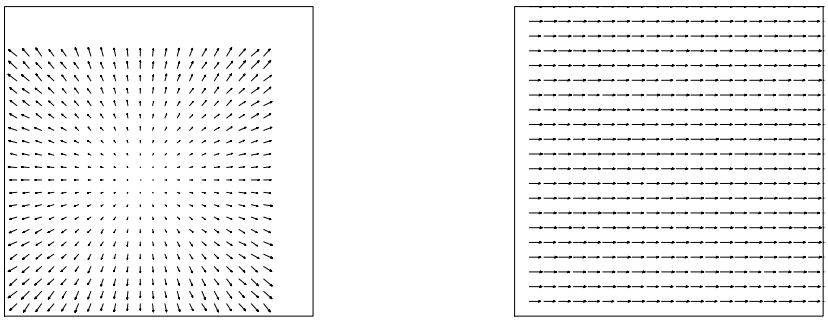


Figure 5: Optical flow for forward and sideways translation

Hence it is interesting to see whether in natural images the normal flow field provides sufficient information to calculate the FOE. This is because the normal flow can be computed relatively easily and more robustly than the optical flow field. Aloimonos and Duric propose a method to compute the FOE location from normal flow components [3]. Our requirements are for a faster method since we need to use this estimation in a control loop to ensure the reliability of the final results. The advantage of performing the task in a closed-loop fashion is that we need not have a precise estimate of the FOE; we need only an approximate value that will reduce an appropriately defined error. Indeed, ideally, the control can cope up with a method that is approximate when the FOE is *outside* the image, but is reasonably accurate when the FOE is close to the center of the image. The method used here meets this requirement, and is fast enough to be implemented on a real robot-camera combination.

Consider the direction of the optical flow vectors. Intuitively, for example, if all the optical flow components are pointing "leftward," then the motion is "rightward," so the rotation should move the camera to look to the right. A similar intuition applies for the vertical motion. We can use the normal flow instead of the optical flow. In essence, what we show in Appendix. 1 is that the direction of the components of the optical flow coincide with the direction of the components of the normal flow, with high probability. Since the control proposed here uses only the direction of the components, such a consistency might be sufficient, and it has been verified experimentally. Recent experimental evidence [34] confirms that this FOE computation method also yields good results on standard image sequences.

We consider the optical flow induced due to the translation of the camera. When the camera is translating forward (i.e., with the FOE at the center of the image), the flow field obtained is *radial*, similar to the one shown in Fig. 5 (left). On the other hand, if the motion is *sideways*, the flow field obtained is a collection of nearly parallel vectors (see Fig. 5-right). The goal is to rotate the camera in such a way as to result in a radial flow field; one could potentially begin with a parallel flow field (i.e., with the camera looking sideways or upward).

The clue about the x coordinate of the FOE is given by the distribution of the horizontal components of the flow vectors. Consider the two half-regions of the image, indicated by L and R in Fig. 6. If the FOE is at the center (that is, on the line dividing the two regions L and R), then the horizontal components of the flow vectors in L will be oriented towards the left, and the horizontal components of the flow vectors in R will all be oriented towards the right, as for the situation in Fig. 5(left). If this is not the case –that is, for example, if the vectors in L are oriented to the right– then the FOE is not located on the dividing line between the two regions. In particular, if *all* the flow vectors on the image are oriented to the right (Fig. 5-right), then the FOE is on the left, outside the image. In general, the ratio of the number of vectors oriented to the left to the number of vectors oriented to the right yields an estimate of the x coordinate of the FOE. With this approach, if the FOE is outside the image, we can only determine the direction to the FOE. In a similar fashion, we can estimate the y coordinate of the FOE, using the vertical components of the flow vectors.

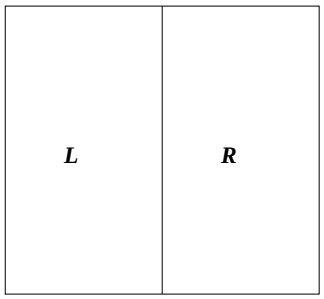


Figure 6: Two halves of the image

The fact that we only estimate the direction to the FOE when the FOE is outside the image does not affect the control greatly; we assume that the FOE is on the border of the image in the computed direction and calculate the control. The method described here permits an implementation that is rapid and is accurate enough for the control scheme when the FOE is not exactly at the center. After convergence, however, if the control is not withdrawn, the approximate computation of the FOE creates a minor oscillation of the center about the FOE. As will be seen in the experiments, it is small enough to be negligible but perfect precision is not guaranteed.

We now summarize the method with more details: the normal flow field is computed, and the fraction h_p of positively oriented (i.e., towards the left) horizontal components and the fraction v_p of positively oriented (upwards) vertical components are determined by counting.

The approximate FOE is computed as

$$(-(h_p - 0.5) * W_I/2, -(v_p - 0.5) * H_I/2),$$

where W_I and H_I are the width and the height of the image in focal length units. This gives a location on the image if the FOE is within the image, or a location on the border of the image if the FOE is on the border or is outside the image; in the last situation where the FOE is outside the image, the computed position is in the direction of the FOE. More details and experimental results of this particular method of computing the FOE are reported in [34].

5 Control using the affine parameters

In this section, we propose to use, under the same servoing formalism, measures obtained from the affine motion parameters. We consider the same (alignment) task as in the previous section, to illustrate the derivation method. We begin by describing the parameters of the affine motion model, and then derive the control laws to achieve the alignment task using these parameters.

5.1 The 2D affine motion parameters

The 2D affine motion model is often useful. It is possible to derive expressions for the first-order parameters (affine parameters) assuming that an analytical surface is imaged (i.e., it is possible to describe the depth by a Taylor series expansion). It has been already shown [28] that the affine parameters can be reliably estimated. Multiresolution methods for the estimation of the affine parameters [8] have proved to yield accurate values.

The optical flow equations are repeated here from Eqn. 2 (see also Fig. 2):

$$\begin{array}{rcl} u(x,y) & = & \frac{1}{Z(x,y)}\left[-U + xW\right] + A\left[xy\right] - B\left[1 + x^2\right] + Cy, \\ v(x,y) & = & \frac{1}{Z(x,y)}\left[-V + yW\right] + A\left[1 + y^2\right] - B\left[xy\right] - Cx. \end{array} \tag{10}$$

Let the first-order approximation be

$$\begin{array}{rcl} u(x,y) & = & a_1 + a_2 x + a_3 y, \\ v(x,y) & = & a_4 + a_5 x + a_6 y. \end{array} \tag{11}$$

Let the first-order model for the imaged surface be

$$Z = Z_0 + \gamma_1 X + \gamma_2 Y.$$

It is very easy to show that

$$\frac{1}{Z} = \frac{1}{Z_0}(1 - \gamma_1 x - \gamma_2 y), \tag{12}$$

From Eqns. (10), (11), and (12), we get [9, 28].

$$\begin{array}{rclrcl} a_1 & = & -\frac{U}{Z_0} - B, & a_4 & = & -\frac{V}{Z_0} + A, \\ a_2 & = & \frac{1}{Z_0}(\gamma_1 U + W), & a_5 & = & \frac{1}{Z_0}(\gamma_1 V) - C, \\ a_3 & = & \frac{1}{Z_0}(\gamma_2 U) + C, & a_6 & = & \frac{1}{Z_0}(\gamma_2 V + W). \end{array} \tag{13}$$

5.2 The control methods

Several possibilities exist to accomplish the required control. We will examine two of them in detail here. The first one attempts to set certain affine parameters to zero; the second one zeroes a function of the affine parameters. The latter is more effective due to reasons to be explained soon; we validate this observation by simulation results presented in the next section.

5.2.1 Angular acceleration control

We have,

$$a_1 = -\frac{U}{Z_0} - B, a_4 = -\frac{V}{Z_0} + A.$$

If we assume that the external motion is purely translational, the values of a_1 and a_4 will be zero when the alignment has been achieved. This assumes that the first-order development is done with respect to the center of the image. Examining the expressions for a_1 and a_4, in the absence of rotational velocity, if we control the values of a_1 and a_4 to reach zero, the alignment will be achieved. It is this observation that motivates the following derivation. The flaw in this observation is that the rotational velocity will *not* be zero once control has begun. We will examine later the consequence of this flaw.

The derivatives are given by

$$\begin{aligned} \dot{a_1} &= -\frac{\dot{U}}{Z_0} + \frac{U\dot{Z_0}}{Z_0^2} - \dot{B} \\ \dot{a_4} &= -\frac{\dot{V}}{Z_0} + \frac{V\dot{Z_0}}{Z_0^2} + \dot{A}. \end{aligned}$$

The change in the velocity components due to the control rotation can be determined. The components of the velocity, which remain constant in a global coordinate system, change however in the camera coordinate system because of the rotation of the camera axes. Since the rotational velocity is $\Omega = (A, B, C)$ about the three axes, the change in the translational velocity $T = (U, V, W)$ is simply the cross product

$$\dot{T} = -\Omega \times T. \tag{14}$$

In detail,

$$\begin{pmatrix} \dot{U} \\ \dot{V} \\ \dot{W} \end{pmatrix} = \begin{pmatrix} CV - BW \\ AW - CU \\ BU - AV \end{pmatrix}. \tag{15}$$

Thus, we have

$$\begin{aligned} \dot{a_1} &= \frac{1}{Z_0}(BW - CV) + \frac{U}{Z_0}\frac{\dot{Z_0}}{Z_0} - \dot{B}, \\ \dot{a_4} &= \frac{1}{Z_0}(CU - AW) + \frac{V}{Z_0}\frac{\dot{Z_0}}{Z_0} + \dot{A}. \end{aligned}$$

Observing that

$$\frac{U}{Z_0} = -a_1 - B, \text{and } \frac{V}{Z_0} = -a_4 + A,$$

and simplifying, we get

$$\begin{pmatrix} \dot{a_1} \\ \dot{a_4} \end{pmatrix} = L \begin{pmatrix} A \\ B \end{pmatrix} + \begin{pmatrix} -\dot{B} \\ \dot{A} \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}, \tag{16}$$

where

$$\begin{aligned} L &= \begin{pmatrix} -C & \frac{W}{Z_0} - \frac{\dot{Z}_0}{Z_0} \\ -(\frac{W}{Z_0} - \frac{\dot{Z}_0}{Z_0}) & -C \end{pmatrix} \\ b_1 &= a_4 C - a_1 \frac{\dot{Z}_0}{Z_0}, \text{and} \\ b_2 &= -a_1 C - a_4 \frac{\dot{Z}_0}{Z_0}. \end{aligned}$$

When the planar approximation to the viewed surface does not have a large angle of inclination with respect to the camera, [1]

$$\frac{1}{\tau_c} = \frac{W}{Z_0} \approx -\frac{\dot{Z}_0}{Z_0} \approx \frac{a_2 + a_6}{2}, \quad C \approx a_r = \frac{a_3 - a_5}{2}. \tag{17}$$

where τ_c is used to denote the instantaneous *time-to-collision.*

Using these simplifications, we get,

$$\begin{aligned} L &\approx L_a = \begin{pmatrix} -a_r & \frac{2}{\tau_c} \\ -\frac{2}{\tau_c} & -a_r \end{pmatrix}, \\ b_1 &\approx b_{1a} = a_4 a_r + \frac{a_1}{\tau_c}, \\ b_2 &\approx b_{2a} = -a_1 a_r + \frac{a_4}{\tau_c}. \end{aligned}$$

For this, the task function is, as before,

$$\mathbf{e} = \mathbf{s} - \mathbf{s}^*,$$

where

$$\mathbf{s} = \begin{pmatrix} a_1 \\ a_4 \end{pmatrix}.$$

For an exponential decay of the task function,

$$\begin{pmatrix} \dot{a_1} \\ \dot{a_4} \end{pmatrix} = -\lambda \begin{pmatrix} a_1 - a_1^* \\ a_4 - a_4^* \end{pmatrix}.$$

[1] The approximations are also valid near convergence ($U \approx 0, V \approx 0$).

With all these simplifications, and rearranging the terms in Eqn. 16, we obtain the following control law:

$$\begin{pmatrix} -\dot{B} \\ \dot{A} \end{pmatrix} = -\lambda \begin{pmatrix} a_1 - a_1^* \\ a_4 - a_4^* \end{pmatrix} - L_a \begin{pmatrix} A \\ B \end{pmatrix} - \begin{pmatrix} b_{1a} \\ b_{2a} \end{pmatrix}. \tag{18}$$

For the alignment task, $a_1^* = a_4^* = 0$. Thus, Eqn. 18 can be used to calculate the control angular accelerations required to achieve the desired values for the affine parameters a_1 and a_4.

There are certain problems in using this control law; the first is that it requires the control of angular acceleration. The camera control mechanism should have this facility in order to be able to use this control law. Secondly, to compute the control acceleration, we need to know the instantaneous angular velocities A and B; these can be potentially computed by integrating the angular acceleration that has been already applied. Lastly, the angular velocities are not guaranteed to be zero on convergence; in other words, it is possible to obtain a *fixating* situation where the angular velocity is a specific function of the translation (see Appendix 2 for details), resulting in maintaining the velocity at the image plane origin to be zero (which would ensure that a_1 and a_4 are both zero). Thus, the required task of alignment is not accomplished; instead, *fixation* ensues. The last problem can be resolved by continuing to apply the control until the angular velocity values are zero; naturally, this requires the monitoring of the angular velocity either by integration of the applied acceleration control or by external means.

Yet another possibility for angular acceleration control is to add additional parameters in the control law. Examining the equations for the 2D affine motion parameters (Eqns. 13), we find that a_3 and a_5 are suitable because they involve U and V, whereas a_2 and a_6 are not because they involve in addition, W. We have,

$$a_3 = \frac{\gamma_2 U}{Z_0} + C \ , \ a_5 = \frac{\gamma_1 V}{Z_0} - C.$$

These parameters are zero when alignment is achieved, provided that the rotation about the optical axis, namely C, is not controlled, and thus set to zero. It can be shown easily that

$$\begin{pmatrix} \dot{a_3} \\ \dot{a_5} \end{pmatrix} = \begin{pmatrix} \gamma_2 C & -\frac{2\gamma_2}{\tau_c} \\ \frac{2\gamma_1}{\tau_c} & \gamma_1 C \end{pmatrix} \begin{pmatrix} A \\ B \end{pmatrix} + \begin{pmatrix} -Ca_4 + \frac{a_1}{\tau_c} \\ Ca_1 - \frac{a_4}{\tau_c} \end{pmatrix}.$$

The equations for $\dot{a_3}$ and $\dot{a_5}$ do not contain the acceleration terms! This precludes combining $\dot{a_3}$ and $\dot{a_5}$ with $\dot{a_1}$ and $\dot{a_4}$. In addition, the linear terms a_3 and a_5 are more sensitive to noise than the constant terms a_1 and a_4. The simulation results in Section 6.1 confirm our observations about the drawback of the angular acceleration methods presented so far.

5.2.2 Angular velocity control

One possibility is to treat the angular accelerations as being zero. Thus, we can set

$$\dot{A} = 0, \dot{B} = 0$$

in the equations in Eqn. 16. This will permit us to obtain a control law wherein the control is performed on the angular velocity instead of the angular acceleration:

$$\begin{pmatrix} A \\ B \end{pmatrix} = -\lambda L_a^+ \begin{pmatrix} a_1 - a_1^* \\ a_4 - a_4^* \end{pmatrix} - L_a^+ \begin{pmatrix} b_{1a} \\ b_{2a} \end{pmatrix}, \tag{19}$$

where L_a^+ is the pseudo-inverse of the matrix L_a. Using $C = a_r = 0$, we obtain the following simpler form:

$$\begin{pmatrix} A \\ B \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 0 & -\tau_c \\ \tau_c & 0 \end{pmatrix} \left(\lambda \begin{pmatrix} -a_1 \\ -a_4 \end{pmatrix} - \begin{pmatrix} \frac{a_1}{\tau_c} \\ \frac{a_4}{\tau_c} \end{pmatrix} \right). \tag{20}$$

This control method will not have the *fixation* problem encountered earlier, and is implementable. The reason the fixation problem disappears here is due to the controlling of the angular velocities; such a control guarantees that the affine parameters a_1 and a_4 go to zero at convergence. This alone will not avoid the fixation, but the above control law ensures that the angular velocities also go to zero, and this avoids the fixating situation (also see the discussion following Eqn. 18 and Appendix. 2). We find that using a different approach, a closely related control law is obtained in a more direct manner. We describe this alternative approach now.

Consider the two "parameters" $U_z = \frac{U}{Z_0}$ and $V_z = \frac{V}{Z_0}$. If we apply control in such a way to result in zero values for these variables, we will achieve the goal of setting the components U and V of the translational velocity to zero (the tacit assumption is that infinite depth does not occur). This is an intuitive set of parameters to control, because the quantities that we desire to control (namely U and V), are directly related to the chosen parameters.

We would like to have the controlled parameters follow an exponential decay towards the target value of zero. The derivatives of these parameters are given by

$$\begin{pmatrix} \dot{U}_z \\ \dot{V}_z \end{pmatrix} = \begin{pmatrix} \frac{\dot{U}}{Z_0} - \frac{U}{Z_0}\frac{\dot{Z}_0}{Z_0} \\ \frac{\dot{V}}{Z_0} - \frac{V}{Z_0}\frac{\dot{Z}_0}{Z_0} \end{pmatrix}. \tag{21}$$

Substituting for $\dot{U}$ and $\dot{V}$ (from Eqn. 15) in Eqn. 21, we get

$$\begin{pmatrix} \dot{U}_z \\ \dot{V}_z \end{pmatrix} = \begin{pmatrix} 0 & -\frac{W}{Z_0} + \frac{\dot{Z}_0}{Z_0} \\ \frac{W}{Z_0} - \frac{\dot{Z}_0}{Z_0} & 0 \end{pmatrix} \begin{pmatrix} A \\ B \end{pmatrix} + \begin{pmatrix} C\frac{V}{Z_0} + a_1\frac{\dot{Z}_0}{Z_0} \\ -C\frac{U}{Z_0} + a_4\frac{\dot{Z}_0}{Z_0} \end{pmatrix}.$$

For an exponential decay of the task function, the following control law is obtained using the approximations (and setting C to zero, since it is not controlled) in Eqn. 17:

$$\begin{pmatrix} A \\ B \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 0 & -\tau_c \\ \tau_c & 0 \end{pmatrix} \left(\lambda \begin{pmatrix} U_z \\ V_z \end{pmatrix} - \begin{pmatrix} \frac{a_1}{\tau_c} \\ \frac{a_4}{\tau_c} \end{pmatrix} \right). \tag{22}$$

Let us finally note that the observations U_z and V_z are given by

$$\begin{aligned} U_z &= -a_1 - B, \text{ and} \\ V_z &= -a_4 + A, \end{aligned}$$

where we use the previous measured values for A and B (under the conditions presented here[2], it is nothing but the control rotational velocity applied at the preceding instant).

We thus obtain a control law which is almost the same as the one given in Eqn. 20.

[2]We assume there is no external agent causing rotational velocity.

5.3 Calculating the 2D affine parameters

The 2D affine parameters can be calculated directly from the spatial and temporal derivatives of the image sequence. Recalling that the model is

$$\begin{aligned} u(x,y) &= a_1 + a_2 x + a_3 y, \\ v(x,y) &= a_4 + a_5 x + a_6 y, \end{aligned} \tag{23}$$

and the motion constraint equation [22] is

$$I_x u(x,y) + I_y v(x,y) + I_t = 0.$$

It is easy to see that a direct substitution of $u(x,y)$ and $v(x,y)$ from Eqn. 23 in the last equation yields one equation in six unknowns. Thus, in principle, knowing the spatial and temporal derivatives at six points (belonging to a region whose projected motion satisfies the affine model), we can solve for all the six affine parameters. In practice, for robustness reasons, a large number of points are considered, and a standard procedure for solving an overconstrained set of equations is used to determine the parameters.

The procedure presented above is the simplest possible approach; other methods using robust computations [30] have been developed, but they are time-consuming. The sensitivity of the affine parameters has been analyzed in [28]. In our experiments, we use the simple procedure presented above, with points taken from all over the image.

6 Experiments

Three of the alignment methods, namely the one using the FOE and the ones using angular velocity control, have been implemented in a real system. We carried out simulation studies on all the methods before choosing those that were implemented in the real system; our studies indicated that the method we discarded (see Fig. 4) has problems that we describe in the following section.

6.1 Simulation experiments

Here we present the approach to the simulation, the results of the simulation, and a brief discussion of the results.

6.1.1 Approach

The simulation was carried out using a discrete-time approach. Time ticks were chosen at unit intervals, and the control and the state of the system were calculated at these instants in time. This approach would closely approximate a real implementation with no asynchronous activities (i.e., time duration of actions such as acquiring the images, and computing the control consume roughly constant time in different iterations).

All the methods were implemented using the software package MATLAB. The computations were exact (with an exception we will explain soon). No noise was added during the simulations to avoid the tedium of modeling the noise correctly; instead, we carried out the validated methods on a real system.

Parameters that change continuously over time were discretized with a fine resolution. For example, the vector T undergoes a continuous change while the control is being applied; this change is given by Eqn. 14, repeated here:

$$\dot{T} = -\Omega \times T.$$

During the simulation, at a given time tick, we need to recalculate the translation vector T to account for the change since the last time tick; we divided the time interval between the two ticks into 500 equal parts and incrementally modified T.

We describe in the following paragraphs the results of the simulations, and how the results were used to validate the methods. Four different graphs are used to present the results. The first one contains curves showing the components of the 3D translational velocity in the camera coordinate system. For proper alignment, we expect the curves corresponding to U and V to go to zero, and the curve for W to achieve a large postive value equal to the speed (magnitude of the velocity) of the camera. The second graph shows the corresponding variation of the angular velocity components; we expect them to go to zero at convergence. The third graph shows the variation of the norm of the task function vector $(s - s^*)$ (see Eqn. 3, $M = I$) constituting the controlled parameters; this should go to zero at convergence with an exponential decay. The fourth and final graph shows the variation of the angle between the translation vector T and the optical axis Z of the camera; this should go to zero as well, at convergence.

6.1.2 Results

For all the experiments, an initial translational velocity of $[2, 8, 2]$ was used; this choice is not critical, and was used only for demonstration purposes. Other initial translational velocity were experimented with, and the results were qualitatively similar.

The curves in Fig. 7 show the results from simulating the FOE-based algorithm. A λ value of 0.2 was used. The translational velocity is controlled as desired, with U and V going to zero, leaving W as the only non-zero component. The angular velocities are computed correctly to provide the necessary correction to bring about the alignment. The "clipping" of the angular velocities is related to the qualitative nature of the FOE computation that is simulated. That is, when the FOE is outside the image, we can only compute the point on the border which is in the direction of the FOE from the center of the image. Thus, the angular velocity remains clipped until the FOE "enters" the image. The task function, and the angular separation betwen the optical axis and the translation vector go to zero, as expected.

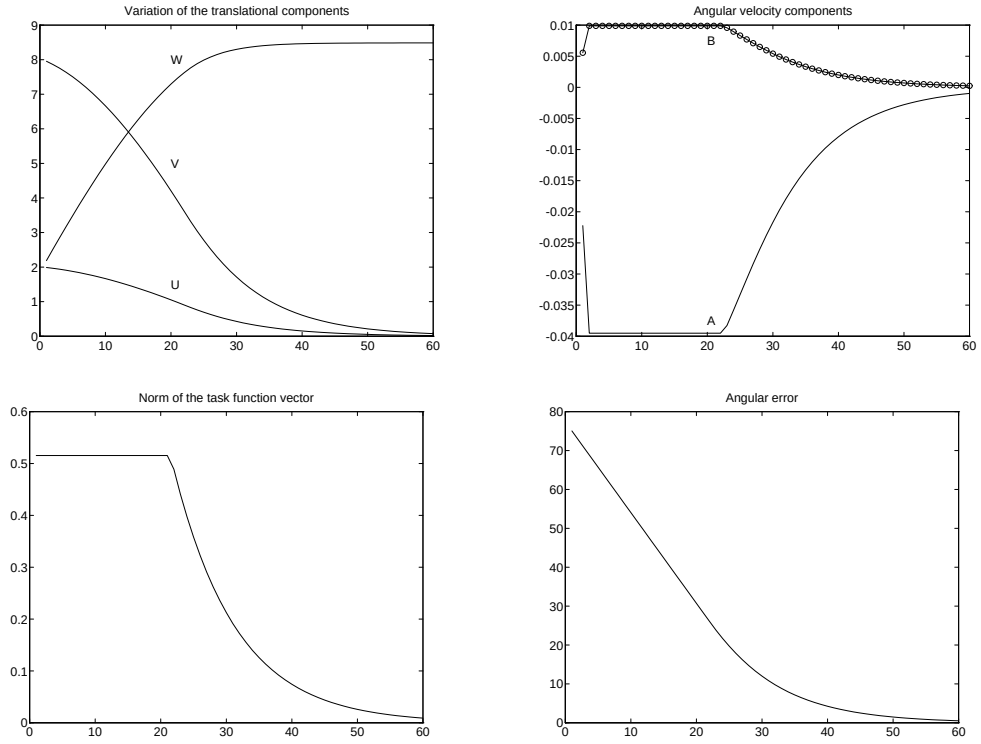


Figure 7: The FOE-based method: results. The top row shows the change in the velocity components (translational velocity components on the left, and the angular velocity components on the right). The bottom row shows the task function decrease on the left, and on the right, the angle between the translational direction and the optical axis. The flat parts of the curves on top right and bottom left are the due to a feature of the simulated FOE detection method that determines only the direction towards an FOE outside the image.

In the next simulation, the angular acceleration is calculated using Eqn. 18. We assume that each iteration corresponds to one time-step and that the angular velocity can be calculated by discrete summation of the angular acceleration. A λ value of 0.1 was used. Translational velocity of [6,8,10] focal units was used. The problems with the angular acceleration are illustrated by the results shown in Fig. 8. The curves are organized as before. It can be seen that alignment ensues ($U = V = 0$), but W is negative, resulting in a backwards translation. Also, the task function rapidly goes towards zero and then shoots up again. This puzzling result comes about due to the tendency for the system to go towards *fixation* (corresponding to the first minimum of the task function attained after the second iteration), as explained following the derivation of Eqn. 18 (also see Appendix 2). Thus, the control goes in the opposite sense to what is desired, as can be easily verified from the fact that the control to achieve fixation is opposite to that for alignment in the forward direction. This makes the method unsuitable for our purposes. There might be the interesting possibility of using this method to achieve fixation, but such a possibility is not explored here; besides, in the form presented here, the method is unstable at fixation unless the task function is carefully monitored to cut off control when fixation (the first minimum) is encountered.

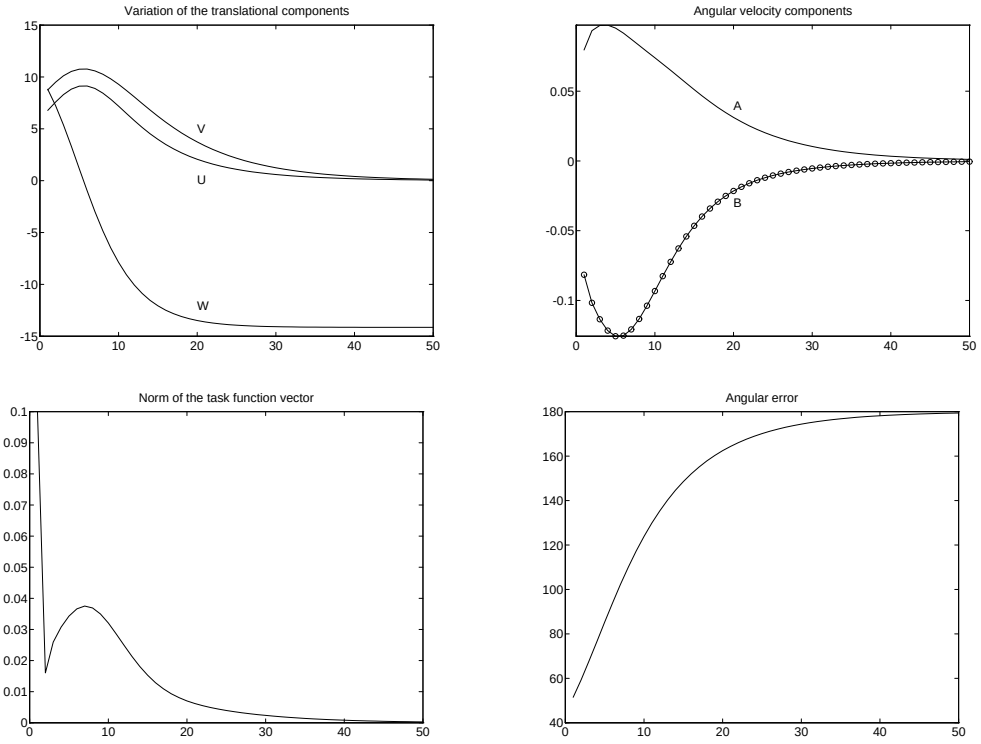


Figure 8: The angular acceleration based method: results. It can be seen clearly on the task function curve at bottom left the tendency to fixate, achieving the first minimum; the procedure is continued, resulting in "pulling out" of this minimum and achieving the alignment, but in the reverse direction, as evidenced by angular difference curve at bottom right that shows 180 degrees at convergence.

We now examine the possibility suggested at the beginning of Section 5.2.2, that of setting the angular acceleration values to zero, thereby obtaining a control law for the angular velocity (see Eqn. 20). Results from a simulation using the control law in Eqn. 20 are presented in Fig. 9. The initial translational velocity is $[2, 8, 2]$ focal units, and a λ of 0.05 is used. Here we see at the beginning of the simulation the perturbations on the task function arising from the assumption that the angular acceleration is zero. During this initial phase, large change in angular velocity components dominates the parameters a_1 and a_4, resulting in errors because the angular acceleration quantities are non-negligible. Another possibility is that the λ value chosen is too large. Repeated simulations with various values of λ suggests that this is not the case. In the real experiments we present later, we will see that the initial perturbation may not be a critical issue in a practical implementation.

The final simulation we present is the control using angular velocity, from Eqn. 22. The curves are shown in Fig. 10 with initial translational velocity of $[2, 8, 2]$ focal units. The λ value was 0.1. The control results in the goal configuration without the undesirable effects in Fig. 8 and Fig. 9.

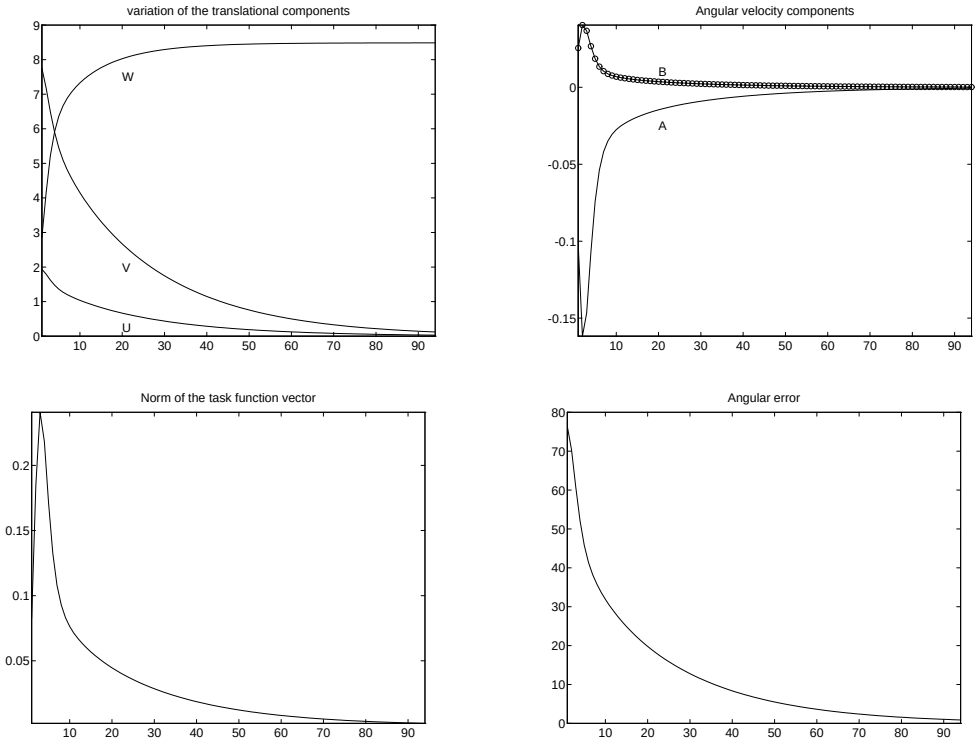


Figure 9: The angular velocity based method of Eqn. 20: results. We note that the velocity components are perturbed by the assumption that the angular acceleration is zero, resulting in an overshoot in the norm of the task function vector.

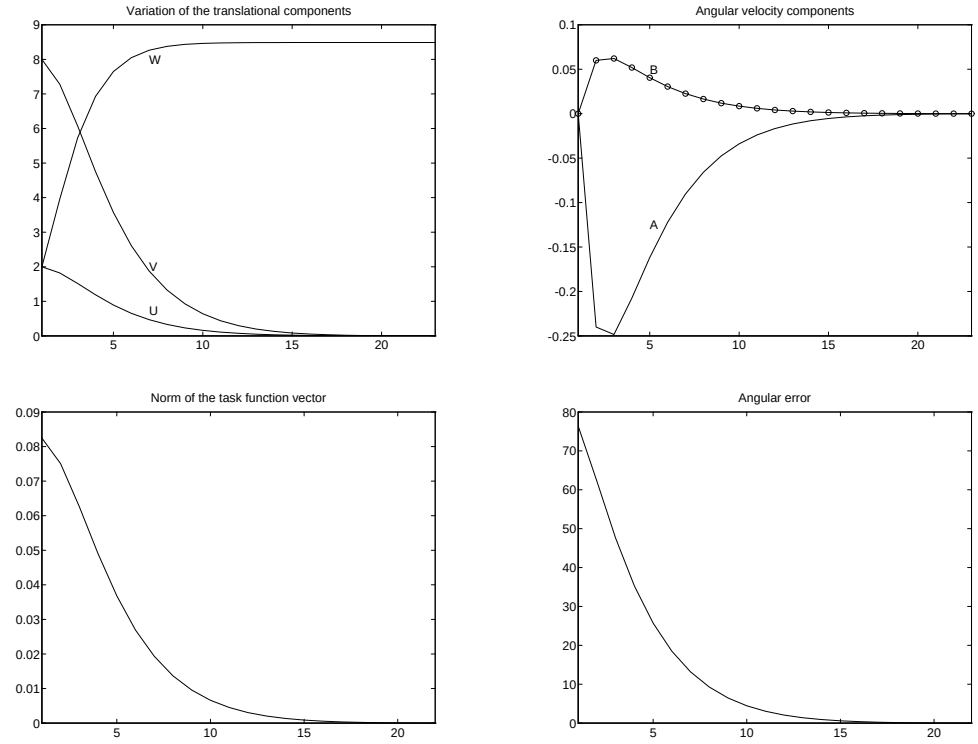


Figure 10: The angular velocity based method of Eqn. 22 : results.

6.1.3 Discussion

Our study using the simulations showed that the FOE-based method (Eqn. 8) and the angular velocity based control (Eqn. 22) are stable and achieve the convergence rapidly. On the other hand, the angular acceleration method's undesirable performance is confirmed by the simulations. Therefore, for the experiments with a real set-up, we implemented only the FOE-based method and the angular velocity based methods. The simulation results for the FOE-based method and the angular velocity based methods are qualitatively correct, and should be treated as such; this is so because we have not attempted to make the simulations completely realistic. However, the real experiments we present in the next section confirm that the methods are applicable in realistic situations.

6.2 Real experiments

In the experiments, we used a camera with a field of view of about 35 degrees mounted on a six degrees-of-freedom cartesian robot AFMA (see Fig. 11). The camera can be positioned and oriented in the workspace with an accuracy of 0.5 mm and 0.05 degrees. The camera output is digitized by an image processing board (EDIXIA). For the experimental results reported here, the images are digitized and then subsampled by a factor of four (without any filtering) by the EDIXIA board and sent to the host (Sun Spark 10). The size of the images processed is 128×182 pixels. All the image processing and control velocity computations are carried out on the host and the computed control is transmitted to the robot controller. The transmissions to and from the host occur via a BIT 3 Sbus/VMEbus board. A minimum of ten milliseconds are required for the controls to take effect. About 100 milliseconds are required for the entire process of acquiring, subsampling and transferring an image. Note that for the implementation, the camera parameters obtained from calibration, namely the center of the image and the interpixel distances, are required.

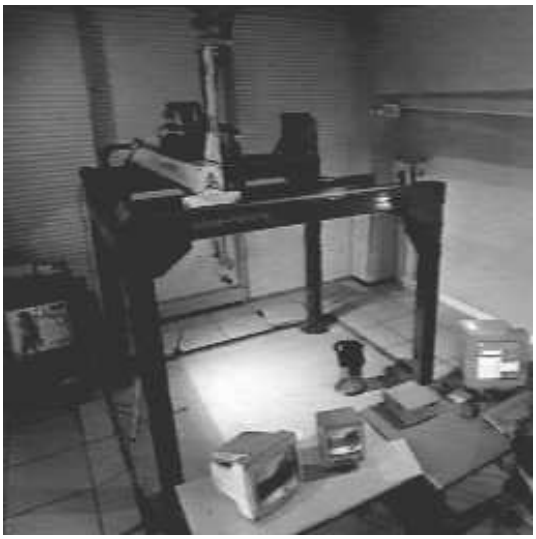


Figure 11: Experimental cell where all the experiments were conducted. See text for description.

Smoothing of the image is done before the computation of the spatial and temporal derivatives (always done with only two images) using a separable Gaussian filter whose discrete representation is

$$[0.0223211 \quad 0.229742 \quad 0.495015 \quad 0.229742 \quad 0.0223211],$$

which corresponds to a σ of 0.75. The derivatives are calculated using a $4 \times 4 \times 2$ cell in the x, y and t directions respectively, as explained in [22].

The experiments were conducted indoors; a sample image can be seen in Fig. 12. The translational motion was towards the floor with cluttered objects; the floor was not fronto-planar, but with an average angle of inclination in the range 45-70 degrees (note that it is 90 degrees for a frontal planar surface) between the floor surface and the optical axis.

Figure 12: A typical image in the sequence processed.

Numerous experiments were carried out using our implementation of the methods described. We only report results from representative experiments here.

6.2.1 The FOE method

The FOE method described in Section 4 has been implemented. The FOE is calculated as described in Section 4.1 (also see [10, 34]).

The robot is commanded to move the camera with a certain translational velocity. The control loop consists of the following steps which are repeated:

- obtain two successive images,
- compute the FOE location,
- compute the rotational velocity control required using the control law in Eqn. (8),
- and apply the control rotational velocity for a finite duration.

Note that the control is applied for only a finite duration during each iteration. This is because the qualitative method used for the FOE computation works only for pure translation. The total time spent in one iteration is about one second.

The result of a typical experiment is shown in Fig. 13. The plot shows the variation of the angle between the direction of translation T and the optical axis Z with respect to time. As expected, the angle decreases and converges to zero. The final error plotted in Fig. 13 is 0.25 degrees.

6.2.2 The Affine Parameters method

The affine parameters methods described in Section 5 has also been implemented. The affine parameters are computed using an over-constrained set of equations by considering the intensity derivatives from all over the image, thresholded by gradient magnitude to suppress contribution from relatively uniform regions where estimates are noisy.

The robot, as before, is commanded to move the camera with a certain translational velocity. The control loop consists of the following steps which are repeated:

- Obtain two successive images,
- compute the affine parameters of the flow field,
- compute the rotational velocity control required using the control law in Eqn. 20 or Eqn. 22,
- and apply the control rotational velocity.

Here, for each control law, two programs, one in which the control is applied for a finite duration and another in which the control is applied in a continuous manner, have been implemented. Each iteration took about one second.

The error plots from experiments using the two implementations are shown in Fig. 14 and Fig. 15. In both figures, on the left is the plot for the implementation where the control is applied for a finite duration (discrete control), and on the right, for continuous control. In detail, for the discrete control, while the two successive images are acquired, the control is withdrawn; this means that the affine parameters a_1 and a_4 provide only the translational velocity information, and hence the control computation is accurate. On the other hand, for the continuous control, a_1 and a_4 contain the rotational velocity terms in addition to the *observation*, namely the translational velocity terms (which is the information we need). As an effect of the inaccuracies introduced in the observations, the continuous control converges more slowly and less precisely. We finally note that the different assumptions (frontal planar surface, $a_r = 0$, etc; see Eqn. 17) made in deriving the control laws do not have any noticeable influence in the realization of the task.

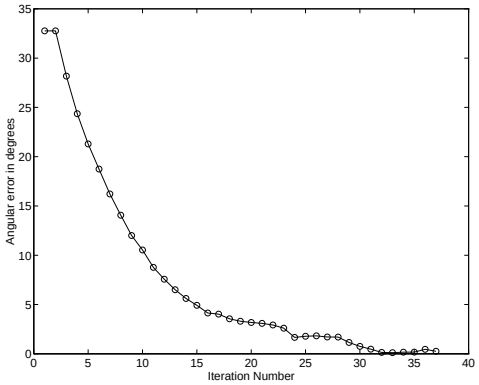


Figure 13: The angular error plot for the FOE method.

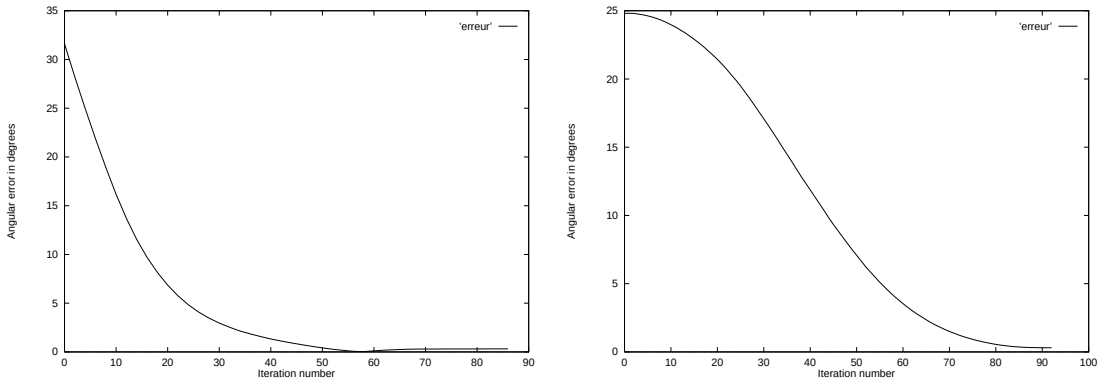


Figure 14: The angular error plots for the affine parameter method of Eqn. 20; on the left is the plot for the method using discrete control, and on the right is the plot for the method using continuous control.

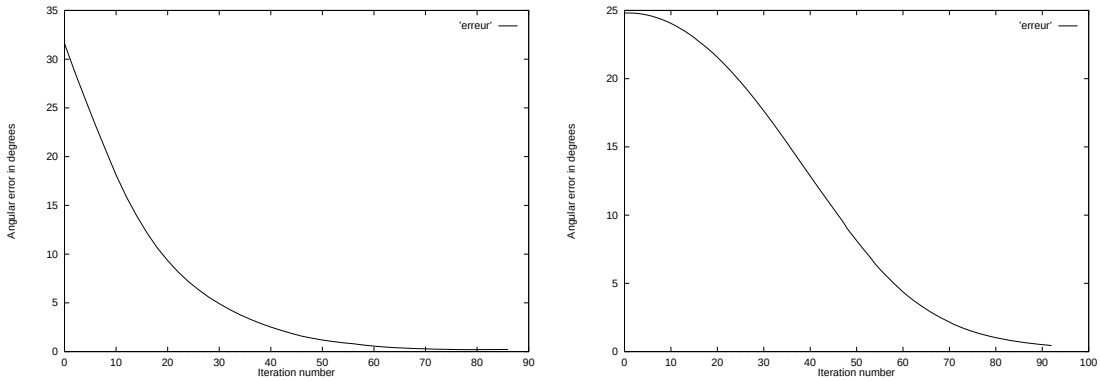


Figure 15: The angular error plots for the affine parameter method of Eqn. 22; on the left is the plot for the method using discrete control, and on the right is the plot for the method using continuous control.

6.3 An application

Here we discuss one application of the alignment task. Autonomous systems moving in an unknown environment must have the capabilities of detecting moving objects in a scene and of characterizing their motion in order to, for example, avoid such obstacles (for a navigation task), or to track them (in a surveillance task). Achieving such a processing using the information supplied by a camera attached to the system of interest is a difficult task whenever the camera is moving.

It is not always necessary to perform complete recovery of the 3D scene content and scene motion to perform the tasks such as those mentioned above. Goal-oriented procedures can provide fast, reliable and sufficient information. A qualitative dynamic scene analysis method [9] is an example. However, this particular method has one strong limitation. Useful qualitative motion information can be retrieved efficiently only for specific camera motion configuration (i.e., axial motion as far as translation is concerned). However, we cannot usually assume that the camera is aligned with the translation axis of the system on which it is mounted. It might be possible to rigidly align the camera by mechanical means; however, this would be at the cost of flexibility; i.e, certain degrees of freedom will be unavailable for other useful tasks such as panning or fixation, or for the use of active vergence such as in [20].

To improve this solution, the main idea is not to find a general solution which extends the previous one but to use system capabilities to circumvent the restrictions in order to be able to still use the initial method which is simple enough and known to be robust and efficient. This is where the alignment task plays a useful role. The alignment method is used to place the camera in the tractable configuration, and then the qualitative method of [9] is used to detect moving objects. For a more detailed presentation of this approach, the reader is referred to [10].

Three different kinds of experiments were conducted. The first is to demonstrate the active scheme in the absence of object motion (i.e., only the camera is moving). The second is to demonstrate the ability to achieve the alignment in the presence of moving objects. For this demonstration, we have not relied on the segmentation results of the qualitative scheme. In fact, the whole image, along with the moving object, is considered for the active alignment task. In the experiments shown, the moving object does not occupy a large portion of the image. The third experiment is to show the performance of the qualitative scheme. It is done independently of the alignment because a real-time implementation is still incomplete. But, we note that together these experiments convincingly demonstrate the validity and applicability of the approach in real world environment. All the experiments were carried out with the camera viewing the same scene.

The results of all the experiments are shown in Fig. 16. It can be seen that in both the cases (with and without object motion), the alignment has been achieved quite well. The motion of the object has caused a larger error in the alignment but we are confident that with the use of the motion segmentation part of the qualitative scheme, this error will be reduced.

In the third experiment, the optical axis of the camera was roughly aligned with its translational direction (there were errors of about four to eight degrees, in fact more than the residual

errors in the alignment experiment–see the plot in Fig. (16)).The motion interpretation by the qualitative scheme is shown in Fig. 16 (right). The box in the center of the image (see Fig. 12) was in motion, pulled by a cord. The interpretation has segmented the moving box (along with its shadows which have the same motion) and has correctly indicated that the motion is of the "lateral" type (as opposed to moving towards/away). The moving region has been rendered in gray.

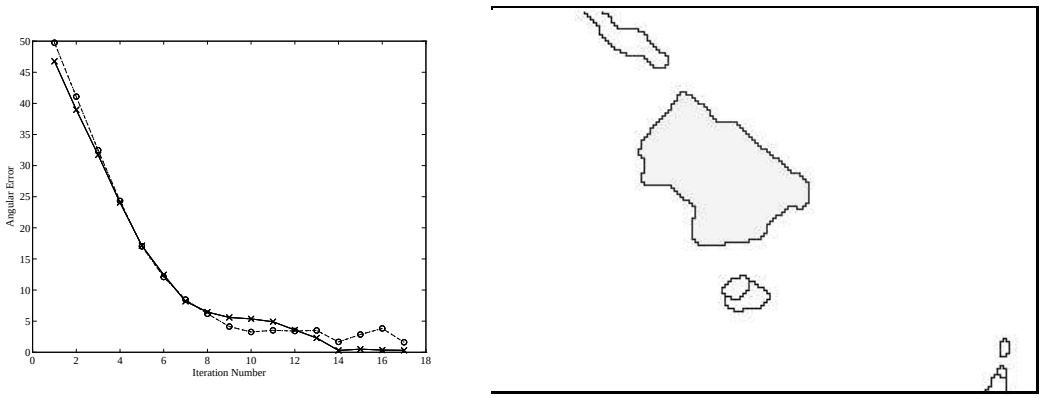


Figure 16: On the left is the error plot, showing the decreasing angle between the translational direction and the optical axis over iterations; the solid line corresponds to the case when the box was stationary; the dotted line is the result when the box was moving. See Fig. 12 to see the final image in the sequence. On the right is the motion interpretation; the moving region has been rendered in gray.

We believe that the demonstrated method is *purposive* [2]. But, to achieve the goal-oriented task of motion detection, we make use of the capabilities of the system within the framework of visual servoing. The general approach advocated is rather to couple active vision and qualitative vision. We think that this provides a rich source of solutions to be investigated.

7 Discussion

The method we have proposed here to control the viewing direction of a camera qualifies to be a "locomotor program" advocated by Lee and Lishman [26] to control locomotion by constant "monitoring in terms of the program and any deviations corrected by adjusting the ongoing program." We believe that our proposal provides a stable formalism for camera control using visual motion analysis, and based on the well-studied concept of visual servoing.

Barth and Tsuji [7] also describe a method for achieving alignment of the optical axis with the translational direction. The method relies on the ability of the camera to fixate at a scene point, and based on a simple analysis of the optic flow field near the fixation point, saccades are made to eventually achieve alignment. They present a theoretical justification for their method, and show simulation and real experiments. Their scheme differs from ours in that they require the camera to be able to track a 3D point (fixation); the saccade is calculated in a qualitative

fashion, and, considered with the qualitative method we use to calculate the FOE, demonstrates the feasibility of control schemes relying on simple information from images. Also, their scheme requires the computation of optic flow; even though their method does not require an accurate flow field, they do not indicate how to take advantage of this.

In this paper, we restricted ourselves to the pure translation situation. It would be interesting to examine the general case where there is rotation also. We have also not examined the possibility of predicting the motion. Standard methods such as Kalman filtering could be employed to predict the focus of expansion or the affine motion parameters; this would increase the stability of the methods.

In the affine parameter method, we assume that the affine approximation to the optical flow field is valid. This is supported by several useful methods developed based on the affine approximation [9, 28]. Nevertheless, this approximation can fail for the entire image when there are objects located at very different depth in the scene, or moving objects of significant size. Motion-based segmentation of the image into regions [9] could be one possible solution, but far too complex to be implemented in such a closed-loop procedure. However, recently designed multi-resolution robust estimation methods such as [30] can cope up with these situations.

Our implementation remains limited because the processing is performed on the host by transferring the images from the image-processing board. All the timings noted here are expected to improve drastically when the implementation (in progress) of the computations on the image-processing board is completed.

8 Conclusions

In this report, we have proposed the use of motion information in the visual servoing framework where only geometric information has been used so far. Two control schemes, one using the focus of expansion, and the other using affine motion parameters, were presented. Experimental results from a camera mounted on a robot serve to validate our proposal. The control method presented here has been used successfully [10] to provide an interesting new direction of coupling between qualitative and active visual methods. Future work includes processing on the image-processing board to improve speed and performance, and to investigate other forms of tight coupling between camera behavior and motion information.

We believe that the work presented here provides a starting point for formal approaches to closed-loop control using motion information; these are expected to be useful for active visual tasks involving a camera undergoing motion and/or is monitoring moving objects.

Acknowledgements

The first author was supported by an INRIA post-doctoral fellowship during the period of this work. The authors thank Sylvain Cochepain who implemented in part some of the experimental work reported here.

APPENDIX

1 Consistency in direction between normal and optical flow vectors

Consider the optical flow vector $\vec{V}$ at a point (x, y) on the image plane. Let the local image gradient at (x, y) be along the unit normal vector $\vec{n}$, and $\vec{e_x}$ and $\vec{e_y}$ be the unit vectors along the x and y axes respectively. Then the normal flow is $\vec{V_n} = (\vec{V} \cdot \vec{n})\vec{n}$. If $\vec{V} \cdot \vec{e_x} > 0$, then $\vec{V_n} \cdot \vec{e_x} > 0$ with high probability; similarly for the case $\vec{V} \cdot \vec{e_x} < 0$, and for $\vec{V} \cdot \vec{e_y}$. The situation is described pictorially in Fig. 17. In the figure, without loss of generality, we have chosen the flow vector $\vec{V}$ in one of the quadrants. If the angle subtended by $\vec{V}$ with the x axis is Φ, then the probability that the sign of $\vec{V_n} \cdot \vec{n}$ agrees with the sign of $\vec{V} \cdot \vec{e_x}$ is given by

$$\frac{\pi - \Phi}{\pi},$$

assuming a uniform distribution of the direction of $\vec{n}$. In other words, the signs will disagree if the normal vector is found in the shaded region of Fig. 17, and this happens with probability Φ/π. At the same time, the probability that the sign of $\vec{V_n} \cdot \vec{e_y}$ agrees with the sign of $\vec{V} \cdot \vec{e_y}$ is

$$\frac{\frac{\pi}{2} + \Phi}{\pi}.$$

Since $\Phi \leq \pi/2$, the probability of being correct is more than half (in fact, if one of them is close to half, the other one will be close to one, which means that if the sign of one component (x, say) being correct is close to "chance," then the sign of the other component (y) will be correct with a probability close to 1). For clarity of explanation, we have omitted the degenerate cases $\Phi = 0$ and $\Phi = \pi/2$.

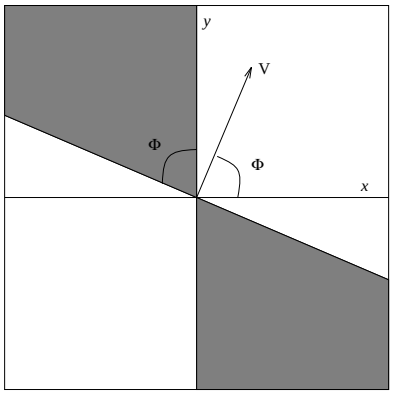


Figure 17: Illustration for the agreement of the signs of $\vec{V} \cdot \vec{e_x}$ and $\vec{V_n} \cdot e_x$.

2 The fixating situation

The constant part of the 2D affine motion model is given by:

$$a_1 = -\frac{U}{Z_0} - B \ , \ a_4 = -\frac{V}{Z_0} + A.$$

If we set $a_1 = a_4 = 0$, we get the following:

$$A = \frac{V}{Z_0} \ , \ B = -\frac{U}{Z_0}$$

Under these conditions, the affine parameters a_1 and a_4 are zero (i.e., the image flow at the center of the region where the 2D affine parameters are calculated is zero, but the angular velocity components A and B are non-zero, and are proportional to the components of the translational velocity in such a way as to retain the point at the center in the same position. This is clearly a *fixating* situation, as opposed to an *aligned* situation where the angular velocity components will be zero.

References

[1] P.K. Allen, A. Timcenko, B. Yoshimi, and P. Michelman. Automated tracking and grasping of a moving object with a robotic hand-eye system. *IEEE Trans. on Robotics and Automation*, 9(2):152–165, 1993.

[2] J. Aloimonos. Purposive and qualitative vision. In *Proc. DARPA Image Understanding Workshop*, pages 816–828, 1990.

[3] Y. Aloimonos and Z. Duric. Active egomotion estimation: a qualitative approach. In G. Sandini, editor, *Proceedings of ECCV*, pages 497–510, Santa Margherita, May 1992.

[4] Y. Aloimonos, I. Weiss, and A. Bandyopadhyay. Active vision. *IJCV*, 1:333–356, 1988.

[5] R. Bajcsy and M. Campos. Active and exploratory perception. *CVGIP: Image Understanding*, 56(1):31–40, July 1992.

[6] D.H. Ballard and C.M. Brown. Principles of animate vision. *CVGIP: Image Understanding*, 56(1):3–21, July 1992.

[7] M. J. Barth and S. Tsuji. Egomotion determination through an intelligent gaze control strategy. *IEEE Trans. on Sys. Man and Cyb.*, 23(5):1424–1432, 1993.

[8] J.R. Bergen, P. Anandan, K.J. Hanna, and R. Hingorani. Hierarchical model-based motion estimation. *Proc. 2nd European Conf. on Computer Vision, Genova*, pp 237–252, May 1992.

[9] P. Bouthemy and E. François. Motion segmentation and qualitative dynamic scene analysis from an image sequence. *IJCV*, 10(2):157–182, 1993.

[10] P. Bouthemy and V. Sundareswaran. Qualitative motion detection with a mobile and active camera. In A. G. Constantinides et. al., editor, *Procs of the Intl. Conf. on Digital Signal Processing*, pages 444–449, Cyprus, July 1993.

[11] F. Chaumette, S. Boukir, P. Bouthemy, and D. Juvin. Optimal estimation of 3d structures using visual servoing. In *Proc. CVPR*, pages 347–354, Seattle, June 1994.

[12] F. Chaumette and A. Santos. Tracking a moving object by visual servoing. In *Proc. 12th world congress IFAC, Vol. 4*, pages 409–414, Sydney, July 1993.

[13] D. Coombs and C. Brown. Real-time binocular smooth pursuit. *International Journal of Computer Vision*, 11(2):147–164, 1993.

[14] D. Coombs and K. Roberts. Centering behavior using peripheral vision. In *CVPR*, pages 440–445, New York City, June 1993.

[15] C. Samson, B. Espiau, and M. Le Borgne. *Robot Control: the task function approach.* Oxford University Press, 1991.

[16] B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. *IEEE Trans. on Robotics and Automation*, 8(3):313–326, 1992.

[17] J.T. Feddema and O.R. Mitchell. Vision-guided servoing with feature-based trajectory generation. *IEEE Trans. on Robotics and Automation*, 5(5):691–700, October 1989.

[18] C. Fermüller and Y. Aloimonos. The role of fixation in visual motion. *IJCV*, 11(2):165–186, June 1993.

[19] E. Grosso and D. Ballard. Head-centered orientation strategies in animate vision. In *Proc. of the 4th ICCV*, pages 395–402, Berlin, May 1993.

[20] E. Grosso, M. Tistarelli, and G. Sandini. Active/dynamic stereo for navigation. In G. Sandini, editor, *Proceedings of ECCV*, pages 516–525, Santa Margherita, May 1992.

[21] K. Hashimoto. *Visual Servoing.* Volume 7 of *World Scientific Series in Robotics and Automated Systems*, World Scientific, 1993.

[22] B.K.P Horn. *Robot Vision.* The MIT Press, 1987.

[23] J. Santos-Victor, G. Sandini, F. Curotto, and S. Garibaldi. Divergent stereo for robot navigation: learning from bees. In *CVPR*, pages 434–439, New York City, June 1993.

[24] J. J. Koenderink and A. J. Van Doorn. Invariant properties of the motion parallax field due to the movement of rigid bodies relative to an observer. *Optica Acta*, 22(9):773–791, 1975.

[25] E. Krotkov. Focusing. *International Journal of Computer Vision*, 1:223–237, 1987.

[26] D. N. Lee and R. Lishman. Visual control of locomotion. *Scandinavian Journal of Psychology*, 18:224–230, 1977.

[27] D.W. Murray, P.F. McLauchlan, I.D. Reid, and P.M. Sharkey. Reactions to peripheral image motion using a head/eye platform. In *Proc. of the 4th ICCV*, pages 403–411, Berlin, May 1993.

[28] S. Negahdaripour and S. Lee. Motion recovery from image sequences using only first order optical flow information. *IJCV*, 9(3):163–184, December 1992.

[29] C. Nelson and J. Aloimonos. Obstacle avoidance using flow field divergence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11:1102–1106, 1989.

[30] J.-M. Odobez and P. Bouthemy. Robust multiresolution estimation of parametric motion models in complex image sequences. In *Proc. 7th Conf. Eusipco*, Edinburgh, Scotland, September 1994.

[31] T. J. Olson and D. J. Coombs. Real-time vergence control for binocular robots. *International Journal of Computer Vision*, 7(1):67–89, 1991.

[32] K. Pahlavan, T. Uhlin, and J-O. Eklundh. Dynamic fixation. In *Proc. of the 4th ICCV*, pages 412–419, Berlin, May 1993.

[33] N. Papanikolopoulos, P. Khosla, and T. Kanade. Visual tracking of a moving target by a camera mounted on a robot: a combination of control and vision. *IEEE Trans. on Robotics and Automation*, 9(1):14–35, 1993.

[34] V. Sundareswaran. Qualitative visual motion analysis. manuscript, in preparation.

[35] V. Sundareswaran, P. Bouthemy, and F. Chaumette. Active camera alignment by visual servoing. In *Proc. of Quatrièmes Journées ORASIS*, pages 17–20, Mulhouse, Oct 1993.

[36] V. Sundareswaran, P. Bouthemy, and F. Chaumette. Active camera alignment using dynamic image parameters. In O. Eklundh, editor, *Lecture Notes in Computer Science 801: Computer Vision –ECCV '94*, pages 111–116, Springer-Verlag, 1994.

[37] V. Sundareswaran, F. Chaumette, and P. Bouthemy. Visual servoing using image motion information. In W. Martin, editor, *Proc. IAPR/ IEEE workshop on visual behaviors*, pages 102–106, Seattle, June 1994.

[38] L.E. Weiss, A.C. Sanderson, and C.P. Neuman. Dynamic sensor-based control of robots with visual feedback. *IEEE Trans. on Robotics and Automation*, 3(5):404–417, October 1987.

[39] A. Yuille and D. Geiger. Stereo and controlled movement. *International Journal of Computer Vision*, 4:141–152, 1990.

Unité de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unité de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unité de recherche INRIA Rhône-Alpes, 46 avenue Félix Viallet, 38031 GRENOBLE Cedex 1
Unité de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unité de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
ISSN 0249-6399