Correspondence

Image-Based Robot Task Planning and Control Using a Compact Visual Representation

Carlo Colombo and Benedetto Allotta

Abstract—**In this paper, we present an approach for the design and control of both reflexive and purposive visual tasks. The approach is based on the bidimensional appearance of the objects in the environment and explicitly takes into account independent object motions. A linear model of camera–object interaction is embedded in the control scheme, which dramatically simplifies visual analysis and control by reducing the size of visual representation. We describe the implementation of three visual tasks of increasing complexity, obtained with the proposed scheme and based on active contour analysis and polynomial planning of image contour transformations. Both simulations and real-time experiments with a robotic eye-in-hand configuration are discussed, validating the approach in terms of robustness and applicability to visual navigation, active exploration and perception, and human-robot interaction.**

Index Terms—**Active and real-time vision, task specification and planning, vision-guided robotics.**

I. Introduction

A combination of sensory processes and motor control actions is required for a robot vision system to execute its assigned task. The simplest visual tasks can be regarded as reactive transformations from perception to action, where motor actions are reflexes to visual data [1]. One step of complexity above reactive tasks are active tasks, which use an *a priori* knowledge and representation of the visual environment for the purposive planning of visuo-motor strategies [2], [3]. Active visual tasks usually rely on the underlying execution of lower level reactive tasks to provide the necessary amount of knowledge about the visual environment. Consider for example the task of catching a moving object [4]: reactive tracking of object motion is required during the purposive positioning of the grasping tool w.r.t. the object. The problem of the integration of multiple tasks is of key importance for the design of active vision systems and for more general robotic systems as well [5]. A general framework for the integration of reactive visual processes where the problem of the hierarchical organization of control processes is addressed, was presented recently in [6].

In the last two decades, much work has been done on the design of architectures for closed loop vision based robot control, or *visual servoing* [7]–[9]. For both technological and theoretical reasons, in nearly all existing visual servoing systems camera motion commands are issued to a high sampling rate robot position controller [10]. A modern approach to visual servoing is to close the visual loop at the bidimensional (2-D) image level instead of at the 3-D work-space level, so as to reduce the system sensitivity to modeling inaccuracies and estimation uncertainties [11]. Which kind of image features—points, curves, regions, etc.—to use for visual servoing is discussed in [12] from both the computational and robustness points of view. The closure of the control loop at the image level is made by matching visual feature appearance against a target one: such a visual error is conveniently mapped into camera motion commands according to a linear 2-D/3-D transformation [13]. Besides feedback, the visual error signal can be based on feedforward information derived from the specification of the task at hand. Although the issue of building visual servoing systems based on image feedback has been addressed by many authors, the problem of how to generate and use feedforward information in visual servoing has not received the same attention. Yet, exploiting feedforward can significantly improve system performance.

In this paper, we present an approach to the design, control and composition of elementary active and reactive visual tasks. Such an approach has evolved from an earlier theoretical framework called affine visual servoing (AVS). One of the distinguishing features of AVS is the combination of differential control with an affine model of camera-object interaction for the design of relative positioning tasks of a robot w.r.t. fixed and planar objects by means of visual information. AVS steers the 2-D visual appearance of the objects via feedforward and feedback sensori-motor strategies involving object fixation during task execution [14]. The approach presented here, referred to as Extended Affine Visual Servoing (EAVS), improves AVS in several directions. First, in order to enlarge the set of application scenarios of the system, the fixation constraint is removed and objects are allowed to move in the environment; the resulting system architecture has applications in landmark-based visual navigation, active exploration and perception, and human–robot interaction. Second, the intrinsic representation ambiguities and task singularities which arise due to the affine model of camera-object interaction are pointed out, and methods to cope with them while keeping the advantages of linearizing interaction are devised. Third, a mechanism for task layering is expounded, enabling to stack elementary tasks one above the other hierarchically for the design of complex tasks. Fourth, the system has been implemented and experimented using a manipulator-mounted camera and active contours as image primitives. The results obtained demonstrate that the system is robust w.r.t. modeling uncertainties and difficult operating conditions.

The paper is organized as follows. Section II discusses the main theoretical (modeling and control) issues of the EAVS approach. Section III illustrates the implementation and combination of three different visual tasks. In Section IV they are shown results of experiments with an eye-in-hand robotic setup. Finally, in Section V the major contribution of the work is summarized and future work is outlined.

II. Formulation

A. Overview

Fig. 1 illustrates the main functional blocks used in the EAVS approach for the control of elementary visual tasks. In the following, an overview of the approach is provided, and the role played by each block is expounded.

Visual Representation: Changes in viewpoint determined by relative motion of camera and object influence the raw image data according to the nature of both camera projection and object shape.

Manuscript received December 22, 1995; revised January 7, 1997 and June 21, 1998.

C. Colombo is with the Dipartimento di Elettronica per l'Automazione, University of Brescia, I-25123 Brescia, Italy (e-mail: columbus@bsing.ing.unibs.it).

B. Allotta is with Scuola Superiore Sant'Anna, I-56127 Pisa, Italy (e-mail: ben@mousebuster.sssup.it).

Publisher Item Identifier S 1083-4427(99)00240-4.

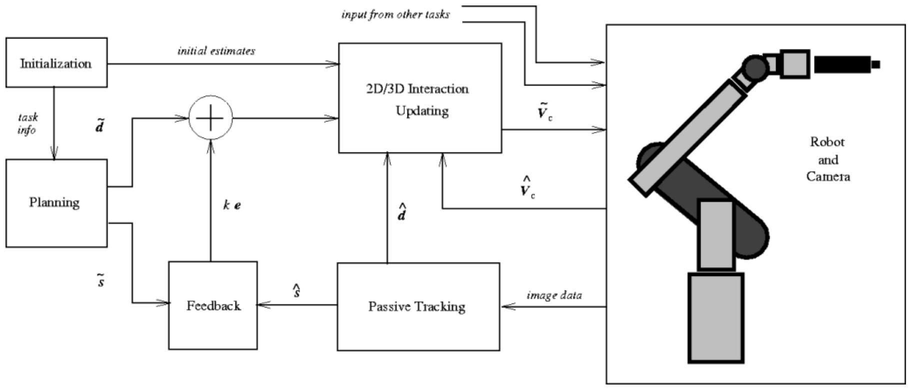


Fig. 1. Control scheme for an elementary visual task.

Given a model of camera-object interaction, a *visual representation* $\{\boldsymbol{s}, \boldsymbol{d}\}$ can be defined where, at each time t, $\boldsymbol{s}(t)$ is an m-dimensional parameterization of the object's visual appearance, and $\boldsymbol{d}(t)$ is a set of n image differential parameters which describe 2-D changes of image appearance caused by the 3-D relative velocity twist of camera and object.

Passive Tracking: The visual representation is estimated as $\{\hat{\boldsymbol{s}}, \hat{\boldsymbol{d}}\}$ through visual analysis, according to a tracking process in the image plane, referred to as *passive tracking*. Such a process, which takes place independently of camera motion, reproduces the covert attentional mechanism allowing humans to reallocate visual computational resources without changes of viewpoint [15].

Camera-Object Interaction: Camera-object interaction is expressed in terms of the $n \times 6$ Jacobian $\mathcal{L}$ encoding the differential transformation from relative twist $\Delta\boldsymbol{V} = \boldsymbol{V}_c - \boldsymbol{V}_o$ to appearance changes (interaction matrix):

$$\boldsymbol{d} = \mathcal{L}\,\Delta\boldsymbol{V}, \tag{1}$$

where $\boldsymbol{V}_c = [\boldsymbol{T}_c^T\ \ \boldsymbol{\Omega}_c^T]^T$ is the velocity twist of the camera and $\boldsymbol{V}_o = [\boldsymbol{T}_o^T\ \ \boldsymbol{\Omega}_o^T]^T$ is the velocity twist of the object.

Task Specification and Planning: Any purposive (or active) task is described as a target evolution $\tilde{\boldsymbol{s}}(t)$ of object image appearance. Such a task is specified in differential terms by a trajectory $\tilde{\boldsymbol{d}}(t)$ and suitable endpoint conditions. Reactive tasks lack a planned trajectory and are thus regarded as a particular class of active tasks, in which control always results in a pure regulation-to-zero scheme.

Control of Visual Tasks: Once the structure of $\mathcal{L}$ is identified, the following strategy is used for task control:

$$\tilde{\boldsymbol{V}}_c = \hat{\boldsymbol{V}}_o + \mathcal{L}^+[\tilde{\boldsymbol{d}} + k\ \boldsymbol{e}(\hat{\boldsymbol{s}}, \tilde{\boldsymbol{s}})], \tag{2}$$

where $\tilde{\boldsymbol{V}}_c$ is the required motion of the camera, $\hat{\boldsymbol{V}}_o$ is an estimate of object motion, $\boldsymbol{e}(\hat{\boldsymbol{s}}, \tilde{\boldsymbol{s}})$ is an n-dimensional error signal derived from a suitable comparison between the estimated object appearance and the target one, $k \in [0, 1]$ is the feedback gain, and $\mathcal{L}^+$ is the $6 \times n$ pseudo-inverse of $\mathcal{L}$. Control law (2) ensures a zero steady-state error for a constant object motion. If k is properly tuned, position feedback compensates for various modeling inaccuracies (manipulator kinematics, camera-object interaction model, camera parameters, finite differences approximation, etc.). Specifically, thanks to the feedback loop closure at the image level rather than in space, control converges even when the camera is poorly calibrated: indeed, calibration affects more speed of convergence than stability (see e.g. [16]). The anticipation term $\hat{\boldsymbol{V}}_o$ is evaluated from (1) as $\hat{\boldsymbol{V}}_o = \hat{\boldsymbol{V}}_c - \mathcal{L}^+\hat{\boldsymbol{d}}$, where the camera motion estimate $\hat{\boldsymbol{V}}_c$ is simply obtained from joint position sensor data.

3-D Estimation and Updating: The interaction matrix has time-dependent entries related to the current camera-object relative geometry; these entries are estimated at startup and updated at run-time. As for the camera parameters, the estimation accuracy of the entries of the interaction matrix is not critical, due to the presence of feedback in the control.

Composition of Visual Tasks: Since in the EAVS approach several tasks can be executed in parallel, there is a danger of tasks issuing conflicting commands to hardware and computing resources. Such conflicts can be resolved by organizing the tasks into a hierarchy based on the processing time (or bandwidth) of the control processes, i.e., on the feedback gain of each task. With such techniques, slower tasks, working in more abstract reference spaces, provide the reference signal to lower level tasks. A similar mechanism takes place, at a different control bandwidth scale, in the layering of proprioceptive and exteroceptive tasks [17]: the former, usually simple PID's, are considered as virtually instantaneous w.r.t. the latter in terms of loop time.

B. A Compact Visual Representation

So far, nothing has been said about choosing an effective visual representation for task control. This is a key point in the design of a real time functioning visual task. A regulation-to-zero (no planning) scheme for camera motion control based on the interaction matrix concept is introduced in [11], where $\boldsymbol{d} = \dot{\boldsymbol{s}}$, i.e. $n = m$. As the size of $\mathcal{L}$ $(\propto n)$ is directly related to the number $(\propto m)$ of visual features—mainly points and lines—used to represent an object, the use of such a scheme is limited to rather simple object shapes. However, thanks to a careful modeling of the interaction, a compact visual representation and task specification can be obtained where control complexity is decoupled from shape complexity, i.e., where n is independent of m. We derive herein such a representation, and show in the next Section how to use it in the basic control scheme, thus augmenting regulation with a suitable planning strategy based on contour features and obtaining beneficial effects on loop time, stability, and task complexity.

Let us refer to a pinhole camera with fixed focal length f and optical axis Z. Let $Z(X, Y)$ denote the object's visible surface in camera-centered coordinates, and $\boldsymbol{x} = [x\ \ y]^T$ be the perspective projection of a visible point $\boldsymbol{P} = [X\ Y\ Z(X, Y)]^T$ onto the image plane:

$$\boldsymbol{x} = f/Z(X, Y) \cdot [X\ Y]^T. \tag{3}$$

The differential interaction between camera and object can be expressed, at a generic image point, in terms of the 2-D motion field $\boldsymbol{v}(x, y) = \dot{\boldsymbol{x}}$ arising in the image plane due to both surface shape and 3-D relative speed. The motion field evaluates as $\boldsymbol{v}(x, y) =$

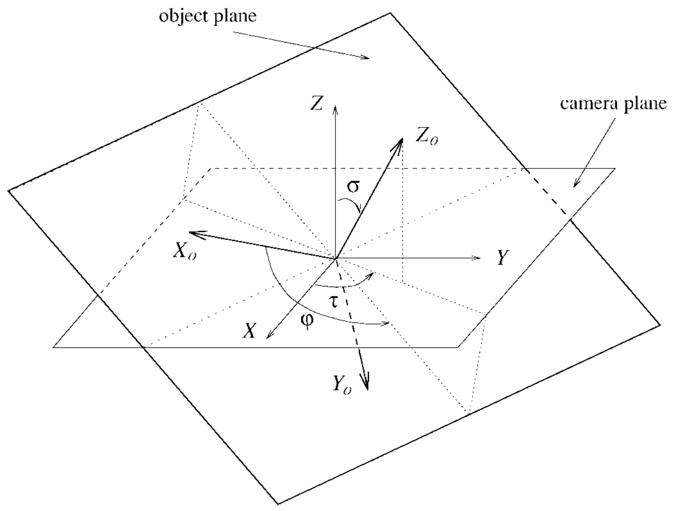


Fig. 2. Camera-object relative orientation parameters.

$\mathcal{F}(x,y)\,\Delta\boldsymbol{V}$, where the 2×6 *motion field matrix,* in (4) shown at the bottom of the page, is obtained by expressing $\dot{\boldsymbol{P}}$ in terms of relative twist and position and using (3). The dependency of the motion field matrix on image coordinates is obtained, for a given image projection model, from assumptions on the shape of the visible surface [18]. Using a perspective camera, the motion field matrix associated to a planar surface

$$Z(X,Y) = p\,X + q\,Y + c \tag{5}$$

is a quadratic function of image coordinates. Indeed, by combining (3) and (5), it is easy to show that $z^{-1}(x,y)$ s.t. $z(fX/Z, fY/Z) = Z$ is a linear function of image coordinates expressing the reciprocal of object depth directly in terms of plane coefficients. Let us assume now that $\sqrt{x^2+y^2} \ll f$ at any image projected point $\boldsymbol{x}$. Such a condition is met either in narrow field of view cameras, or—with wider fields of view—if transverse object dimensions are small w.r.t. its depth, i.e., $\sqrt{X^2+Y^2} \le Z/5$. This assumption allows us to neglect quadratic terms in (4), and obtain a new motion field matrix

$$\mathcal{V}(x,y) = \begin{bmatrix} -f/z(x,y) & 0 & x/c & 0 & -f & y \\ 0 & -f/z(x,y) & y/c & f & 0 & -x \end{bmatrix} \tag{6}$$

which, as far as interaction with planar surfaces is concerned, is a *linear function* of image coordinates. The interaction linearization is also extended to nonplanar surfaces, provided that the surface has one principal inertia moment at least ten times the other two, thus allowing to define an approximating plane, referred to as *object plane*. As shown in Fig. 2, the object plane has equation $Z_o = 0$ in an object-centered frame $\{X_o, Y_o, Z_o\}$ fixed to the visible surface centroid $[X_B\ Y_B\ Z_B]^T$ and having its Z_o-axis aligned with the major inertia axis.

Thanks to the linearization above, the 2-D velocity at any image point is linear w.r.t. image position. Hence, the dynamic evolution of any image patch enclosing the object has six degrees of freedom, three describing rigid motion (translation and rotation) of the whole patch, and three describing patch shape changes (isotropic and nonisotropic deformations). The parameters, time-dependent and spatially constant in the image, are the components of a translation 2-vector $\boldsymbol{v}_B = \boldsymbol{v}(x_B, y_B)$—the velocity of the centroid's projection $\boldsymbol{x}_B$—and a motion parallax vector $\boldsymbol{w} = [\mathrm{div}\ \mathrm{curl}\ \mathrm{def1}\ \mathrm{def2}]^T$ taking into account field deformations [14]. As for the motion field, we relate the motion parallax to the camera relative twist through a 4×6 *motion parallax matrix* $\mathcal{W}$ s.t. $\boldsymbol{w} = \mathcal{W}\,\Delta\boldsymbol{V}$, which evaluates as

$$\mathcal{W} = \begin{bmatrix} p/c & q/c & 2/c & 0 & 0 & 0 \\ -q/c & p/c & 0 & 0 & 0 & -2 \\ p/c & -q/c & 0 & 0 & 0 & 0 \\ q/c & p/c & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{7}$$

The motion field matrix $\mathcal{V}_B = \mathcal{V}(x_B, y_B)$, the motion parallax matrix $\mathcal{W}$ and the square 6×6 matrix $\mathcal{U}$ s.t.

$$\boldsymbol{u}_B = [\boldsymbol{v}_B^T\ \boldsymbol{w}^T]^T = \mathcal{U}\ \Delta\boldsymbol{V} \tag{8}$$

are used in the next Section as interaction matrices for the design of visual tasks. When $\mathcal{U}$ is used, a one-one correspondence is established between the six degrees of freedom describing the appearance evolution and those of relative camera-object motion. The linearization above allows a compact (minimal) representation to be used for differential interaction, this being evident from the small dimensions of $\mathcal{V}_B$ $(n=2)$ and $\mathcal{W}$ $(n=4)$. Moreover, as n is clearly independent of m, *control complexity is decoupled from shape complexity*; the same could not be achieved with a full perspective, non linear interaction model such as the one used in [11].

III. Design of Visual Tasks

The interaction matrices defined above have been used in the design and implementation of three different visual tasks; their characteristics are summarized in Table I. The first two tasks are reactive, while the third one, which also contains a planning strategy, is active. The tasks are introduced in order of complexity, this being related to computational burden and bandwidth, initial conditions and type of object representation required.

A. Task Specification

Fixation Pursuit: The simplest task is a fixation pursuit, i.e., the tracking of an object point. Fixation is one of the most basic tasks in the human visual system, allowing one to concentrate the major part of visual resources in the processing of foveal data, and to keep the high resolution fovea centered on the object of interest, while using the image periphery for attentive processing and monitoring [19]. The importance of doing fixation in active vision systems has already been emphasized in several recent works [20], [21]. The object centroid is chosen as a fixation point, whose 3-D tracking can be accomplished, as a result of our linearized interaction model, through a simple 2-D tracking of the imaged object centroid. Indeed, due to linear approximation, the latter coincides with the projection of the 3-D visible surface centroid. From (6) it is apparent that, under fixation, a constraint is determined among the translational and rotational relative speed components parallel to the image plane: $c\cdot[\Delta\Omega_x\ \Delta\Omega_y]^T = [\Delta T_y\ -\Delta T_x]^T$. As a result, to all tasks which "run slower" than fixation, the number of independent camera degrees of freedom is reduced from six to four, thereby further simplifying the interaction matrices.

$$\mathcal{F}(x,y) = \begin{bmatrix} -f/Z & 0 & x/Z & xy/f & -(f+x^2/f & y \\ 0 & -f/Z & y/Z & (f+y^2/f) & -xy/f & -x \end{bmatrix} \tag{4}$$

TABLE I
SPECIFICATION OF THE THREE TASKS

TASK SYNOPSIS	*Fixation Pursuit*	*Reactive Tracking*	*Active Positioning*
Initial Conditions	$\{\boldsymbol{x}_i^0\}$	$\{\boldsymbol{x}_i^0\}$	$\{\boldsymbol{x}_i^0\}, \{\boldsymbol{x}_i^{\mathrm{T}}\}$
Visual Representation	$\{\boldsymbol{x}_{\mathrm{B}}, \boldsymbol{v}_{\mathrm{B}}\}$	$\{\{\boldsymbol{x}_i\}, \boldsymbol{u}_{\mathrm{B}}\}$	$\{\{\boldsymbol{x}_i\}, \boldsymbol{u}_{\mathrm{B}}\}$
Task Description	$\tilde{\boldsymbol{x}}_{\mathrm{B}} = \mathbf{0}$	$\{\tilde{\boldsymbol{x}}_i(t)\} = \{\boldsymbol{x}_i^0\}$	$\{\tilde{\boldsymbol{x}}_i(T)\} = \{\boldsymbol{x}_i^{\mathrm{T}}\}$
Interaction Matrix	$\mathcal{V}_{\mathrm{B}}$	$\mathcal{U}$	$\mathcal{U}$
Task Type	reactive	reactive	active

Reactive Tracking: With the term reactive tracking we refer to a visual task in which an object is tracked by active movements of the camera instead than by the pseudo-attentive movements used in passive tracking. The goal of reactive tracking is *to reproduce the motion of an object in the visual environment*. This kind of task can prove useful in the design of human-robot interfaces (mimicking human gestures has many applications for instance in teleoperation scenarios), and also provide—by simply reading from encoders the 3-D camera velocity—an estimate of 3-D object motion. A fixation point also exists for this task—indeed, the centroid's speed in constrained to be zero. However, this is generally a point of the visual periphery, since the centroid's initial position for this task is not necessarily the image origin. Hence, during reactive tracking the direction of gaze does not necessarily coincide with the direction of attention, and the system is able to react to world changes with the "corner of the eye."

Active Positioning: The active positioning task consists of purposively changing the relative spatial configuration (pose, distance) of the camera w.r.t. a still or moving object. As such, active positioning is extremely important for the optimal execution of complex perceptive and explorative tasks in space, service and industrial robotics as well. The view transformation is driven by the comparison between the initial object appearance and a target object view. It must be present in a long-term memory of the system and selected at startup. On the basis of the above interpretation of reactive tracking in terms of covert attentional shifts, we can relate the movements of the object's visual appearance in the image to corresponding "attentional movements" from one region to another of image periphery. Better still, reactive tracking itself can be regarded as a particular case of active positioning, where the final appearance of the target coincides with its initial appearance, thereby simply requiring the extraction of the target object view directly at startup from image data and not from memory.

B. Implementation

Passive Tracking and Feedback: To estimate the current visual representation of the object (visual appearance and differential parameters) we use quadratic B-spline active contours [22]. These exploit a Kalman filter to robustly track affine deformations of a template contour, and allow to compactly represent object shape—small values of m for a fixed shape complexity—in terms of a set $\{\boldsymbol{x}_i\}$ of M control points and associated spline basis functions $\{f_i\}$: $\boldsymbol{x}(s) = \Sigma_{i=1}^M f_i(s)\, \boldsymbol{x}_i, s \in [0,1]$. Active contours are used to estimate the affine transformations related to deformations of visual appearance. The six parameters of the affine transformation $\hat{\boldsymbol{u}}_B$ between two successive contour estimates, $\{\hat{\boldsymbol{x}}_i(t)\}$ and $\{\hat{\boldsymbol{x}}_i(t+1)\}$, are obtained via least squares. The feedback error (centroid, shape) is evaluated analogously, from the least squares matching of the target visual appearance, $\{\tilde{\boldsymbol{x}}_i\}$, against $\{\hat{\boldsymbol{x}}_i\}$, the estimated appearance. To enhance the quality of all visual measurements (visual representation, object motion), simple IIR filters are used.

Initializing and Updating the Interaction Matrices: The interaction matrix embeds, in the object plane coefficients p, q and c, information on 3-D relative camera-object pose and translation. The initial and run-time value of these parameters must be known, at least approximately, to improve the convergence of the control scheme. As shown in Fig. 2, relative pose is uniquely determined by the three angles $\sigma \in [0, \pi/2]$ (slant), $\tau \in [-\pi, \pi]$ (tilt) and $\varphi \in [-\pi, \pi]$ (azimuth), to which plane parameters are related as $p = -\tan\sigma\cos\tau, q = -\tan\sigma\sin\tau, c = Z_B\cdot(1 - px_B/f - qy_B/f)$.

A raw estimate of initial object pose and distance is obtained by assuming a scaled orthography model of camera projection [23], i.e. a linearization of (3) holding accurate for $px_B/f + qy_B/f$:

$$\begin{bmatrix} x - x_B \\ y - y_B \end{bmatrix} = \begin{bmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \end{bmatrix} \begin{bmatrix} X_o \\ Y_o \end{bmatrix}. \tag{9}$$

An estimate of $\{t_{ij}\}$ is easily obtained from the least squares comparison of the current appearance of the object and an *a priori* model stored into memory, e.g. the frontoparallel view of the object at unit distance and scale. Once this is known, pose (p, q) and scale c are computed from the triplet (τ, σ, φ) solution of the non linear system

$$\begin{aligned} t_{11} + t_{22} &= \lambda\ \cos(\tau - \varphi)\ (\cos\sigma + 1) \\ t_{21} - t_{12} &= \lambda\ \sin(\tau - \varphi)\ (\cos\sigma + 1) \\ t_{11} - t_{22} &= \lambda\ \cos(\tau + \varphi)\ (\cos\sigma - 1) \\ t_{21} + t_{12} &= \lambda\ \sin(\tau + \varphi)\ (\cos\sigma - 1) \end{aligned} \tag{10}$$

where $\lambda = f/z(x_B, y_B)$ is an isotropic image scaling factor inversely proportional to object depth. Notice that all perspective linearizations introduce a pose ambiguity, i.e., two dual solutions exist to (10), as two distinct object poses—(τ, σ, φ) and $(\tau + \pi, \sigma, \varphi + \pi)$ for scaled orthography—share the same visual appearance. To disambiguate the pose, one has to refer back to the full perspective model, and choose as the "true" pose the one providing the best least squares fit against image data [24]. Thus far parameter initialization was addressed. At run-time, pose and distance parameters are obtained by combining the current estimates—derived from (1) using the differential measurements $\hat{\boldsymbol{u}}_B$ and $\widehat{\Delta \boldsymbol{V}}$—and their predicted value—obtained using finite differences and expressing $\dot{p}, \dot{q}, \dot{c}$, as functions of p, q, c, and $\Delta\boldsymbol{V}$ as expounded in [14].

Planning and Control: A planning strategy is used to produce a viewpoint shift (pose, translation). Such a shift is associated with an accordingly smooth change of the object's visual appearance of duration T from an initial contour $\{\boldsymbol{x}_i^0\}$ to a final desired contour $\{\boldsymbol{x}_i^T\}$. The mapping "in the large" between these contours is evidently affine

$$\boldsymbol{x}_i^T - \boldsymbol{x}_B^T = \mathcal{A}_T(\boldsymbol{x}_i^0 - \boldsymbol{x}_B^0) \tag{11}$$

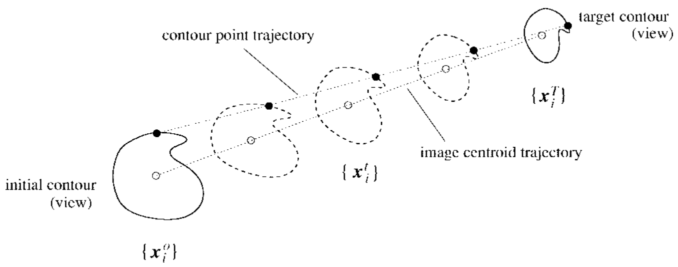
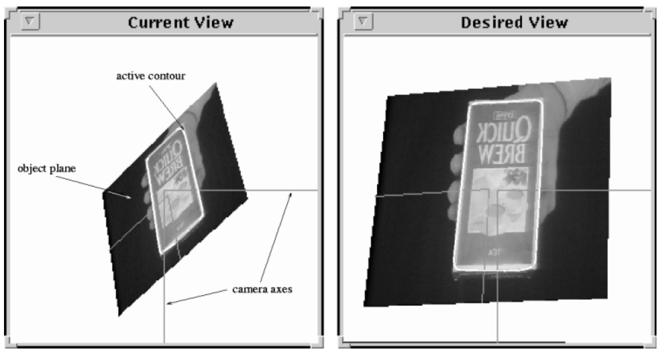


Fig. 3. Image contour planning.

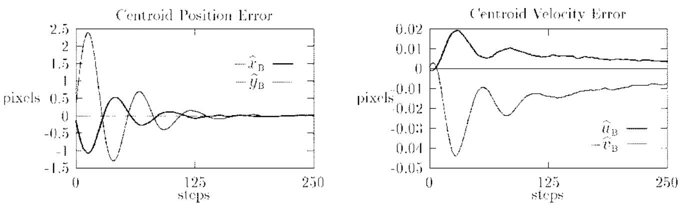


(a) (b)

Fig. 4. Simulation experiment (a) initial object view and (b) target configuration for active positioning.

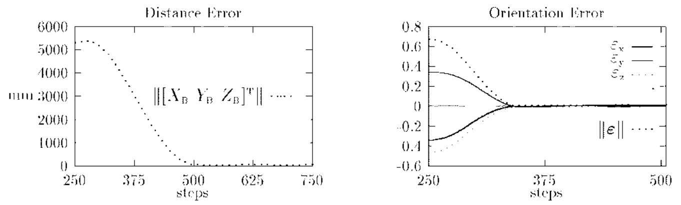
as the result of a sequence of affine transformations "in the small." The automatic generation of the target view is trivial if a CAD model of the object at hand is available. Should it not be the case, the target view can be obtained in a simple way from the knowledge of 1) initial relative pose, 2) relative pose in the final (desired) configuration, and 3) initial view. The reference contour evolution is planned according to a trajectory for each of the control points, which is polynomial (degree $h \geq 1$) in time and linear in the image space so that each contour point follows a linear trajectory through the image, with a specific speed profile (Fig. 3). The polynomial constants must be determined based on boundary conditions. A basic condition is that the contour evolution starts with the initial contour and terminates with the desired contour. Additional constraints on the polynomial derivatives can be imposed at the trajectory endpoints with $h \geq 3$, thus providing beneficial effects on contour tracking, visual analysis and camera velocities and accelerations at the expense of a slower rate of convergence. Smooth trajectories are obtained with cubic $(h = 3)$ or quintic $(h = 5)$ polynomials by imposing zero endpoint velocity and acceleration.

The smooth pose shift produced by the planning strategy can be represented as a curvilinear path on a semisphere of all possible relative orientations between camera and object plane. As it is, planning always produces, out of the two dual poses $\boldsymbol{Q}$ and $\boldsymbol{Q}'$ sharing the same goal appearance under linearized interaction, the one which is closer to the initial pose moving along a geodesic path on the semisphere—let us say $\boldsymbol{Q}$. In order to reach the farthest pose $\boldsymbol{Q}'$, it is required to split the path $\boldsymbol{P} \mapsto \boldsymbol{Q}'$ in two parts, $\boldsymbol{P} \mapsto \boldsymbol{O}$ and $\boldsymbol{O} \mapsto \boldsymbol{Q}'$, and to pass through a suitably scaled frontoparallel view $\boldsymbol{O}$. Note that the frontoparallel pose corresponds to the sole control algorithmic singularity, in that the determinant of $\mathcal{W}$ vanishes identically for $p = q = 0$. In order to avoid ill-conditioning and arbitrarily large desired camera speeds around the frontoparallel configuration, the control loop is opened whenever the slant angle $\sigma = \arctan(\sqrt{p^2 + q^2})$ is estimated to be smaller than five degrees.



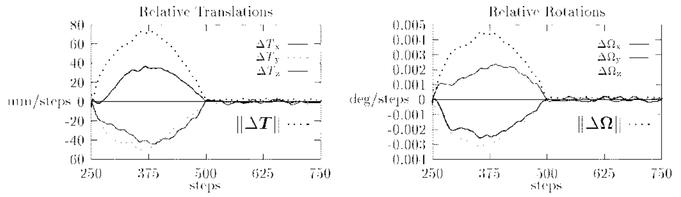
(a) (b)

Fig. 5. Fixation pursuit (steps 0–250). (a) Image centroid position error and (b) image centroid velocity error.



(a) (b)

Fig. 6. Active positioning (steps 250–500) and reactive tracking (steps 500–750) (a) distance error and (b) orientation error.



(a) (b)

Fig. 7. Camera-object relative speed during active and reactive tracking (a) translations and (b) rotations.

C. Task Composition

A simulation environment running under X Window was developed for testing elementary visual tasks and their composition. In Fig. 4, the left window shows the current view of the object plane (featuring a hand holding a box), together with the associated active contour, while the right window shows a target object appearance as required for active positioning. Simulations are carried out with a focal length $f = 18$ mm, and an object speed $\boldsymbol{T}_o = [1.8\ 3.6\ 5.4]^T$ mm/steps and $\boldsymbol{\Omega}_o = [0.01\ 0.02\ 0.03]^T$ °/steps. The initial configuration is $c(0) = 21\ 600$ mm, $\sigma(0) = 60°, \tau(0) = 30°, \varphi(0) = 150°$, while the target configuration has parameters $c(T) = 16\ 200$ mm, $\sigma(T) = 30°, \tau(T) = 60°, \varphi(T) = 150°$. Position measurements are smoothed using an IIR filter with gain $k_p^{\mathrm{IIR}} = 0.05$, while the IIR velocity filter gain is $k_v^{\mathrm{IIR}} = 0.005$.

Results for a sequence of 750 steps are shown in Figs. 5–7. In the first part of the sequence (steps 0–250), the fixation pursuit task is let to run alone, with a feedback gain set to $k^{\mathrm{fix}} = 0.2$ producing a slightly underdamped system behavior. Fig. 5(a) shows that after less than 250 steps the initial position error of Fig. 4(a) is completely recovered, due to the large associated feedback gain; the velocity error has a slower convergence instead [see Fig. 5(b)]. After step 250, while fixation pursuit continues running, an active positioning task is started attempting to change the object view as in Fig. 4(b) based on a cubic contour planning. This provides an example of task layering: as the active positioning feedback gain is less than the fixation gain by one order of magnitude $(k^{\mathrm{pos}} = 0.02)$, the system globally executes an *active positioning task w.r.t. a fixated moving object*. The positioning strategy thus consists in suitably commanding camera translation $\boldsymbol{T}_c$ and cyclotorsion Ω_z through the control of motion parallax, using a simplified motion field interaction matrix.

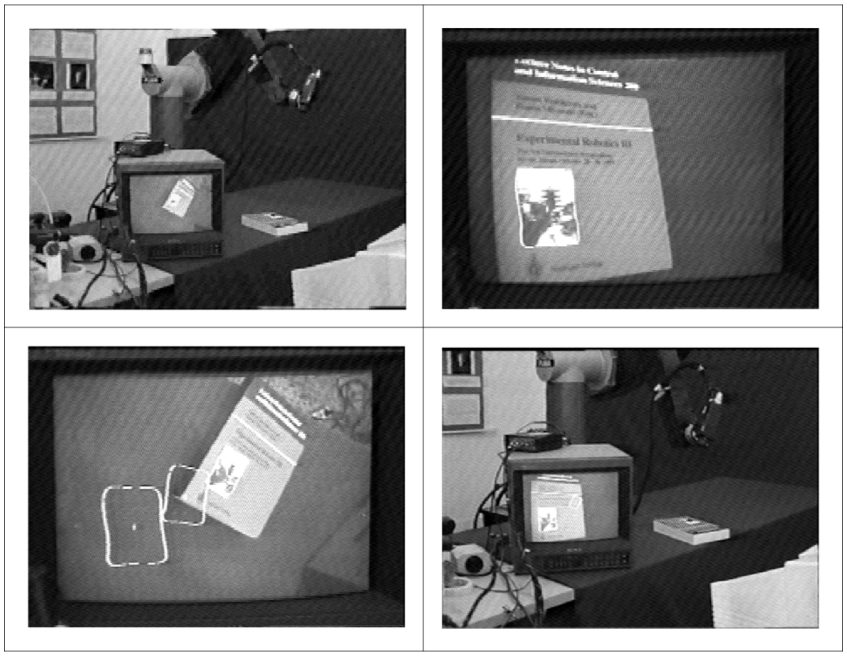


Fig. 8. An active positioning session. Workspace and image plane are shown. The monitor upon the table displays the current scene as seen by the camera.

Fig. 6 shows the 3-D relative orientation and distance error for the task, obtained from the comparison of current and target camera frames $\{\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k}\}$ and $\{\tilde{\boldsymbol{i}}, \tilde{\boldsymbol{j}}, \tilde{\boldsymbol{k}}\}$. An overall orientation error is also evaluated as $\varepsilon = \frac{1}{2}\,[\boldsymbol{i} \wedge \tilde{\boldsymbol{i}} + \boldsymbol{j} \wedge \tilde{\boldsymbol{j}} + \boldsymbol{k} \wedge \tilde{\boldsymbol{k}}]$ according to the formula introduced in [25]. The target configuration is computed with an error of within a few degrees (orientation angles) and millimeters (relative distance). It should be remarked that although the 3-D position and orientation errors are calculated during the simulation, they are not used for control purposes. The sole feedback used in the control scheme comes from 2-D image data. Fig. 7 shows the relative speed of camera and object. Due to the cubic-based planning and to a good tracking and filtering of the object's independent motion, the relative speed profile varies gracefully, with gradual relative accelerations and decelerations. Also notice that, since the fixation pursuit task is still under execution, relative translations and rotations are related according to the fixation constraint [see Section III-A]. When, after step 250, the goal of planning is reached, the active task degenerates into a reactive tracking task, thus ensuring (steps 500–750, see again Figs. 6 and 7) that the new relative configuration of camera and object be properly maintained.

IV. Experiments with a Robotic Platform

A. Setup and Operating Modes

Experiments were performed using an eye-in-hand robotic platform. The hardware setup consists of a PUMA 560 manipulator with MARK III controller equipped with a wrist-mounted camera and a PC equipped with a frame grabber and a graphic accelerator. The PC features a 80 486-66 MHz processor. Concerning the software, the MARK III controller runs VAL II programs and communicates with the PC via the ALTER real-time protocol using an RS-232 serial interface. The ALTER protocol allows us to modify the Cartesian setpoint of the robot arm every 28 ms. Due to the computational cost of tracking algorithms, a multirate real-time control has been implemented. New velocity setpoints are generated by the PC with a sampling rate $T_2 = N\ T_1$, where $T_1 = 28$ ms is the sampling rate of the ALTER protocol and N is an integer, depending on the number of control points of the B-spline contour. The computation time when using 10 control points is less than 100 ms, allowing sampling time T_2 to be equal to $4\ T_1$. A "fast" communication process maintains the handshake with the MARK III controller by letting the most recent velocity twist setpoints generated by the high level "slow" process be available in a mailbox. A typical interaction session with the system is illustrated in Fig. 8. The figure summarizes the execution of an active positioning task w.r.t. a planar object (a book upon a table). They are shown: the initial relative configuration (top left); the target image appearance (top right); initial and target contours, and an intermediate planned contour (bottom left), and (bottom right) the final configuration reached.

In order to assess the convergence and stability characteristics of the control scheme and tune up control parameters, the scheme for active positioning, which is the most complex of the three, is decomposed into its main "modes," which are tested separately. In increasing order of complexity, we have the modes:

Output Regulation: This mode, which can be used to properly set the control gains in order to have stable behaviors is relative to no planning (i.e., $\tilde{\boldsymbol{s}} = \boldsymbol{s}(T)$ *and* $\tilde{\boldsymbol{d}} = \boldsymbol{0}$); the system has to move simply from an initial configuration to the target one. In this case, the camera velocity commands derive from the 2-D position error between the target and the current contour configurations.

Servoing: In this mode, only the planned feature evolution $\tilde{\boldsymbol{s}}(t)$ is provided for feedback error estimation, while the feedforward speed command is kept to zero ($\tilde{\boldsymbol{d}} = \boldsymbol{0}$). This is used to assess the tracking performance of the control scheme, as the system is forced to compensate the feedback error $\boldsymbol{e}(\hat{\boldsymbol{s}}, \tilde{\boldsymbol{s}})$.

Planning: This is the normal mode, corresponding to the complete scheme. A considerable improvement w.r.t. servoing (no lag) is obtained by reintroducing the feedforward term $\tilde{\boldsymbol{d}}(t)$ in the control.

B. Results

Results on three different kinds of experiments are reported, the first two of which address robustness issues. The first experiment

TABLE II
ROBUSTNESS TO VARIATION OF OPERATING CONDITIONS

ORIENTATION ERROR	*Starting Slant* (degrees)					
Starting Distance (mm)	*10*	*20*	*30*	*40*	*50*	*60*
400	0.05132	0.04746	0.03759	0.05570	0.06342	0.09652
550	0.10385	0.07223	0.05352	0.04976	0.05498	0.09795
700	0.08820	0.03870	0.04443	0.03942	0.05788	0.10006
850	0.06157	0.01822	0.02794	0.03732	0.05965	0.11385
1000	0.05930	0.02494	0.03242	0.04781	0.06572	0.09992
1150	0.06929	0.03537	0.03327	0.07276	0.06987	0.11009

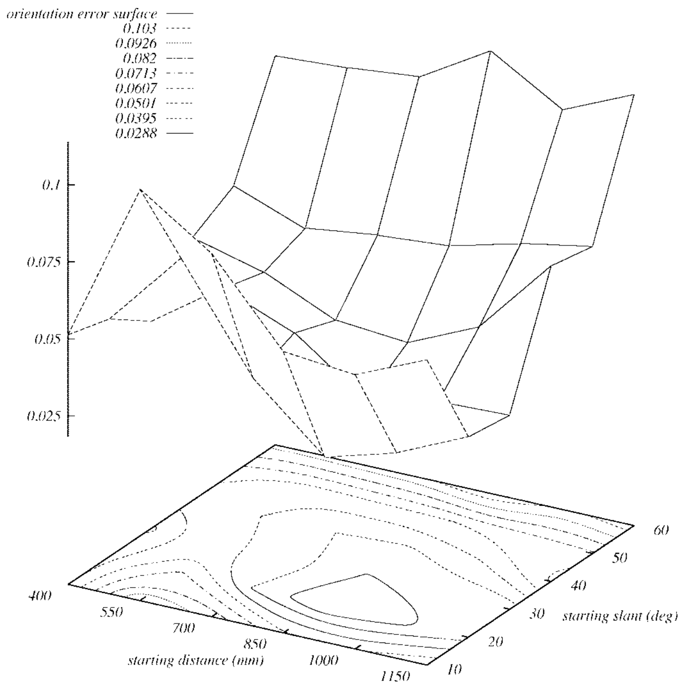


Fig. 9. Orientation error surface.

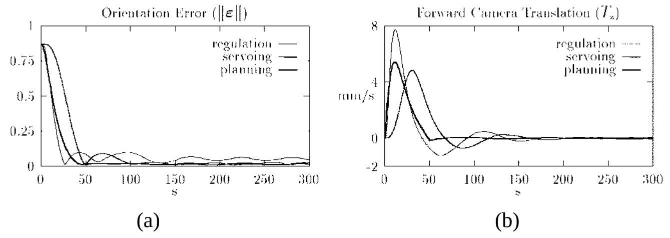


Fig. 10. Comparison among regulation, servoing, and planning modes (a) orientation error and (b) forward camera translational velocity.

is devoted to test the performance degradation as a function of operating conditions. The second experiment features the comparison among the three different operating modes and accounts for the relative robustness of such schemes. The third experiments provides a further insight into system behavior in the execution of an active positioning task.

Robustness of Modeling and Control: To test robustness in a wide range of operating conditions, the system is run in planning mode on 36 different active positioning tasks. All tasks involved use the eye-in-hand system of Fig. 8 to approach the object upon the table starting from different initial camera-object distances and slant angles. The target configuration was generated with the rule

$$\text{target distance} = 0.75 \cdot \text{initial distance}$$

$$\text{target slant} = 20^\circ + \text{initial slant}$$

to ensure a constant overall position change in each trial. Table II reports on positioning accuracy measured in terms of the overall orientation error defined in Section III-C between ground truth data and the actually reached relative configuration. This error varies between approximately 0.01 and 0.1, thus meaning an angular error between 0.6° and 6°. Concerning depth errors, these are by far less critical than orientation errors, due to the fact that the orthographic scaling model used embeds depth more accurately than orientation. To better understand how distance and slant affect system behavior, a 3-D orientation error surface was created, with accuracy data plotted against initial conditions (Fig. 9). The figure shows a plateau, located approximately in the middle of the plot, where the orientation error reaches a minimum; the error is maximum for large initial slants, and it is also quite large for small slant values, especially at intermediate distances. The performance variations related to the slant are easily interpreted by recalling that the slant value affects both (through p and q, see Section III) the conditioning number of the interaction matrix used for control, and (through X, Y, and Z, see Section III) the condition of validity of the interaction model itself. As the slant increases, control conditioning improves but interaction becomes nonlinear. On the other hand, the linearity condition gets better with increasing distances, but at the expense of a degradation of run-time image estimates $\hat{\boldsymbol{u}}_B(t)$—see Section III—due to a decrease in image resolution. The results show that system behavior is quite stable w.r.t. operating conditions, with errors which are typically a few percent of the overall position change.

Stability and Scheme Comparison: A comparison among the three operating modes presented in Section IV-A is illustrated in Fig. 10. Fig. 10(a) shows the time dependence of the orientation error and Fig. 10(b) shows the component T_z of the camera translation velocity for a relative positioning task. The task to fulfill is to reach a target orientation and distance of the robot camera w.r.t. a still object starting from a given initial relative pose while maintaining always fixation on the object centroid. The initial slant angle and distance are about 10° and 1.15 m, respectively, while the final values are 70° and 0.4 m. The trajectory between initial and final configurations is planned as described in Section III-B using quintic polynomials. The duration time for the servoing and planning mode was 50 s. The same feedback and image data filtering gains, experimentally

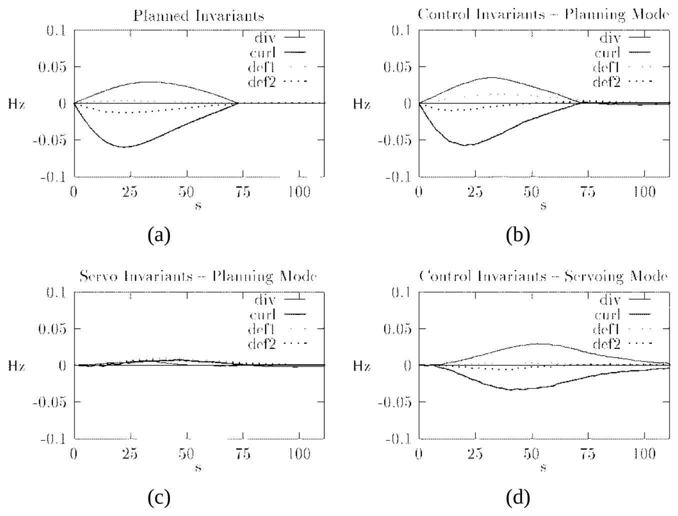


Fig. 11. Planned, control, and servo invariants for the servoing and planning modes.

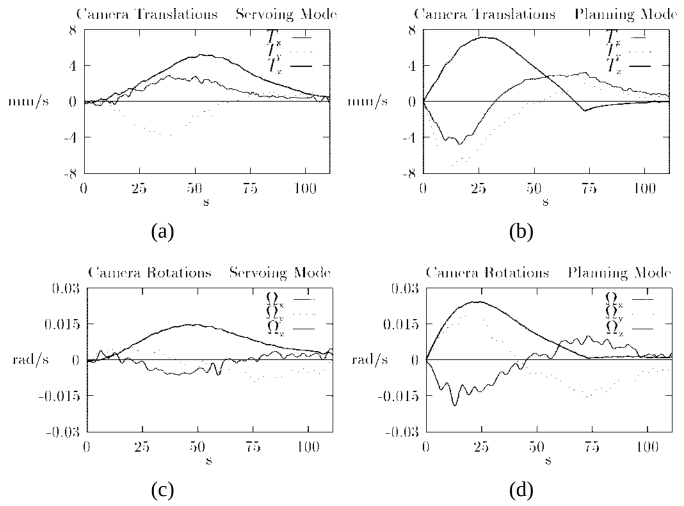


Fig. 12. (a)–(b) Camera translational and (c–d) rotational velocities. (a) and (c) servoing mode and (b) and (d) planning mode.

tuned, were used in the three cases. The tuning was made so as to obtain a slightly underdamped behavior, as it appears from the slowly vanishing oscillations particularly evident in the case of pure regulation.

The regulation scheme, which exhibits a behavior closely similar to the one described in [11], is as fast as the planning scheme but shows a slower convergence due to underdamping and model inaccuracies. Gain undertuning would be required in this case to obtain an acceptable damping. However, due to the limited number of bits used to represent robot commands, the reduction of feedback gains can often result in no motion of the robot thus yielding considerable errors. The servoing scheme, which does not use feedforward information, shows worse performance compared to the other two schemes in the transient. However, it excites smaller oscillations of the system, thus reaching an acceptable error faster than pure regulation. The planning scheme shows the best orientation error performance of all the three schemes, and achieves a fast and smooth completion of the task. Such a good performance requires from the robot actuators velocity resources similar to those required by the servoing mode and much lower than those of the regulation mode.

An Active Positioning Experiment: To gain an insight into the two basic system operating modes—servoing and planning—, the results for the positioning task of Fig. 8 are reported. Fig. 11(a) shows the planned differential evolution of image shape $(\tilde{\boldsymbol{d}})$: the task here is to displace the camera so as to reach an optimal configuration for reading the book's cover, so that the system is required to approach the object (positive divergence), rotate clockwise around the optical axis (negative curl), and slightly decrease the slant toward a frontoparallel interaction with the object (nonzero divergence terms).

In the tests, the gains of the filters as well as those of the feedback control term are tuned experimentally and are the same in the two experiments. To further stress the stability characteristics of the systems, the interaction matrix $\mathcal{L}$ is initialized at startup but not updated at run-time. Fig. 11 also shows the differential invariants generated by feedback (servo invariants, $k\ \boldsymbol{e}$) and the control invariants $\tilde{\boldsymbol{d}} + k\ \boldsymbol{e}$ used to produce the camera Cartesian velocity twist (of course, control and servo invariants are identical in servoing mode).

It is noted that the main effect of introducing the feedforward term $\tilde{\boldsymbol{d}}$ in the control is to significantly alleviate the job of the feedback and reduce the tracking lag. Yet, as apparent from Fig. 12 showing the translational and angular velocities of the camera, planning mode is more sensitive than servoing mode to 3-D interaction data (the entries of $\mathcal{L}$). Indeed, as the feedforward term is dropped out after 75 s, only the z-components of translational and rotational velocity are almost zero, while the T_x, T_y, Ω_x, and Ω_y components of camera velocity twist are still above zero, thus meaning that the anticipation has left a position error to be recovered after planning.

V. Conclusion and Future Work

In this paper, an approach for the control and layering of visual tasks has been proposed. Simulation and experimental results demonstrate that exploiting changes in the visual appearance of objects allows an easy design of elementary visual tasks, which can then be hierarchically stacked one above the other to yield more complex tasks. System complexity and time efficiency are greatly enhanced by introducing linear models of camera-object interaction, in that both visual analysis and control are simplified by the resulting reduction of the size of visual representation. We have also suggested a procedure to deal with the intrinsic ambiguities and singularities which arise due to such a reduction of representation. The natural applications of the proposed architecture are in the fields of visual navigation, active exploration and perception, and human-robot interaction.

Future work will address two main issues. First, adopting suitable techniques to augment the degree of autonomy of the system. For instance, object recognition/location techniques could be used to avoid manual contour initialization, and adaptive tuning of system parameters could help avoiding to set up manually control gains. Second, improving the strategies for singularity treatment and ambiguity removal. Specifically, we plan using a perspective approximation to ensure that the frontoparallel configuration be not singular.

Acknowledgment

The authors would like to thank Prof. B. Espiau for useful comments on an earlier draft of this work, and G. Borlizzi for his help during the implementation. The anonymous referees' comments were also greatly appreciated.

References

[1] J. L. Crowley, J. M. Bedrune, M. Bekker, and M. Schneider, "Integration and control of reactive visual processes," in *Proc. 3rd Eur. Conf.*

Computer Vision ECCV'94, Stockholm, Sweden, J. O. Eklundh, Ed., 1994, pp. II:47–58.

[2] S. J. Dickinson, H. I. Christensen, J. Tsotsos, and G. Olofsson, "Active object recognition integrating attention and viewpoint control," in *Proc. 3rd Eur. Conf. Computer Vision ECCV'94,* Stockholm, Sweden, J. O. Eklundh, Ed., 1994, pp. II:3–14.

[3] R. Cipolla and N. J. Hollinghurst, "Visual robot guidance from uncalibrated stereo," in *Realtime Computer Vision,* C. M. Brown and D. Terzopoulos, Eds., 1994.

[4] P. K. Allen, A. Timcenko, B. Yoshimi, and P. Michelman, "Trajectory filtering and prediction for automated tracking and grasping of a moving object," in *Proc. 1992 IEEE Int. Conf. Robotics Automation ICRA'92,* Nice, France, 1992, pp. 1850–1856.

[5] R. Brooks, "A robust layered control system for a mobile robot," *IEEE J. Robot. Automat.,* pp. 14–23, 1986.

[6] J. L. Crowley and H. I. Christensen, *Vision as Process.* Berlin, Germany: Springer Verlag, 1994.

[7] N. P. Papanikolopoulos, P. K. 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. Robot. Automat.,* vol. 9, no. 1, pp. 14–35, 1993.

[8] K. Hashimoto, Ed., *Visual Servoing: Real-Time Control of Robot Manipulators Based on Visual Sensory Feedback.* Singapore: World Scientific, 1993.

[9] K. Hosoda and M. Asada, "Versatile visual servoing without knowledge of true jacobian," in *Proc. Int. Conf. Intelligent Robots Systems IROS'94,* 1994, pp. 186–193.

[10] A. C. Sanderson and L. E. Weiss, "Image-based visual servo control using relational graph error signals," *Proc. IEEE,* vol. 68, pp. 1074–1077, 1980.

[11] B. Espiau, F. Chaumette, and P. Rives, "A new approach to visual servoing in robotics," *IEEE Trans. Robot. Automat.,* vol. 8, pp. 313–326, June 1992.

[12] J. T. Feddema, C. S. G. Lee, and O. R. Mitchell, "Weighted selection of image features for resolved rate visual feedback control," *IEEE Trans. Robot. Automat.,* vol. 7, pp. 21–31, Feb. 1991.

[13] C. Samson, M. Le Borgne, and B. Espiau, *Robot Control: The Task-Function Approach.* Oxford, MA: Clarendon, 1991.

[14] C. Colombo, B. Allotta, and P. Dario, "Affine visual servoing: A framework for relative positioning with a robot," in *Proc. IEEE Int. Conf. Robotics Automation ICRA'95,* Nagoya, Japan, May 1995, pp. 464–471.

[15] M. I. Posner, "Orienting of attention," *Quart. J. Exper. Psychol.,* vol. 32, pp. 3–25, 1980.

[16] B. Espiau, "Effect of camera calibration errors on visual servoing in robotics," in *Lecture Notes Control Information Sciences, Experimental Robotics III, 3rd Int. Symp.,* Kyoto, Japan, 1993, pp. 182–192.

[17] J. S. Albus, *Brains, Behavior, and Robotics.* New York: BYTE Books, 1981.

[18] J. K. Aggarwal and N. Nandhakumar, "On the computation of motion from sequences of images—A review," *Proc. IEEE,* vol. 76, no. 8, pp. 917–935, 1988.

[19] A. L. Yarbus, *Eye Movements and Vision.* New York: Plenum, 1967.

[20] G. Sandini and M. Tistarelli, "Active tracking strategy for monocular depth inference over multiple frames," *IEEE Trans. Pattern Anal. Machine Intell.,* vol. 12, no. 1, pp. 13–27, 1990.

[21] M. J. Swain and M. A. Stricker, "Promising directions in active vision," *Int. J. Comput. Vis.,* vol. 11, no. 2, pp. 109–126, 1993.

[22] A. Blake, R. Curwen, and A. Zisserman, "A framework for spatiotemporal control in the tracking of visual contours," *Int. J. Comput. Vis.,* vol. 11, no. 2, pp. 127–145, 1993.

[23] J. L. Mundy and A. Zisserman, "Projective geometry for machine vision," in *Geometric Invariance in Computer Vision,* J. L. Mundy and A. Zisserman, Eds. Cambridge, MA: MIT Press, 1992.

[24] R. Horaud, S. Christy, F. Dornaika, and B. Lamiroy, "Object pose: Links between paraperspective and perspective," in *Proc. 5th IEEE Int. Conf. Computer Vision ICCV'95,* Cambridge, MA, 1995, pp. 426–433.

[25] J. Y. S. Luh, M. W. Walker, and R. P. C. Paul, "Resolved acceleration control of mechanical manipulators," *IEEE Trans. Automat. Contr.,* vol. 25, pp. 468–474, 1980.