Uncalibrated Visual Tasks via Linear Interaction

Carlo Colombo[1] and James L. Crowley[2]

[1] Dipartimento di Elettronica per l'Automazione, Università di Brescia,
Via Branze 38, I-25123 Brescia, Italy***
[2] LIFIA-IMAG, Institut National Polytéchnique de Grenoble,
46 Avenue Félix Viallet, F-38031 Grenoble Cedex, France

Abstract. We propose an approach for the design and control of both reflexive and purposive visual tasks with an uncalibrated camera. The approach is based on the bi-dimensional appearance of the objects in the environment, and explicitly takes into account independent object motions. The introduction of a linear model of camera-object interaction dramatically simplifies visual analysis and control by reducing the size of the visual representation. We discuss the implementation of three tasks of increasing complexity, based on active contour analysis and polynomial planning of image contour transformations. Real-time experiments with a robot wrist-mounted camera demonstrate that the approach is conveniently usable for visual navigation, active exploration and perception, and man-robot interaction.

1 Introduction

Active vision systems, often borrowing from biological systems, combine selective sensing strategies and motor control techniques to optimize the execution of complex tasks. The simplest visual tasks can be regarded as reactive transformations from perception to action, where motor actions are reflexes to incoming visual data [6]. In addition, active tasks involve the purposive planning of visuo-motor strategies, and require an a priori knowledge of the visual environment [8, 3]. The problem of the integration of multiple tasks is of key importance for the design of active vision systems and more general robotic systems as well [2]. Some recent implementations of interacting and cooperating tasks – e.g. using saccadic shifts to recover from pursuit errors [12], or executing reactive saccades before an active recognition "scanpath" [5] – are explicitly inspired from the human visual system. A general framework for the integration of reactive visual processes was presented recently, in which the problem of the hierarchical organization of control processes was addressed [7]. Much work has been done, in the last few years, on the design of architectures for active camera control (visual servoing). A modern approach to visual servoing is to close the visual loop at the image level instead than in the tri-dimensional (3D) work-space, so as to reduce the system sensitivity to uncertainties in camera calibration, kinematic modeling, etc. [10].

*** Formerly at the ARTS Lab of Scuola Superiore Sant'Anna, Pisa, Italy.

In this paper, we present an approach to the design and control of active and reactive visual tasks with an uncalibrated camera. The approach, which is based on the bi-dimensional (2D) visual appearance of rigid objects in the work-space, allows independent object motions and features a task layering mechanism, has evolved from an earlier framework called Affine Visual Servoing (AVS) [4]. One of the distinguishing features of AVS is the combination of differential control and an affine model of camera-object interaction which, once that the ambiguities intrinsic to the linearization are solved, dramatically simplifies both object representation and visual servoing. We discuss a system implementation with a manipulator-mounted camera which uses active contours as image primitives and includes three different tasks: fixation, motion imitation, and relative positioning. In the latter case, we show how to generate camera displacements from polynomial planning of image contours. The techniques described here have natural applications in landmark-based visual navigation, active exploration and perception, and man-robot interaction.

2 Overview and Control of Visual Tasks

Given a model of camera-object interaction, a *visual representation* $\{\mathbf{p}, \mathbf{d}\}$ can be defined where, at each time t:

- $\mathbf{p}(t)$ is an m-dimensional parameterization of visual appearance;
- $\mathbf{d}(t)$ is a set of n differential parameters describing 2D changes of image appearance caused by the 3D relative velocity twist of camera and object.

Any visual task can be described as a desired evolution $\tilde{\mathbf{p}}(t)$ of object appearance. From a differential viewpoint, the task is specified as a trajectory $\tilde{\mathbf{d}}(t)$. This is nonzero only in the case of active tasks, while reactive tasks do not require planning.

At run-time, the current representation is estimated as $\{\hat{\mathbf{p}}, \hat{\mathbf{d}}\}$ via visual analysis and an image tracking process which we refer to as *passive tracking*, as it takes place also when the camera is fixed.

The $n \times 6$ *interaction matrix* $\mathcal{L}$ encodes the differential transformation from relative twist $\Delta\mathbf{V}$ to appearance changes:

$$\mathbf{d} = \mathcal{L}\,\Delta\mathbf{V} = \mathcal{L}\,(\mathbf{V}_C - \mathbf{V}_O) \ , \tag{1}$$

where $\mathbf{V}_C = [\mathbf{T}_C^T\ \boldsymbol{\Omega}_C^T]^T$ is the camera velocity twist and $\mathbf{V}_O = [\mathbf{T}_O^T\ \boldsymbol{\Omega}_O^T]^T$ is the object velocity twist.

A differential strategy is adopted for task control:

$$\tilde{\mathbf{V}}_C = \hat{\mathbf{V}}_O + \mathcal{L}^+(\tilde{\mathbf{d}} + k\,\mathbf{e}(\tilde{\mathbf{p}}, \hat{\mathbf{p}})) \ , \tag{2}$$

where $\tilde{\mathbf{V}}_C$ is the desired camera motion, $\hat{\mathbf{V}}_O$ is an estimate of object motion, $\mathbf{e}(\tilde{\mathbf{p}}, \hat{\mathbf{p}})$ is an n-dimensional error resulting from the comparison of the desired and estimated appearances, $k \in [0, 1]$ is the feedback gain, and $\mathcal{L}^+$ denotes the $6 \times n$ pseudo-inverse of $\mathcal{L}$. The anticipation $\hat{\mathbf{V}}_O$ is obtained as $\hat{\mathbf{V}}_O = \hat{\mathbf{V}}_C - \mathcal{L}^+\hat{\mathbf{d}}$, where

the camera motion $\widehat{\mathbf{V}}_C$ is estimated directly from joint data and robot kinematics. Position feedback, if k is properly tuned, compensates for various modeling and estimation inaccuracies (robot kinematics, interaction model, camera parameters, finite differences approximation, initial conditions, etc.). A regulation-to-zero (no planning) scheme for the control of a full-perspective camera based on the interaction matrix concept is introduced in [10], where $\mathbf{d} = \dot{\mathbf{p}}$, i.e. $n = m$. As the size of $\mathcal{L}$ ($\propto n$) is directly related to the number of visual features used to represent an object ($\propto m$) – mainly points and lines –, the use of this scheme is limited to rather simple object shapes. Below we show how, thanks to a careful modeling of the interaction, it is possible to decouple control complexity from shape complexity – n independent of m – and easily augment regulation with a suitable planning strategy based on contour features, with significant improvements over the basic scheme both in terms of loop time and stability.

As several tasks may be executed independently 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 transformations and, in ultimate analysis, 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.

3 Models and Measurements

3.1 Interaction Model

The differential interaction between camera and object can be expressed, at a generic image point $\mathbf{x} = [x\ y]^T$, in terms of the 2×6 matrix $\mathcal{V}(x, y)$ s.t.

$$\mathbf{v}(x, y) = \mathcal{V}(x, y)\, \Delta\mathbf{V} \ , \tag{3}$$

relating image velocity $\mathbf{v} = \dot{\mathbf{x}}$ (motion field) to 3D relative velocity. Under full perspective and unit focal length, the motion field matrix evaluates as

$$\mathcal{V}(x, y) = \begin{bmatrix} -1/z & 0 & x/z & xy & -(1+x^2) & y \\ 0 & -1/z & y/z & (1+y^2) & -xy & -x \end{bmatrix} \ , \tag{4}$$

with $z = z(x, y)$ s.t. $z(X/Z, Y/Z) = Z$ bringing into play the depth of the visible surface $Z = Z(X, Y)$.

Under para-perspective projection (a linearization of perspective [11]), the visible surface is approximated by a plane $Z(X, Y) = pX + qY + c$ passing through the object's centroid $[X_B\ Y_B\ Z_B]^T$ (*object plane*), with

$$c/z = 1 - px - qy \ . \tag{5}$$

Besides, for any object point para-projected in $\mathbf{x}$, it holds $(\mathbf{x} - \mathbf{x}_B)^T(\mathbf{x} - \mathbf{x}_B) \simeq 0$, where $\mathbf{x}_B$ is the centroid's image. Thus we can neglect quadratic and higher order

terms in the Taylor's development of $\mathbf{v}(x, y)$ around (x_B, y_B), and obtain a linear motion field:

$$\mathbf{v}(x, y) = \mathbf{v}_B + \mathcal{M}_B [x - x_B \; y - y_B]^T \; . \tag{6}$$

In ultimate analysis, the dynamic evolution of any object image patch has six degrees of freedom (DOF), namely the two components of $\mathbf{v}^B$ (rigid translation of the whole patch), and the motion parallax

$$[m_{11} \; m_{12} \; m_{21} \; m_{22}]^T = \mathbf{w}_B \leftrightarrow \mathcal{M}_B = \begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix} \; , \tag{7}$$

which accounts for affine image shape transformations.

The linearization of eq. (6) allows a compact representation of dynamic interaction (n small and independent of m), not achievable with a full-perspective model. Indeed, by combining eqs. (3) through (7), we can easily construct the three interaction matrices $\mathcal{V}_B$ ($n = 2$), $\mathcal{W}_B$ ($n = 4$) and $\mathcal{U}_B$ ($n = 6$) s.t.

$$\mathbf{v}_B = \mathcal{V}_B \, \Delta\mathbf{V} \; , \quad \mathbf{w}_B = \mathcal{W}_B \, \Delta\mathbf{V} \; , \quad \mathbf{u}_B = [\mathbf{v}_B^T \; \mathbf{w}_B^T]^T = \mathcal{U}_B \, \Delta\mathbf{V} \; , \tag{8}$$

and use them for designing visual tasks (see Sect. 4). Notice that the matrix $\mathcal{U}_B$ establishes a one-one correspondence between the six object DOF in the image and those in the work-space.

3.2 Passive Tracking and Feedback

To estimate at each time the current visual representation of the object (visual appearance and differential parameters) we use quadratic B-spline active contours [1]. These use 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 their M control points $\mathbf{x}_i$ and optimize image processing computations.

The six parameters of the affine transformation $\widehat{\mathbf{u}}_B$ between two successive contour estimates, $\{\widehat{\mathbf{x}}_i(t)\}$ and $\{\widehat{\mathbf{x}}_i(t+1)\}$, are obtained via least squares. The feedback error (centroid, shape) is evaluated analogously, as the least squares matching of the desired visual appearance, $\{\widetilde{\mathbf{x}}_i\}$, against $\{\widehat{\mathbf{x}}_i\}$, the estimated appearance.

To enhance the quality of all visual measurements (visual representation, object motion), simple mobile-mean filters are used [4].

3.3 Initializing and Updating the Interaction Matrix

The interaction matrix embeds, in the object plane coefficients p, q and c, information on 3D relative camera-object pose and translation (extrinsic camera parameters). Fig. 1 (left) shows an object-centered frame $\{X_{obj}, Y_{obj}, Z_{obj}\}$ fixed on the centroid $[X_B \; Y_B \; Z_B]^T$, s.t. $Z_{obj} = 0$ denotes the object plane. Relative pose is uniquely determined by the three angles $\sigma \in [0, \pi/2]$ (slant), $\tau \in [-\pi, \pi]$ (tilt) and $\varphi \in [-\pi, \pi]$, to which plane parameters are related as follows:

$$p = -\tan\sigma \cos\tau \; , \quad q = -\tan\sigma \sin\tau \; , \quad c = Z_B \cdot (1 - p x_B - q y_B) \; . \tag{9}$$

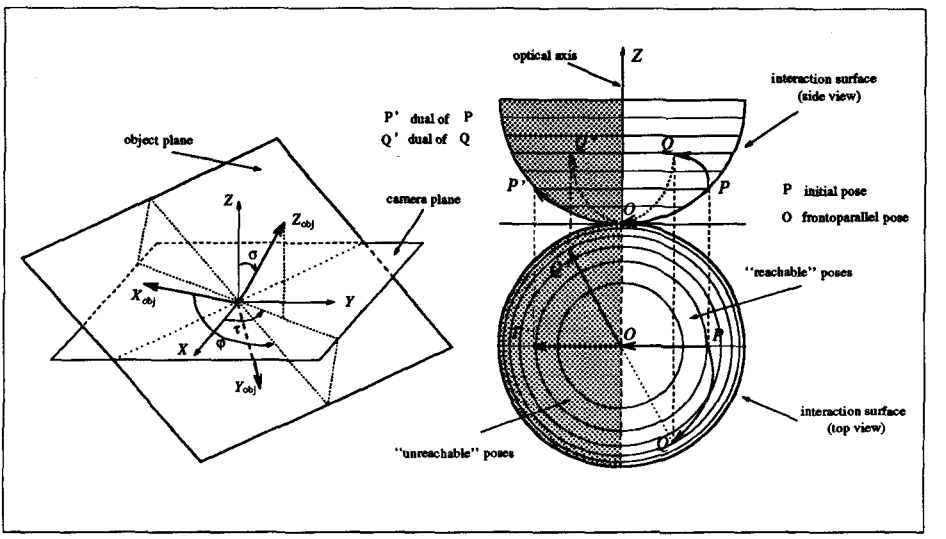

Fig. 1. *Left*: Definition of pose parameters. The camera frame has been translated for convenience in the object frame's origin. *Right*: Interaction surface and pose ambiguity for weak perspective (see Subsect. 3.4).

It can be shown [9] that the loop closure in the image rather than in the workspace greatly enhances stability w.r.t. conventional servoing approaches, and convergence is ensured even for bad initial estimates of either intrinsic and extrinsic camera parameters (uncalibrated camera). Still, having the interaction matrix even roughly estimated at start-up and updated at run-time improves the speed of convergence of the control scheme. A raw initial estimate of object pose and centroid depth is obtained using the simple weak perspective camera model $Z(X,Y) \simeq Z_B$ in the place of para-perspective, which yields $[x - x_B \; y - y_B]^T = \mathcal{T}^{wp} [X_{obj} \; Y_{obj}]^T$, with

$$\mathcal{T}^{wp} = \frac{1}{Z_B} \begin{bmatrix} \cos\tau & -\sin\tau \\ \sin\tau & \cos\tau \end{bmatrix} \begin{bmatrix} \cos\sigma & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\varphi & \sin\varphi \\ -\sin\varphi & \cos\varphi \end{bmatrix} . \tag{10}$$

An estimate of the weak perspective matrix, $\widehat{\mathcal{T}}^{wp} = \begin{bmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \end{bmatrix}$, is easily obtained from the least squares comparison of the current appearance of the object and an a priori model, e.g. the frontoparallel view of the object at unit distance and scale. Once that this is known, we can estimate both pose and scale by solving the following nonlinear system:

$$\begin{cases} t_{11} + t_{22} = 1/Z_B \cos(\tau - \varphi)(\cos\sigma + 1) \\ t_{21} - t_{12} = 1/Z_B \sin(\tau - \varphi)(\cos\sigma + 1) \\ t_{11} - t_{22} = 1/Z_B \cos(\tau + \varphi)(\cos\sigma - 1) \\ t_{21} + t_{12} = 1/Z_B \sin(\tau + \varphi)(\cos\sigma - 1) \end{cases} . \tag{11}$$

Notice that – a fact common to all perspective linearizations – a pose ambiguity exists for weak perspective. I.e., two dual solutions exist to eq. (11), as two distinct object poses share the same visual appearance: $\mathcal{T}^{\text{WP}}(Z_{\text{B}}, \tau, \sigma, \varphi) = \mathcal{T}^{\text{WP}}(Z_{\text{B}}, \tau + \pi, \sigma, \varphi + \pi)$. To disambiguate the pose, we can refer back to the full-perspective model, and choose as the "true" pose the one providing the best least squares fit against image data [11].

At run-time, pose and distance parameters are obtained by combining current estimates, obtained via eq. (1) and the differential measurements $\widehat{\mathbf{u}}_{\text{B}}$ and $\widehat{\Delta\mathbf{T}}$, and their predicted value, obtained by expressing $\dot{p}$, $\dot{q}$, $\dot{c}$ as functions of p, q, c and $\Delta\mathbf{T}$, and using finite differences [4].

3.4 Planning and Task Disambiguation

A planning strategy is used to produce a viewpoint shift (pose, translation). Such a shift is associated with an according smooth change of object's visual appearance of duration T from an initial contour $\{\mathbf{x}_i^0\}$ to a final desired contour $\{\mathbf{x}_i^{\text{T}}\}$. The mapping "in the large" between these contours is evidently affine – $\mathbf{x}_i^{\text{T}} = \mathbf{x}_{\text{B}}^{\text{T}} + \mathcal{A}_{\text{T}}(\mathbf{x}_i^0 - \mathbf{x}_{\text{B}}^0)$ – as the result of a sequence of affine transformations "in the small." 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:

$$\tilde{\mathbf{x}}_i(t) = [\mathbf{a}(t) + \mathbf{x}_{\text{B}}^0] + \mathcal{A}(t)\,(\mathbf{x}_i^0 - \mathbf{x}_{\text{B}}^0) \;, \tag{12}$$

with $\mathbf{a}(t) = \sum_{l=0}^{h} \mathbf{c}_l t^l$ a 2-vector and $\mathcal{A}(t) = \sum_{l=0}^{h} \mathcal{C}_l t^l$ a 2×2 matrix, $\mathbf{c}_l$ and $\mathcal{C}_l$ being constants to be determined based on boundary conditions. The conditions $\mathbf{a}(0) = \mathbf{0}$, $\mathcal{A}(0) = \mathcal{I}$, $\mathbf{a}(T) = \mathbf{x}_i^{\text{T}} - \mathbf{x}_i^0$, and $\mathcal{A}(T) = \mathcal{A}_{\text{T}}$ ensure that the contour evolution starts with the initial contour and terminates with the desired contour. From the solution

$$\mathbf{a}(t) = \xi(t)(\mathbf{x}_{\text{B}}^{\text{T}} - \mathbf{x}_{\text{B}}^0) \;, \quad \mathcal{A}(t) = \xi(t)\mathcal{A}_{\text{T}} + [1 - \xi(t)]\mathcal{I} \;, \tag{13}$$

where $\xi(t) \in [0, 1]$, the desired differential reference is computed as $\tilde{\mathbf{v}}_{\text{B}}(t) = \dot{\mathbf{a}}(t)$ and $\tilde{\mathbf{w}}_{\text{B}}(t) \leftarrow \widetilde{\mathcal{M}}_{\text{B}}(t) = \dot{\mathcal{A}}(t)\mathcal{A}^{-1}(t)$. Additional constraints on the derivatives of $\mathbf{a}(t)$ and $\mathcal{A}(t)$ – with beneficial effects on contour tracking, visual analysis and camera velocities and accelerations at the expense of a slower convergence – can be imposed at the trajectory endpoints with $h \geq 3$. Smooth trajectories are obtained with cubic ($h = 3$) or quintic ($h = 5$) polynomials by imposing zero endpoint velocity and acceleration:

$$\xi(t) = \begin{cases} \chi^2(t)[3 - 2\chi(t)] & \text{if } h = 3 \;, \\ \chi^3(t)[6\chi^2(t) - 15\chi(t) + 10] & \text{if } h = 5 \;, \end{cases} \tag{14}$$

with $\chi(t) = (t/T) \in [0, 1]$ the normalized task time.

The smooth pose shift produced by the planning strategy can be represented as a curvilinear path on the *interaction surface* – the semi-sphere of all possible relative orientations between camera and object plane (Fig. 1, right). As it is,

planning produces always, of the two dual poses $\mathbf{Q}$ and $\mathbf{Q}'$ sharing the same goal appearance under weak perspective, the one which is closest to the initial pose moving along a geodesic path on the interaction surface ($\mathbf{Q}$). To reach the farthest pose ($\mathbf{Q}'$) instead, we split the path $\mathbf{P} \mapsto \mathbf{Q}'$ in two parts, $\mathbf{P} \mapsto \mathbf{O}$ and $\mathbf{O} \mapsto \mathbf{Q}'$, and pass through a suitably scaled frontoparallel view of the object $\mathbf{O}$.

4 Implementation and Results

Three Visual Tasks: Definition and Composition. Tab. 1 introduces, in order of complexity, the three tasks implemented using the interaction matrices defined in eq. (8). Indices of task computational complexity are loop time, degree of object representation and initial conditions required.

TASK SYNOPSIS	Fixation Tracking	Reactive Tracking	Active Positioning
Initial conditions	$\{\mathbf{x}_i^0\}$	$\{\mathbf{x}_i^0\}$	$\{\mathbf{x}_i^0\}, \{\mathbf{x}_i^T\}$
Visual Representation	$\{\mathbf{x}_B, \mathbf{v}_B\}$	$\{\{\mathbf{x}_i\}, \mathbf{u}_B\}$	$\{\{\mathbf{x}_i\}, \mathbf{u}_B\}$
Task Description	$\widetilde{\mathbf{x}}_B = \mathbf{0}$	$\{\widetilde{\mathbf{x}}_i(t)\} = \{\mathbf{x}_i^0\}$	$\{\widetilde{\mathbf{x}}_i(T)\} = \{\mathbf{x}_i^T\}$
Interaction Matrix	$\mathcal{V}_B$	$\mathcal{U}_B$	$\mathcal{U}_B$
Task Type	reactive	reactive	active

Table 1. Task Synopsis.

During *fixation tracking*, a reactive task important in both artificial and biological vision systems [6], the camera is constrained to fix always a specific point of the object. Thanks to the linear interaction model, the centroid of the object's visible surface, chosen here as fixation point, is tracked by forcing the imaged object's centroid to be zero. The goal of *reactive tracking* is to imitate the motion of the object in the visual environment. An estimate of object motion can be also derived directly from joint data. Such a task can be useful to human-robot interfacing (mimicking human gestures, person following, etc.). Differently from fixation, the image point with constant zero speed is not, in general, the image origin, and the direction of gaze does not normally coincide with the direction of attention. Such an attentive shift is possible also in humans, but only if voluntary. The *active positioning* task consists in purposively changing the relative spatial configuration (pose, distance) of the camera with respect to a fixed or moving object. This task can be essential for the optimal execution of more complex perceptive and explorative tasks, for instance vision-based robot navigation. The linear transformation required to plan the task (see Subsect. 3.4) is obtained from the least squares comparison of the initial and goal object appearances. Shifts of visual appearance can be related to corresponding attentional shifts from a region to another of the image. The reactive tracking task can thus be regarded as a particular case of active positioning, where the goal appearance always coincides with the initial one.

As mentioned earlier, the tasks can be composed based on the value of their feedback gains (the higher the gain, the faster the task). Thus, fixation can be composed with positioning to yield the task of positioning w.r.t. a fixated object. After completion, the composite task degenerates into a reactive tracking task, which attempts to preserve the relative position and orientation between the camera and the fixated object.

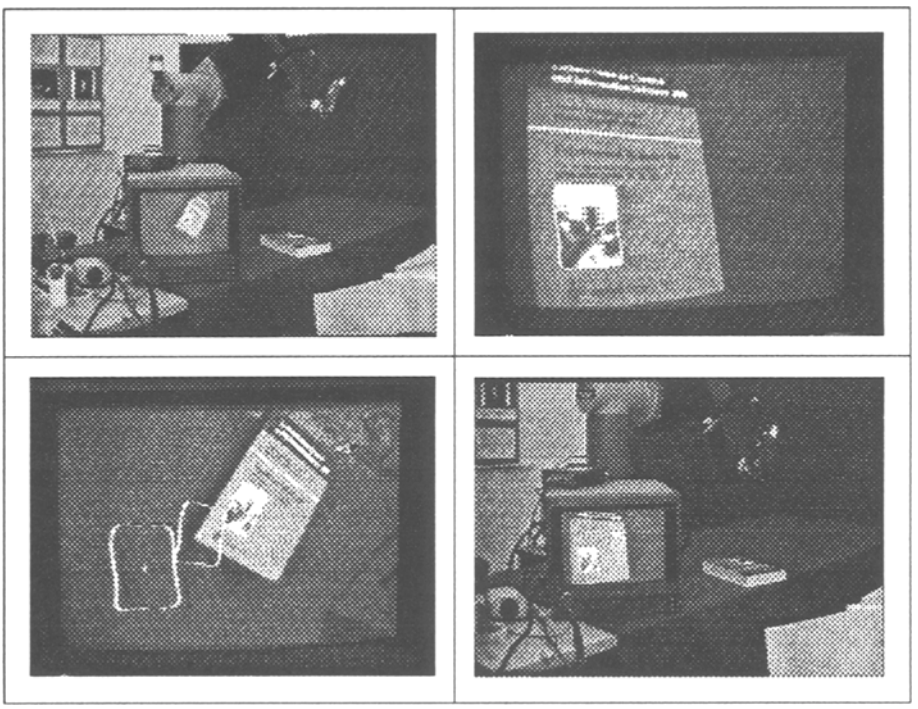

Fig. 2. A positioning experiment (see text). The monitor upon the table displays the current scene as viewed by the camera.

Setup and Parameter Setting. The system is implemented on an eye-in-hand robotic setup featuring a PUMA 560 manipulator equipped with a wrist-mounted off-the-shelf camera. Frame grabbing and control routines run on a 80486/66 MHz PC using an Imaging Technology VISIONplus-AT CFG board. The PUMA controller runs VAL II programs and communicates with the PC via the ALTER real-time protocol using an RS232 serial interface. New velocity setpoints are generated by the PC with a sampling rate $T_2 = NT_1$, where $T_1 = 28$ ms is the sampling rate of the ALTER protocol and the integer N depends on the overall loop time. Using $M = 16$ control points for B-spline contours, loop time is about 100 ms, hence $N = 4$.

Camera optics data-sheets provide a raw value for focal length and pixel dimensions; the remaining intrinsic parameters of the camera are ignored. Smoothing filters and feedback gain, all tuned experimentally, are set to $k_{pos} = 0.1$ (position) and $k_{vel} = 0.01$ (velocity), and $k = 0.1$, respectively. Cubic planning ($h = 3$) is used, which offers a good compromise between smoothness and contour inertia.

The System at Work. Fig. 2 summarizes the execution of an active positioning task, the most complex task among the three, with respect to a planar object – a book upon a table. The top left part of the figure shows the initial relative configuration, and the top right the goal image appearance. At the bottom left the planned trajectory between the initial and goal contours is sketched. At the bottom right of the figure, the obtained final configuration is shown which, as a typical performance, is reached with an error of within a few degrees (pose) and millimeters (translation).

To assess the stability characteristics of control, and tune up the feedback gain so as to obtain a slightly underdamped behavior, active positioning is run in *output regulation mode*. This mode is characterized by the absence of planning ($\tilde{\mathbf{p}} = \mathbf{0}$ *and* $\tilde{\mathbf{d}} = \mathbf{0}$), while the system brings itself, thanks to the feedback, from the initial to the goal configuration. The *servoing mode* ($\tilde{\mathbf{d}} = \mathbf{0}$) is used instead to assess the tracking performance of the control scheme, as the system is forced to compensate the feedback error $\mathbf{e}(\tilde{\mathbf{p}}, \hat{\mathbf{p}})$. Fig. 3 (left) shows the camera velocity as obtained with the servoing mode. Cubic-based planning and stable tracking contribute to obtain graceful relative speed and acceleration profiles.

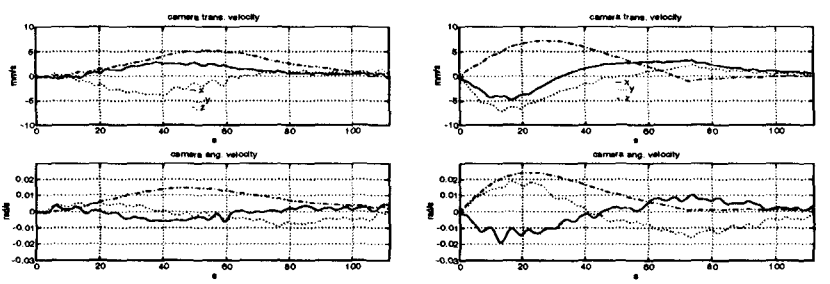

Fig. 3. Camera velocities. *Left*: Without feedforward (servoing mode). *Right*: With feedforward.

Introducing the feedforward term $\tilde{\mathbf{d}}$ in the control scheme significantly alleviates the job of the feedback and reduces the tracking lag; yet, system performance gets more sensitive to 3D interaction data – in Fig. 3 (right), feedforward was dropped out after about 75 s. Still, both control schemes (with and without feedforward) exhibit a nice behavior, also considering that the tests were run without estimating and updating on-line the interaction matrix.

Acknowledgements

Part of this work was done during a stay of C. Colombo at LIFIA-IMAG, Grenoble, as a visiting scientist supported by a fellowship from the EC HCM network SMART. The authors warmly thank Dr. B. Allotta of ARTS Lab, Pisa for his useful comments and help in the experiments.

References

1. A. Blake, R. Curwen, and A. Zisserman. A framework for spatiotemporal control in the tracking of visual contours. *International Journal of Computer Vision*, 11(2):127–145, 1993.
2. R. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, pages 14–23, 1986.
3. R. Cipolla and N.J. Hollinghurst. Visual robot guidance from uncalibrated stereo. In C.M. Brown and D. Terzopoulos, editors, *Realtime Computer Vision*. CUP, 1994.
4. C. Colombo, B. Allotta, and P. Dario. Affine Visual Servoing: A framework for relative positioning with a robot. In *Proc. IEEE International Conference on Robotics and Automation ICRA'95, Nagoya, Japan*, 1995.
5. C. Colombo, M. Rucci, and P. Dario. Attentive behavior in an anthropomorphic robot vision system. *Robotics and Autonomous Systems*, 12(3–4):121–131, 1994.
6. J.L. Crowley, J.M. Bedrune, M. Bekker, and M. Schneider. Integration and control of reactive visual processes. In *Proceedings of the 3rd European Conference on Computer Vision ECCV'94, Stockholm, Sweden*, pages II:47–58, 1994.
7. J.L. Crowley and H.I. Christensen. *Vision as Process*. Springer Verlag Basic Research Series, 1994.
8. S.J. Dickinson, H.I. Christensen, J. Tsotsos, and G. Olofsson. Active object recognition integrating attention and viewpoint control. In *Proceedings of the 3rd European Conference on Computer Vision ECCV'94, Stockholm, Sweden*, pages II:3–14, 1994.
9. B. Espiau. Effect of camera calibration errors on visual servoing in robotics. In *Proceedings of the 3rd International Symposium on Experimental Robotics ISER'93, Kyoto, Japan*, pages 182–192, 1993.
10. B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. *IEEE Transactions on Robotics and Automation*, 8(3):313–326, 1992.
11. R. Horaud, S. Christy, F. Dornaika, and B. Lamiroy. Object pose: Links between paraperspective and perspective. In *Proceedings of the 5th IEEE International Conference on Computer Vision ICCV'95, Cambridge, Massachusetts*, pages 426–433, 1995.
12. I.D. Reid and D.W. Murray. Tracking foveated corner clusters using affine structure. In *Proceedings of the 4th IEEE International Conference on Computer Vision ICCV'93, Berlin, Germany*, pages 76–83, 1993.