Proceedings of the 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems Yokohama, Japan July 26-30, 1993

Robotic Contour Following based on Visual Servoing

Eve Coste-Manière – Philippe Couvignou – Pradeep K. Khosla

The Robotics Institute
Carnegie-Mellon University
Pittsburgh, PA 15213. USA
e-mail: eve@cs.cmu.edu

Abstract

Visual tracking and servoing in robotics has shown significant progress since vision sensors have met real-time requirements. A non-contact virtual link can thus be established between a robot and its visible environment so that advanced robotic tasks like inspection or positionning can be performed. In this paper we present a coherent approach for visual servoing tasks that consist of following the contours of unknown objects using a hand-eye robotic system. The motion of the contour relative to the camera sensor is estimated in real-time by processing the measured optical flow of a set of relevant feature points. The desired motion of the end-effector is then computed to minimize the displacements of the feature points with respect to a reference configuration in the image. Control schemes are used in order to stabilize the robot and to enhance tracking performances. We followed the unifying formalism for sensor-based task specification and control as presented in [18]. We illustrate our approach by experimental results acquired with a real-time environment, where our CMU-Direct-Drive Arm II follows the boundaries of motionless objects positionned in a plane parallel to the image plane. Three degrees of freedom of planar motions are thus servoed to perform the contour-following task[1].

[1]Part of this research was done under a PostDoctoral fellowship for Eve Coste-Manière with a support from the INRIA: *Institut National de Recherche en Informatique et en Automatique* (France). Partial support for Philippe Couvignou was provided by *Renault Automobiles, Direction de la Recherche*, under an exchange program VSNA (*Volontaires Service National Actif*). This work was also supported, in part, by Sandia National Laboratory under contract AC-3752-D.

Eve Coste-Manière is now with the INRIA, Icare Project – 2004, Route des Lucioles – BP93 – 06902 Sophia Antipolis Cedex – France. Philippe Couvignou is with Renault, Direction de la Recherche – Service 0075 – 9, 11 rue du 18 Juin 1940 – 92500 Rueil-Malmaison – France.

1 Introduction

We describe in this paper an approach to *robotic contour following* of unknown objects based on *visual servoing*, using a hand-eye robotic system. Our approach provides information that can be integrated in order to enhance the execution of many industrial applications. Among others, let us mention surface inspection, telerobotics, contour deburring, tracking of seams in arc welding and mobile robot guidance.

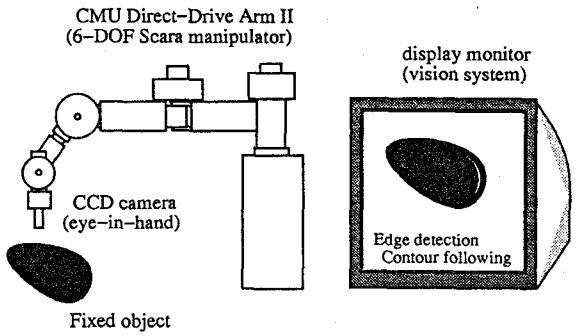

Figure 1: Eye-in-hand configuration for robotics visual contouring.

Visual tracking and servoing in robotics has shown significant progress since vision sensors have met real-time requirements. A non-contact virtual link can thus be established between a robot and its visible environment so that advanced robotic tasks like inspection or positionning can be performed in a real-world. Research in this area include [20], [12], [11], [13], [7], [16].

In [11], the authors propose a general methodology for the design of tasks integrating information provided by a visual sensor inside the control loop. It is based on the *task-function* unifying formalism to the robot control problem ([18]). This approach is particularly attractive because it demonstrates robustness properties. Furthermore, it can be used successfully

0-7803-0823-9/93/$3.00 (C) 1993 IEEE

to describe any sensor-based actions.

In this paper, we take advantage of this approach to design the control scheme of a hand-eye manipulator when following at given speed the contours of unknown motionless objects, positionned in a plane parallel to the image plane, at given depth. The motion of the contour relative to the camera sensor is estimated in real-time by processing the measured optical flow of a set of relevant feature points. The desired motion of the end-effector is then computed to minimize the displacements of the feature points with respect to a reference configuration in the image. The designed control scheme allows to stabilize the robot and to enhance tracking performances. Figure 1 shows the system configuration.

The paper is organized as follows: The task-function approach presented in [18] is recalled in section 2. We formulate our contribution within the framework of vision-based robotic servoing in section 3. We describe the selection of the visual feature points to be integrated in the control scheme in section 4. The robotic configuration of our CMU-Direct-Drive Arm II, the image processing hardware, and the experiments which validate our theoretical approach are presented in section 5.

2 Vision-based control in the task-function framework

2.1 The task-function approach

In this section, we recall the main ideas of the task-function approach ([17], [18]), for robot control and sensor based control. The approach is based on the specification of a desired objective to be reached by the robot by defining an output n-dimensional second-order continuously differentiable vector function, $e(q,t)$, called *task function*, to be regulated to zero within a finite time horizon. Straightforward examples of task-functions include:

- $e(q,t) = q - q_d(t)$, where $q_d(t)$ is a desired trajectory in joint coordinates, and
- $e(q,t) = x - x_d(t)$, where $x_d(t)$ is a desired trajectory in cartesian coordinates.

Tracking and following the contour of an unknown object by using a vision sensor requires the definition of a *hybrid tracking/following task.* Redundancy in the task-function approach allows a rigorous definition of *hybrid tasks.* An adequate choice of the task-function e allows the design of redundant tasks and the derivation of complex sensor-based control schemes. The task-function vector is derived from a first control objective called *primary task error function* e_1, and from a *secondary cost function* h_s to be minimized under the constraint $e_1 = 0$, with gradient $\partial h_s/\partial x$ with respect to the cartesian position vector x of the end-effector. The general form of the *global task error function* e is:

$$e = W^{\dagger} e_1 + \alpha \left(I_6 - W^{\dagger} W\right) \frac{\partial h_s}{\partial x}, \tag{1}$$

where W is a m×6 rectangular matrix (m$\leq$ 6) and α is a positive scalar. $I_6 - W^{\dagger}W$ is a projection (square) matrix of $\mathbb{R}^6$ which projects the 6-DOF end-effector's displacement vectors onto the null space of the matrix W (because $W(I_6 - W^{\dagger}W) = 0$). Therefore, the subspaces of $\mathbb{R}^6$ spanned by the columns of the matrices $W^{\dagger}$ and $I_6 - W^{\dagger}W$ have no vector in common but the null vector, which guarantees the separation of the degrees-of-freedom being controlled by the error vector e_1 and by the minimization of h_s in (1).

2.2 Hybrid task-function for vision-based contour following

Our approach is restricted to eye-in-hand configurations. The camera coordinate system and the end-effector coordinate system are identical, the optical center of the camera being the origin of the coordinate system attached to the end-effector. In this case, the first control objective corresponds to the positioning of the camera in end-effector's coordinates:

$$e = x - x_d. \tag{2}$$

The primary task error function e_1 integrates the displacements computed from the *position error vector* in the image. The scalar function h_s represents the objective of following the contour of the object at a desired velocity.

Let $p = (x_1, y_1, x_2, y_2, \ldots, x_n, y_n)$ be the 2N-dimensional vector of image positions where x_i, y_i, $(i = 1, \ldots, N)$ are the positions coordinates of the perspective projection on the image plane of the different feature points. We define $p - p^*$ as the *position error vector* on the image plane, p^* being the time-invariant vector of desired image positions of the feature points. The $2N$-dimensional vector $p - p^*$ of the feature points' position errors in the image carries the visual measurements that link the end-effector camera to the target object. It may be shown ([4]), that the time-derivative of p can be expressed as a function of the relative velocity vector of the object with respect to the camera as follows:

$$\frac{d}{dt}(p - p^*) = -L\left(T_c - T_M\right). \tag{3}$$

where the 6-dimensional vector T_c denotes the camera velocity, and T_M denotes the generalized velocity vector of the object, with respect to a fixed coordinate system. L is the $2N \times 6$ optical-flow matrix the elements of which depend on x_i, y_i, the depth Z_i of the feature point, the focal length f of the perspective projection model assumed for our CCD camera. We refer the reader to [4] for a detailed formulation of the optical-flow matrix.

Equation 3 shows that the relation between $\dot{p}$ and $T_M - T_c$ is a linear mapping of matrix L. In the general framework of

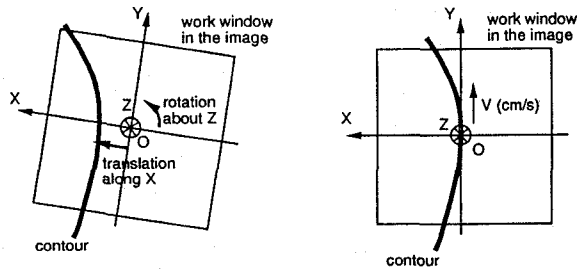

Figure 2: *Left:* Two-DOF cartesian displacements of the end-effector necessary to position the camera on the contour. *Right:* Contouring velocity V.

sensor-based control developed in [18], L is called the *interaction matrix* and is used to model any control involving sensor information that depend on the robot's position (radar, ultrasonic, vision, etc.)

Given a contour measured in the image, we need to find the 2-DOF cartesian displacement of the end-effector that compensates for the contour error: the two degrees-of-freedom consist of translational motions along the X-axis and rotational motions around the Z-axis of the end-effector's coordinate system as shown in figure 2.

Moreover, it is assumed that the error vector $p-p^*$ remains small during the servoing task. Thus, we can use the integral form of equation (3) to estimate the 6-DOF relative camera displacement that would yield the same displacement $p-p^*$ of the feature points in the image. The optimal camera displacement (in the least squares sense) is then the 6-dimensional vector:

$$e_1 = -L^\dagger(p-p^*) \tag{4}$$

The vector $-e_1 = L^\dagger(p-p^*)$ represents the 6-DOF end-effector displacement that would optimally compensate for the image error $p-p^*$ if it were directly performed by the manipulator. The matrix $W^\dagger$ projects the 6-DOF vector e_1 on the subspace spanned by the 2-DOF displacements to be controlled, which are the translation along the X-axis and the rotation around the Z-axis of the end-effector:

$$\begin{aligned} W &= \mathrm{diag}(1,0,0,0,0,1), \\ W^\dagger &= W, \\ I_6 - W^\dagger W &= \mathrm{diag}(0,1,1,1,1,0) \end{aligned}$$

The remaining four degrees-of-freedom are controlled by minimizing the secondary cost function h_s. The corresponding projection matrix is the diagonal matrix $I_6 - W^\dagger W$. Here, the objective is to follow the contour of the object at a constant desired velocity V. Under the constraint $e_1 = 0$, the contour of the object is parallel to the Y-axis of the camera's (or end-effector's) coordinate system. Therefore, the cost function to be minimized is $h_s = \frac{1}{2}(Y - Y(0) - Vt)^2$, and the gradient vector $\partial h_s/\partial x = (0, Y - Y(0) - Vt, 0,0,0,0)^T$ has only one non-zero component that corresponds to a translation along the Y-axis. We also let $\alpha = 1$ in (1). The global task-function error $e = x - x_d$ is then computed from e_1 and $\partial h_s/\partial x$ by applying equation (1); then the task-function vector e becomes input to the robot control law (6) as explained later.

2.3 Remarks: the interaction matrix L

The choice of a perspective projection model and the associated *interaction matrix* L are detailed in [4]. In the experiments, we assume that the initial depth Z_i is perfectly known and is kept constant for every feature point M_i of the object. With this simplifying assumption, no identification scheme is required to estimate the values of Z_i which appear in the matrix L. More sophisticated algorithms make use of adaptive estimation and control techniques [12, 20, 16] to overcome this simplifying restriction. Recursive identification algorithms can be incorporated in our framework. Our assumption about the knowledge of every depth Z_i does not impact the generality of our scheme.

Let us point out that uncertainties about the target's depth Z and sensor's characteristics (γ, f) alter the knowledge of the elements of L. During task execution we approximate the elements of L by their initial value computed at the goal image $p = p^*$. Thus, the optical-flow matrix L and its pseudo-inverse $L^\dagger$ are approximated by constant matrices throughout the tracking process. Doing so, significant computation time is saved at each iteration. The approximation is valid under the assumption of good tracking performance, or equivalently when the image error vector $p - p^*$ remains small. In practice, only the first and sixth rows of $L^\dagger$ need to be computed since the position error term is $W^\dagger e_1 = -W^\dagger L^\dagger(p-p^*) = -(LW)^\dagger(p-p^*)$. More precisely, the pseudo-inversion is better performed on the 2N×6 matrix LW which has all columns zero except the first and sixth columns. The computation of $(LW)^\dagger$ is done after a singular value decomposition of the two non-zero columns of the rectangular matrix LW. In addition, the N feature points must be carefully selected on the object so that the two relevant columns of the corresponding matrix LW are linearly independent [8]. A low condition number for LW is also desirable for the sake of numerical stability.

3 Robot control

3.1 Dynamic model of the robot

We consider a rigid robotic manipulator with n joints, whose positions are given by the vector q. In the sequel, we assume $n = 6$ angular joints to comply with our experimental 6-DOF CMU-Direct-Drive-Arm-II manipulator; here, $q = (\theta_1, \theta_2, \ldots, \theta_6)^T$ is the 6-dimensional vector of angular positions. The dynamic equation of the robot is:

$$\Gamma = M(q)\ddot{q} + N(q, \dot{q}, t), \tag{5}$$

where Γ is the vector of actuator torques, $M(q)$ is the inertia matrix, $N(q,\dot{q},t)$ includes centrifugal, Coriolis, friction and gravity forces. Also, the jacobian matrix, $J = J(q)$, of the manipulator is defined as $J = \partial x/\partial q$ and verifies $\dot{x} = J\,\dot{q}$, where x is the 6-dimensional vector that denotes the position and orientation of the end-effector in cartesian coordinates with respect to a world coordinate system and $\dot{x}$ is the time-derivative of x.

3.2 General control scheme

Once computed, the task-error vector $e = x - x_d$ is used as input to the robot controller. It is shown in [18] that a control law that ideally decouples and ensures a linear behavior of the task error vector e follows the general expression:

$$\Gamma = M\left(\frac{\partial e}{\partial q}\right)^{-1}\left[-kG\left(\mu De + \dot{e}\right)\right] + N - M\left(\frac{\partial e}{\partial q}\right)^{-1} f, \qquad (6)$$

where Γ is the vector of applied actuator torques introduced in equation (5), f comes from terms in the second-order time-derivative of e, $\dot{e} = \frac{\partial e}{\partial q}\dot{q} + \frac{\partial e}{\partial t} = J\dot{q} + \frac{\partial e}{\partial t}$, G and D are positive matrices, and k and μ are positive scalars. In practice, the control in (6) is implemented by using model-based approximations or recursive estimates of the different parameters $M\left(\frac{\partial e}{\partial q}\right)^{-1}, N, f$. Equation (6) covers most of existing schemes: computed torque, resolved motion rate or acceleration control, indirect adaptive control and so forth. The theory also provides sufficient conditions on the parameters to ensure the stability of the control.

3.3 Cartesian PD controller

We experimented with a cartesian PD robot control scheme with gravity compensation, the mathematical model of which is now recalled. Given a desired cartesian position $x_d(t)$, velocity $\dot{x}_d(t)$ and acceleration $\ddot{x}_d(t)$ for the end-effector, the cartesian PD controller computes a torque vector Γ and sends it to the joint actuators. The torque vector Γ is computed as follows ([15]):

$$\Gamma = J^T F + g, \quad \text{with} \qquad (7)$$

$$F = K_p(x_d - x) + K_d(\dot{x}_d - \dot{x}) + \ddot{x}_d, \qquad (8)$$

where g accounts for gravitation forces, F is a 6-dimensional vector of virtual generalized force expressed in cartesian coordinates, K_p and K_d are diagonal gain matrices.

Let us identify the parameters $M\left(\frac{\partial e}{\partial q}\right)^{-1}$, N, D, G, f, k, μ, which appear in the general control law (6) with the parameters K_p, K_d, g, actually used for implementing the cartesian PD controller with gravity compensation as described by equations (7)-(8). In our experiments, the control inputs are restricted to end-effector position errors $e = x - x_d$ computed from visual measurements and to joint angular velocities $\dot{q}$. Therefore, we must let $x_d - x = -e$, $\dot{x}_d = 0$, $\ddot{x}_d = 0$, and $\dot{x} = J\dot{q}$ in equations (7)-(8) which give:

$$\Gamma = -J^T K_p e - J^T K_d J\dot{q} + g. \qquad (9)$$

From the expression for the gradient $\partial h_s/\partial x$ of the chosen secondary cost function h_s, we obtain $|\partial e/\partial t| = \alpha|V| = |V|$. The approximation $\partial e/\partial t = 0$ necessary to simplify $\dot{e} = J\dot{q} + \frac{\partial e}{\partial t}$ in (6) is valid under the assumption that the contouring velocity V is small, which is the case in our experiments ($|V| <$ 2cm/s). Thus, the different parameters in the general control equation (6) are easily identified as follows [2]: $\hat{M}\left(\frac{\partial \hat{e}}{\partial q}\right)^{-1} = J^T$ with an obvious breach of notation. This is justified since the matrices product $\left(\frac{\partial \hat{e}}{\partial q}\right)\cdot\left(\frac{\partial \hat{e}}{\partial q}\right)^{-1} = J\hat{M}^{-1}J^T$ is strictly positive, and therefore the stability of the control is ensured. Also, $\hat{N} = g, \hat{f} = 0, k = \mu = 1, G = K_d$ and $D = K_d^{-1}K_p$.

The overall organization of the visual servoing system is shown in figure 3.

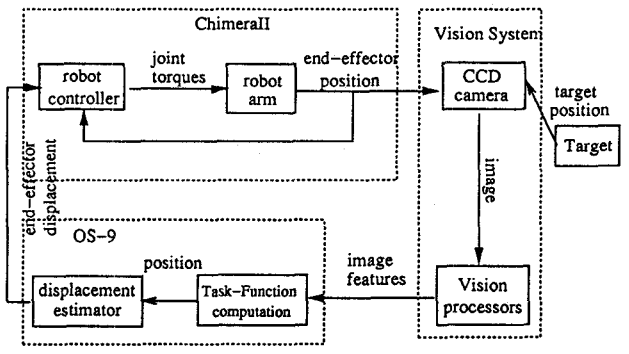

Figure 3: Schematic organization of the visual servoing system.

4 Feature points selection

We now discuss the extraction of the characteristic parameters to be used in the control. Frequent approaches encountered in the literature consist of using contour primitives (like points, corners, segments, rectangles, general ellipses) [10, 3, 9]... or region primitives (like areas, centroids and central moments, etc) [11]... Here, *feature points* are chosen. We do not assume any a priori knowledge of the geometric structure of the 3D scene to analyze. The approach presented above is general enough to perform robust contour following on lines, curves, burrs and so on.

4.1 Pre-positioning

Before carrying out the contour-following task itself, we wish to pre-position the end-effector along the edge of our target object. The user characterizes the edge by setting, in the image plane, a working window "on" the edge of the object. N feature points are automatically extracted on the object's edge,

[2] we use carets to denote the model estimates of the true terms

as local maxima of the gradient magnitude in the gradient direction on N equidistant horizontal lines. It is illustrated in figure 4. The lines allow to define the $2N$-dimensional vector of image positions p. In the mean time, N target-points are automatically generated in the window, on the y axis of the window's frame thus defining the $2N$-dimensional vector of desired image positions p^*. The cartesian displacement of the end-effector can then be computed to compensate for the position error vector on the image plane $p - p^*$, as mentioned in section 2.

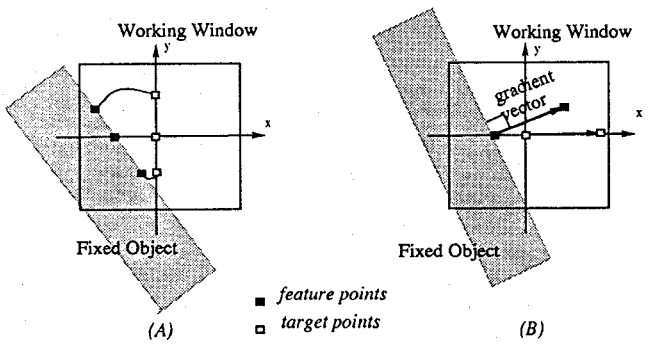

Figure 4: (A): Line Detection – (B): Gradient Computation.

4.2 Contour Following

The local maxima of the gradient magnitude in the gradient direction is now computed along the local x axis of the window frame defining a feature point on the edge of the object. The corresponding target point is the center of the window. A sub-window of fixed size is automatically generated and centered around the feature point. The average image gradient vector is computed in this area. This vector is normalized, and its end-point defines a feature point. The corresponding target point is positioned on the local x axis, and define a unit vector. The end-effector displacement is then generated (cf section 2).

4.3 Remarks

- Instead of using two points to represent the gradient vector, that is to say four redundant scalars while three are enough, we could have used the relevant components for the straight lines as mentioned in [11]. Choosing points eases the understanding of the equations.
- We chose the tracked edge to be vertical in the image. The problem of locating the edge is then reduced to one dimension (along the x axis of the window frame). This simplification reduces the processing time required by the vision process. It also improves the robustness and the speed of the image point detection process (disturbances in the image not parallel to the edge can be avoided).
- Because of noise in the measurement of the image error vector $p - p^*$, the desired end-effector's motions must be filtered before they are fed to the robot controller. The two non-zero displacements of the primary task-error vector We_1 (translation along X and rotation about Z) are filtered in parallel by two *independent* LQG regulators (see [1] and [7] for details). More precisely each displacement measurement passes through a recursive Kalman filter with a constant linear feedback on the optimal state estimate.

5 Experimental results

5.1 Hardware implementation

We present here the results of the experiments performed in real-time with the following robotic equipment:

Our experimental hardware consists of the CMU-DirectDrive-Arm-II manipulator and a vision processing system, both integrated under the Chimera-II [19] real-time operating system. The CMU-DDARM-II is a six degrees-of-freedom Scara-like manipulator, whose joints are actuated by direct-drive DC-brushless motors [15, 14]. We mounted a CCD camera of focal length f=7.5mm in the end-effector of the robot. For further details on the experimental hardware, we refer the reader to [7, 4].

The experiments were performed in conjunction with a cartesian PD controller with gravity compensation at a frequency of 500Hz. The dynamic equations and control algorithms are recalled in section 3.1. The desired end-effector motions are updated every 100 to 200ms (5 to 10Hz). Because the CMU-DDARM-II can move very fast, the reference inputs must be extrapolated by a linear ramp between two consecutive vision measurement inputs. Extrapolated data are then sent to the cartesian PD controller every 2ms (500Hz). As a result, the trajectories of the robot are significantly smoothed.

5.2 Results

During the experiments, two dark objects on a white background, in a plane parallel to the image plane, offered a maximum contrast and were observed under regular lighting condition:

- a dark ruler provides straight line reference trajectories necessary to check the good following of linear contours,
- an unpolished industrial metallic object is used to verify the performance when following curves.

In all experiments, the working window is first selected manually. With a mouse device, the user selects a window which includes a portion of the boundaries of the object. Then the N feature points are automatically generated as explained in section 4; they provide the reference position vector p^* in the image. The user must specify the desired number N of feature points as well as the desired contouring velocity V. It is

clear that N must be greater than or equal to 2 in order to allow the 2-DOF motions of the object (translation along X and rotation about Z) to be detectable and computable. The targets are observed at a known distance of 70 cm from the camera. The maximum contouring velocity at this distance is about $V = 2$ cm/sec.

The vision process computes the desired motion of the end-effector (two translational DOF along X and Y, and a rotational DOF around Z) with a period of $\Delta t = 70$ms (≈ 14Hz) when the N-point method is used, and a period of $\Delta t = 170$ms (≈ 6Hz) when the image gradient is computed.

The plots in figure 5 and 6 show the measured end-effector positions in world-frame coordinates at a desired speed of 1.5 cm/s (the desired trajectories are drawn with dot-dashed lines). The first trajectory follows a portion of the line $aX + bY - 1 = 0$, from point $(X_s, Y_s) = (0.77\text{m}, -0.26\text{m})$ to point $(X_e, Y_e) = (0.85\text{m}, -0.10\text{m})$. The second trajectory shows a portion of the circle $(X - X_o)^2 + (Y - Y_o)^2 - R^2 = 0$, with $(X_o, Y_o) \approx (0.5\text{m}, 0.01\text{m})$, and $R = 200$mm. The target is positioned in the horizontal plane at $Z = -70$cm, so that the desired trajectory of the end-effector remains in the plane $Z = 0$. The tracking error remains within $\pm 3mm$ or ± 10 pixels. The oscillations are partially due to the low accuracy of the cartesian PD control scheme, which does not take into account the full dynamics of the robot arm. Further experiments could be carried out with a computed-torque control scheme (running at 350Hz) to outline the difference in performance. In addition, friction was not totally compensated on the sixth joint of the manipulator.

During the experiments, the proposed method was found to be very robust and accurate. Significant perturbations done by moving the object or experimenting on a patterned background were successfully rejected. Following the contours of small details (like burrs on a molded object, see [5] for a detailed study) has been carried out successfully. The repeatability offers good results for industrial applications, given the accuracy of the DDARM.

6 Summary

We have presented theoretical aspects and experimental results of a contour following scheme in the framework of robotics visual servoing with a hand-eye camera. The contribution of this work is new and original because our method is presented using the generality of the *task-function* approach with all its theoretical advantages. The experimental results achieved on a real-time robotic environment demonstrate the feasibility, performance and robustness of the proposed approach to perform contour following of unknown objects.

Future research issues include the implementation of adaptive control techniques in order to evaluate and control the distance between the camera and the object. Also, we restricted our study to the formal specification in continuous time of the control law. However, the discrete-time, event-driven behavior of the sensor-based task, associated with the logical structure of its execution, could be defined using synchronous programming languages like ESTEREL [2] as advocated in [6]. It would facilitate the handling of exceptions and conditions like joint limitations, feature points occlusion or obstacle avoidance. Particular attention must be paid to handle corners whenever the object has sharp contours. Corners (or discontinuities) can be handled using the same proposed approach, provided that the desired motion is smaller than the maximum rotation allowed by the end-effector. An alternate solution would be to define a "spy" window ahead of the work window, where an accurate corner detection algorithm [9] is implemented. Once detected, it is possible to pass through the discontinuity at a given speed. The contour following task carried out with a hand-eye system can be used as a preliminary step before a hybrid force/position exploration task of the same object.

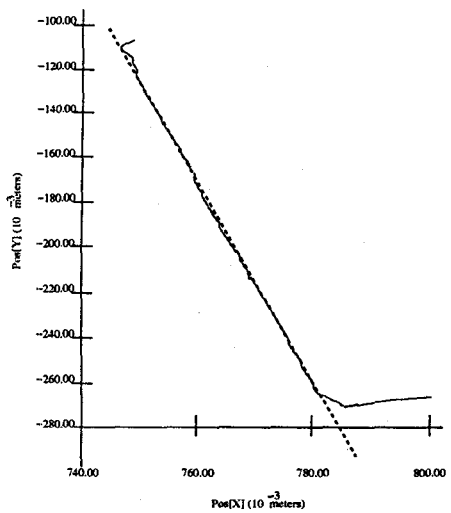

Figure 5: End-effector trajectory in the X-Y plane as the robot is following a line.

References

[1] B.D. Anderson, J.B. Moore, 1990: *Optimal Control - Linear quadratic methods*, Prenctice Hall.

[2] F. Boussinot, R. de Simone, September 1991: *The Esterel Language*, IEEE Special Section on Real-Time Programming.

[3] J.F. Canny, November 1986: *A Computational Approach to Edge Detection*, IEEE/PAMI, Vol.8, No.6, pp. 679-698.

[4] E. Coste-Manière, P. Couvignou, P.K. Khosla, 1993: *Visual Servoing in the Task-Function Framework: a Contour Following Task*, Technical Report, AML, The Robotics Institute, Carnegie-Mellon University.

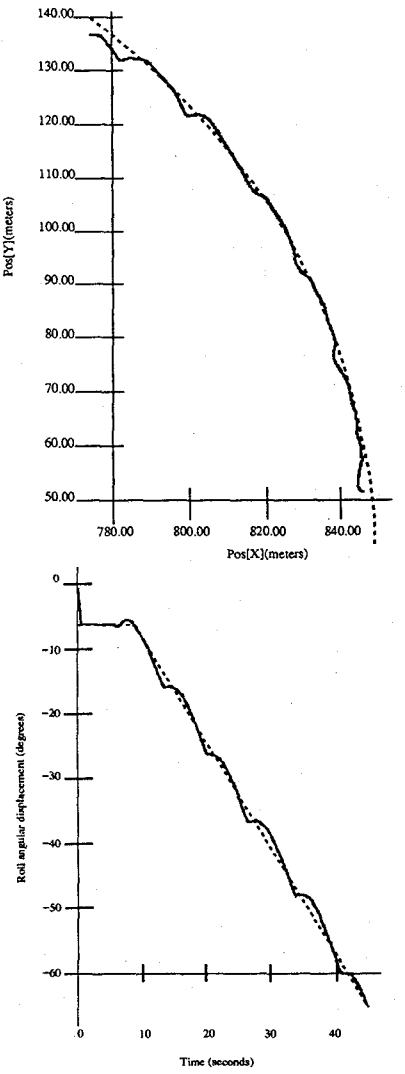

Figure 6: *Top:* End-effector position in the X-Y plane as the robot is following a circular contour (cartesian PD controller). The oscillations are due to uncompensated friction in the robot's sixth joint. *Bottom:* Roll angular displacement of the end-effector *vs.* time.

[5] E. Coste-Manière, M. Gatenholm: *A Contribution of Visual Servoing Techniques to Robotics Deburring*, SPIE: Intelligent Robots and Computer Vision XII: Active Vision and 3D Methods, Boston, to appear September 1993.

[6] E. Coste-Manière, B. Espiau, D. Simon, May 1992: *Reactive Objects in a Task Level Open Controller*, IEEE International Conference on Robotics and Automation, Nice, pp 2732-2737.

[7] P. Couvignou, N. Papanikolopoulos, P.K. Khosla, July 1992: *Hand-Eye Robotic Visual Servoing around Moving Objects using Active Deformable Models*, IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'92), Vol.3, Raleigh.

[8] P. Couvignou, P.K. Khosla, 1993: *Singularities in the Recovery of Rigid Body Motions from Optical Flow Data*, Technical Report, AML, The Robotics Institute, Carnegie-Mellon University.

[9] R. Deriche, G. Giraudon, April 1991: *Accurate Corner Detection: an Analytical Study*, INRIA Research Report N°.1420.

[10] B. Espiau, P. Rives, April 1987: *Closed-Loop Recursive Estimation of 3D Features for a Mobile Vision System*, IEEE International Conference on Robotics and Automation, Vol.3, pp. 1436-1443, Raleigh.

[11] B. Espiau, F. Chaumette, P. Rives, June 1992: *A New Approach to Visual Servoing in Robotics*, IEEE Transactions on Robotics and Automation, Vol.8, No.3.

[12] J.T. Feddema, C.S.G. Lee, 1990: *Adaptive Image Feature Prediction and Control for Visual Tracking with a Hand-eye Coordinated Camera*, IEEE Trans on Robotics and Automation, Vol.20, No.5, pp. 1172-1183.

[13] W. Jang, Z. Bien, April 1991: *Feature-Based Visual Servoing of an Eye-in-Hand Robot with Improved Tracking and Performance*, IEEE International Conf. on Robotics and Automation, pp. 2254-2260.

[14] P.K. Khosla, T. Kanade, April 1989: *Real-Time Implementation and Evaluation of the Computed Torque Scheme*, IEEE Transactions on Robotics and Automation, Vol.5, No.2, pp. 245-253.

[15] P.K. Khosla, T. Kanade, February 1988: *Experimental Evaluation of Non-Linear Feedback and Feedforward Control Schemes for Manipulators*, The International Journal of Robotics Research, Vol.7, No.1.

[16] N. Papanikolopoulos, P.K. Khosla, T. Kanade, April 1991: *Vision and Control Techniques for Robotic Visual Tracking*, IEEE International Conference on Robotics and Automation, pp. 857-864.

[17] C. Samson, B. Espiau, August 1990: *Application of the Task-Function Approach to Sensor-based Control of Robot Manipulators*, IFAC Tallinn, USSR.

[18] C. Samson, M. Le Borgne, B. Espiau, 1991: *Robot Control: the Task-Function Approach*, Oxford Press.

[19] D.B. Stewart, D.E. Schmitz, P.K. Khosla, September 1990: *Implementing Real-Time Robotic Systems using* CHIMERAII , IEEE International Conference on Robotics and Automation, pp. 598-603.

[20] L.E. Weiss, A.C. Sanderson and C.P. Neuman, 1987: *Dynamic Sensor-Based Control of Robots with Visual Feedback*, IEEE Journal of Robotics and Automation, Vol.3, No.5, pp. 404-417.