Controlling an uninstrumented manipulator by visual servoing

E. Marchand, François Chaumette, F. Spindler, M. Perrier

Controlling an uninstrumented manipulator by visual servoing

Éric Marchand, François Chaumette, Fabien Spindler
IRISA - INRIA Rennes
Campus de Beaulieu,
35042 Rennes Cedex, France
{marchand, chaumett, fspindle}@irisa.fr

Michel Perrier
Ifremer Toulon
Zone portuaire de Brégaillon, BP 330
83507 La Seyne-sur-mer Cedex, France
Michel.Perrier@ifremer.fr

Abstract

In this paper we present a method to control the displacement of a robot arm with no proprioceptive sensor. The joint positions are not available and this manipulator is usually open-loop controlled. In order to get a more efficient control interface, we propose a closed-loop system based on an eye-to-hand visual servoing approach. We show that, using such an approach, measurements of the manipulator motion with proprioceptive sensors is not required to precisely control the end-effector motion. We propose solutions for position-based control and velocity control of the manipulator. To maintain the end effector in the camera field of view, the camera orientation is also controlled. Various results show the validity and the efficiency of the approach.

The International Journal of Robotics Research, Vol. 21, No 7, July 2002, pp. 635-641

1 Introduction

Visual servoing [Hutchinson 96] has long proved to be a very efficient method to control manipulator in hostile environments (nuclear environment, space, underwater robotics, etc.). In this paper we present a generic framework to control a roughly modeled and calibrated manipulator using an eye-to-hand visual servoing system [Allen 93, Hager 95, Horaud 98].

The motivation was the control of the manipulator of the Victor 6000. Victor 6000 [Nokin 97] is a deep underwater Remote Operated Vehicle (ROV), built and operated by Ifremer, used for the exploration of the ocean floors. It is a cabled vehicle which is controlled from a support vessel and is designed to make optical surveys, to carry out local assignments for imagery, implement instrumentation and sample water, sediments or rocks. Victor 6000 is equipped with two manipulators: a 6 dof manipulator called Maestro and a 4 dof manipulator called Sherpa. In this underwater context, eye-in-hand visual servoing has already been used to control ROVs (e.g., [Rives 97, Lots 00, Lots 01, Van Der Zwaan 01]). As already stated, the goal is not to control the ROV itself, but to control the motion of its manipulator using information provided by a camera. This camera is mounted on a pan/tilt head available on the ROV and observes the end-effector of the manipulator. To reduce its cost, the manipulator was not instrumented. Due to this lack of proprioceptive sensors, the odometry, and in particular the joints positions $\mathbf{q}$ are not available and therefore the robot is usually open-loop controlled with a joystick. Hence there is no way to measure the manipulator motion and accordingly any control will be imprecise if no external sensor is used to provide a closed-loop system. This motivates the introduction of a generic framework to control such non-instrumented manipulator from a moving camera within the visual servoing framework.

The proposed vision-based approach features several main interests and contributions. Hostile environments (e.g., pressure in deep underwater conditions) may increase dramatically the cost of sensors, in this case proprioceptive sensors. In this paper we show that measurements using these proprioceptive sensors are not required to precisely control the motion of an end-effector. Indeed, the control loop is "closed" by the camera which replaces efficiently odometry sensors. This control can be achieved even if the camera is itself in mo-

Figure 1: The Ifremer Victor 6000 underwater ROV and its 6 dof Maestro manipulator (© Ifremer)

tion. Finally this approach is robust to bad/rough calibration of the system. Furthermore a major interest lies in the independence of the position reached by the arm to the various modeling and calibration errors, as well of the sensor as of the arm, if it is possible to express the task assigned with the manipulator directly in the measurement space of the sensor (i.e., as an image to reach).

In the proposed system, the manipulator is controlled using an eye-to-hand 3D visual servoing approach, while the pan-tilt camera motion itself is also controlled by visual servoing. We will consider a coarse calibrated camera. Indeed, because of the closed loop used in the control, visual servoing techniques are known to be robust with respect to calibration errors. The proposed system features three main capabilities: pose control and velocity control. In the basic **pose control** mode, the manipulator has to achieve a specified displacement. Before carrying out this displacement, a necessary step is to compute the corresponding desired position of the manipulator end-effector in the camera frame. When the desired manipulator position is computed, one can then use the traditional visual servoing techniques to achieve the positioning task. The presented control strategy presents the advantages of position-based control (optimal trajectory : geodesic for rotation, straight-line for translation), while avoiding its main drawback by ensuring that the target remains in the camera field of view due to the pan-tilt control of the camera. This phase of control requires the use of the inverse

manipulator Jacobian which depends on the current value of the manipulator joints. Since these values are unknown, the position of the end-effector in the camera frame is estimated for each acquired image using a pose computation algorithm. The manipulator joints values are then computed by solving the equation of the inverse robot geometrical model. Next, this approach is expanded to permit a **velocity control** mode. From a specified displacement expressed as a velocity, the control problem then consists in determining the trajectory that the end-effector has to track and to propose a control law able to follow this trajectory while minimizing the tracking errors. Finally, it is important to control the camera pan and tilt in order to ensure that the end effector of the manipulator remains in the camera field of view whatever the specified displacement. Though the control of this camera is a trivial problem from a visual servoing point of view, its motion induces a more complex modeling of the eye-to-hand manipulator control.

The reminder of this paper is organized as follows: the next section presents how to control the manipulator motion by visual servoing both in position (the operator specifies a displacement to achieve) and in velocity (in that case the operator specifies a velocity to follow) ; then the image processing algorithm is described ; finally we show on various experimental results the efficiency of the approach.

2 Image-based control

2.1 Notation and definitions

Let us note by ${}^a\mathbf{M}_b$ the transformation between frame $\mathcal{R}_a$ and frame $\mathcal{R}_b$. ${}^a\mathbf{M}_b$ is an homogeneous matrix defined as:

$$ {}^a\mathbf{M}_b = \begin{pmatrix} {}^a\mathbf{R}_b & {}^a\mathbf{T}_b \\ 0 & 1 \end{pmatrix} $$

where ${}^a\mathbf{R}_b$ and ${}^a\mathbf{T}_b$ define respectively the rotation matrix and the translation vector between the two frames. If $(x_b, y_b, z_b, 1)^T$ are the coordinates of a point expressed in $\mathcal{R}_b$, the coordinates $(x_a, y_a, z_a, 1)^T$ of this same point expressed in $\mathcal{R}_a$ are given by $(x_a, y_a, z_a, 1)^T = {}^a\mathbf{M}_b(x_b, y_b, z_b, 1)^T$.

The frames used in this paper are represented on Figure 2. The first letter represents the

origin of the frame (c for camera, e for effector, and o for the object linked to the effector and observed by the camera) and the second letter the position of this frame (i for initial position, c for current, and d for desired). Finally, $\mathcal{F}_m$ represents the base frame of the manipulator while $\mathcal{F}_{pt}$ represents the base frame of the pan/tilt unit. For example ${}^{ci}\mathbf{M}_{od}$ defines the desired position of the object in the initial camera frame.

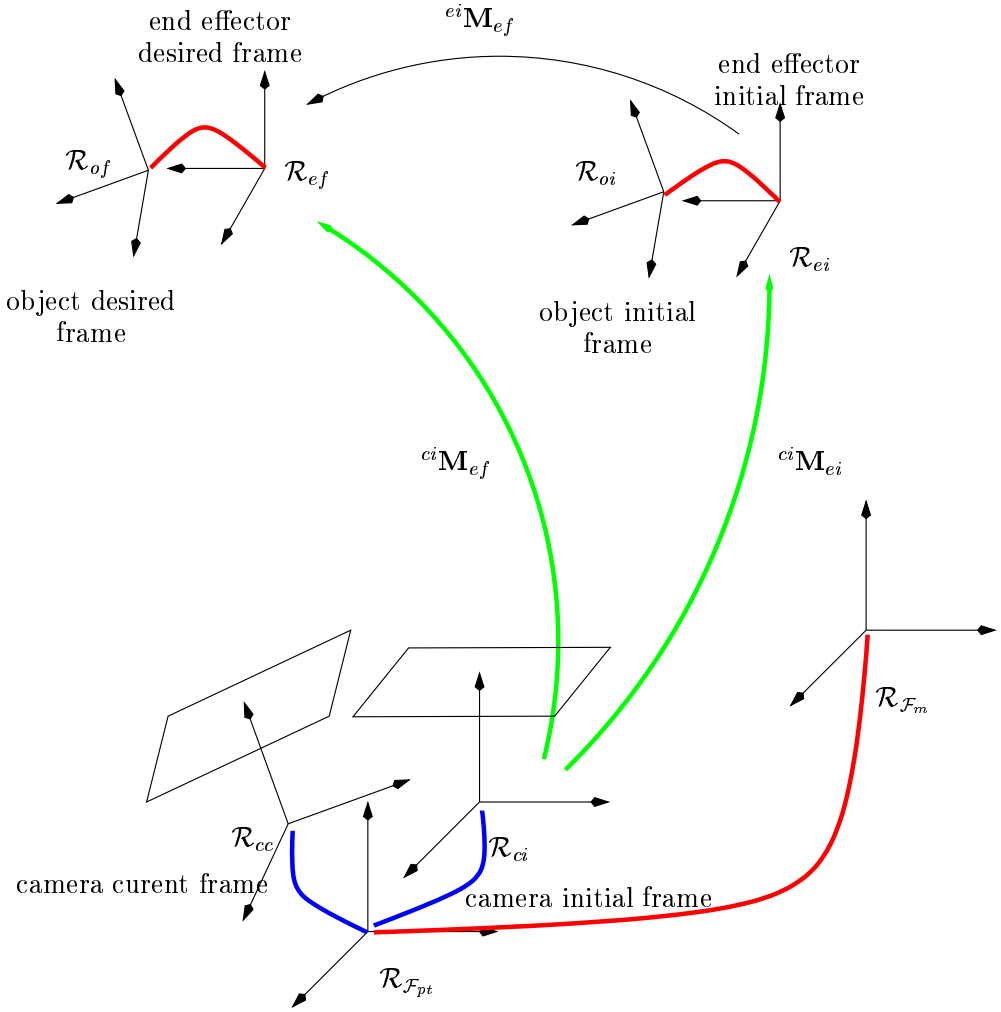

Figure 2: Overview of the various frames

In the reminder of this paper, we will refer to many frame transformations that must be estimated in a rough calibration step using either information provided by the camera or by the system itself:

- ${}^{\mathcal{F}_{pt}}\mathbf{M}_{\mathcal{F}_m}$ is constant and hand-measured ;

- ${}^{ei}\mathbf{M}_{oi} = {}^{ec}\mathbf{M}_{oc} = {}^{ed}\mathbf{M}_{od} = {}^{e}\mathbf{M}_{o}$ is constant and hand-measured ;
- ${}^{\mathcal{F}_{pt}}\mathbf{M}_{c}$ is measured using the pan/tilt head odometry.

2.2 Visual servoing: overview

Visual servoing techniques [Espiau 92, Hutchinson 96, Hashimoto 93, Papanikolopoulos 93] allow automatic positioning of a robot with respect to its environment using visual data. It consists of specifying a task as the regulation of a set of information extracted from the images [Espiau 92, Hashimoto 93].

As usual, a vision-based task $\mathbf{e}$ is defined by [Espiau 92, Samson 91]:

$$\mathbf{e} = \mathbf{J}^{+}(\mathbf{P} - \mathbf{P_d}) \tag{1}$$

where $\mathbf{P}$ denote the set of selected visual features used in the visual servoing task and $\mathbf{P_d}$ their desired value. $\mathbf{J}^{+}$ is the pseudo-inverse of the Jacobian Matrix that links the image space to the operational space of the manipulator.

To make $\mathbf{e}$ decreases exponentially and behaves like a first order decoupled system, the velocity $\mathbf{T} = (\mathbf{V^T}, \mathbf{\Omega^T})^{\mathbf{T}}$ of the end-effector given as input to the manipulator controller is given by:

$$\begin{pmatrix} \mathbf{V} \\ \mathbf{\Omega} \end{pmatrix} = -\lambda \mathbf{e} \tag{2}$$

where λ is a proportional coefficient.

2.3 Control of the manipulator position

2.3.1 Overview of the algorithm

Task specification. The goal of the task is for the manipulator to achieve the displacement specified by the ROV operator. Two methods are available to specify this desired displacement:

- First a direct definition of the desired manipulator displacement ΔT, ΔR in, possibly, three different frames $\mathcal{R}_{ci}$ (initial camera frame), $\mathcal{R}_{oi}$ (initial object frame), and $\mathcal{R}_{\mathcal{F}_m}$ (end effector frame).

- Second a definition in the image space. It corresponds to reach again a position that has been learned in an off-line learning step. From this previously learned image, the desired pose ${}^{ci}\mathbf{M}_{od}$ can be directly computed.

This process is described in detail in Section 2.3.3.

Manipulator control. A classical position-based control law is considered to achieve the displacement. The goal is to minimize the error between the desired and current positions of the object expressed in the manipulator reference frame. Therefore the current and desired positions of the effector in this frame have to be known. Since the robot has no proprioceptive sensors, the current position is a priori unknown and has to be computed using an external sensor. Let us just say here that this is done knowing the position of the object in the camera frame (the pose) and by applying some frame transformations (details are given in Section 2.3.2).

We present now an overview of the control algorithm based on the visual servoing approach that allows to achieve the task.

– Acquire the image and track the object (see Section 3.1) ;

– Compute the current pose ${}^{cc}\mathbf{M}_{oc}$ and desired pose ${}^{cc}\mathbf{M}_{od}$ (see Section 3.1) and determine the current and desired positions of the object in the manipulator reference frame (see Section 2.3.2). A first control law expressed in the reference frame can be given.

– Since the robot is controlled in the articular space we need to compute the articular joint positions $\mathbf{q}$. This can be done knowing the object position and the inverse geometrical model of the manipulator (see Section 2.5). A new control law expressed this time in the articular space can then be simply computed (see Section 2.3.2) and given as input to the robot controller ;

– Since the displacement may be important we control the camera orientation in order to always maintain the object in the camera filed of view. We consider a simple visual servoing task to control the pan and tilt d.o.f of the camera (see Section 2.6).

This process is now described in detail.

2.3.2 Visual features and resulting control law

The choice of the visual features (i.e., of the vector $\mathbf{P}$) is very important with respect to the desired properties of the system: stability, robustness, lack of singularities or local minima, adequate trajectories in both the image and articular space.

It is possible to use 2D visual data [Espiau 92, Hutchinson 96] (say, coordinates of points extracted from the images) or 3D data obtained after a pose computation (for example, coordinates of 3D points [Martinet 96] or the six parameters that represent the displacement to achieve [Wilson 96]). Finally, it is possible to combine 2D and 3D visual features: this is 2D 1/2 visual servoing [Chaumette 00].

In our case, since we use an eye-to-hand camera whose orientation is controlled in order to maintain the object centered in the image, the optimal solution is to choose as visual features $\mathbf{P} = ({}^{\mathcal{F}_m}\mathbf{T}_{oc}^T, \theta\mathbf{u}^T)^T$ where ${}^{od}\mathbf{T}_{oc}$ is the translation that the object has to realize (expressed in the final object frame) and where θ and $\mathbf{u}$ are respectively the angle and the rotation axis of ${}^{od}\mathbf{R}_{oc}$. In this case, we have $\mathbf{P_d} = ({}^{\mathcal{F}_m}\mathbf{T}_{od}^T, \mathbf{0}_3^T)^T$. The rotation and the translation motions are thus fully decoupled. Furthermore, if no errors (wrt. measures and calibration) occur, then the object trajectory is a pure straight line as well in the image as in the 3D cartesian space. We thus obtain a better behavior than classical image-based visual servoing.

The equations that link the variation $\dot{\mathbf{P}}$ of the visual features $\mathbf{P}$ to the object velocity in the reference frame are given by:

$$\begin{pmatrix} {}^{\mathcal{F}_m}\dot{\mathbf{T}}_{oc} \\ \dot{\theta\mathbf{u}} \end{pmatrix} = \begin{pmatrix} \mathbf{I\!I}_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & \mathbf{J}_\omega \end{pmatrix} \begin{pmatrix} \mathbf{V} \\ \mathbf{\Omega} \end{pmatrix}_{\mathcal{R}_{\mathcal{F}_m}} \tag{3}$$

with

$$\mathbf{J}_\omega = \mathbf{L}_\omega \, {}^{oc}\mathbf{R}_{\mathcal{F}_m} \tag{4}$$

where $\mathbf{L}_w$ is such that $\mathbf{L}_w^{-1}\theta\mathbf{u} = \theta\mathbf{u}$ [Malis 99, Chaumette 00].

We finally get the following control law :

$$\begin{aligned}\left(\begin{array}{c}\mathbf{V}\\ \mathbf{\Omega}\end{array}\right)_{\mathcal{R}_{\mathcal{F}_m}} &= -\lambda\left(\begin{array}{cc}\mathbf{I\!I}_{3\times3} & 0_{3\times3}\\ 0_{3\times3} & \mathbf{J}_w^{-1}\end{array}\right)(\mathbf{P}-\mathbf{P_d})\\ &= -\lambda\left(\begin{array}{c}{}^{\mathcal{F}_m}\mathbf{T}_{oc}-{}^{\mathcal{F}_m}\mathbf{T}_{od}\\ {}^{\mathcal{F}_m}\mathbf{R}_{oc}\theta\mathbf{u}\end{array}\right)\end{aligned} \tag{5}$$

where the transformations ${}^{od}\mathbf{M}_{oc}$ and ${}^{oc}\mathbf{M}_{\mathcal{F}_m}$, that allow to compute all the values involved in the computation of the control law, are given by:

$${}^{oc}\mathbf{M}_{\mathcal{F}_m} = {}^{oc}\mathbf{M}_{cc}\,{}^{cc}\mathbf{M}_{\mathcal{F}_{pt}}\,{}^{\mathcal{F}_{pt}}\mathbf{M}_{\mathcal{F}_m} \tag{6}$$

$${}^{\mathcal{F}_m}\mathbf{M}_{od} = {}^{\mathcal{F}_m}\mathbf{M}_{\mathcal{F}_{pt}}\,{}^{\mathcal{F}_{pt}}\mathbf{M}_{ci}\,{}^{ci}\mathbf{M}_{od} \tag{7}$$

$${}^{od}\mathbf{M}_{oc} = {}^{od}\mathbf{M}_{cc}\,{}^{cc}\mathbf{M}_{oc} \tag{8}$$

In these equations:

- ${}^{oc}\mathbf{M}_{cc}$ is the pose computed at each iteration using the image processing algorithm ;
- ${}^{od}\mathbf{M}_{cc}$ is also estimated through pose computation (see (10)).

Our goal is to control the manipulator in the articular space. We finally get:

$$\dot{\mathbf{q}} = J^{-1}_{\mathcal{F}_m}(\mathbf{q})\left(\begin{array}{cc}\mathbf{I\!I}_{3\times3} & {}^{\mathcal{F}_m}\mathbf{R}_{ec}\,\widetilde{{}^{e}\mathbf{T}_{o}}\,{}^{ec}\mathbf{R}_{\mathcal{F}_m}\\ 0_{3\times3} & \mathbf{I\!I}_{3\times3}\end{array}\right)\left(\begin{array}{c}\mathbf{V}\\ \mathbf{\Omega}\end{array}\right)_{\mathcal{R}_{\mathcal{F}_m}} \tag{9}$$

where $J^{-1}_{\mathcal{F}_m}$ is the Jacobian matrix that allows to transform velocities expressed in the manipulator reference frame to joint velocities and where $\widetilde{\mathbf{T}}$ is the skew related to vector $\mathbf{T}$.

2.3.3 Definition of the desired position

As already stated, four methods are possible to define the desired position of the object. The user is able to define a displacement in the initial camera frame, in the initial object frame, in the reference frame, and finally as a desired image.

Let us first examine the first three cases. We define by $\Delta\mathbf{T}$, $\Delta\mathbf{R}$ the required displacement, respectively, in translation and rotation. To use the presented control law, we must compute the transformation ${}^{cc}\mathbf{M}_{od}$ that defines the desired position of the object in the current camera

frame. As the current and initial positions of the camera wrt. to its reference frame $\mathcal{F}_{pt}$ are known, the desired position of the object in the current camera frame ${}^{cc}\mathbf{M}_{od}$ is obtained as follows:

$$ {}^{cc}\mathbf{M}_{od} = {}^{cc}\mathbf{M}_{\mathcal{F}_{pt}} {}^{\mathcal{F}_{pt}}\mathbf{M}_{ci} {}^{ci}\mathbf{M}_{od}. \tag{10} $$

We then have to compute the transformation ${}^{ci}\mathbf{M}_{od}$. If the displacement is given in:

- **the initial camera frame,** in that case, we have:

$$ {}^{ci}\mathbf{T}_{od} = {}^{ci}\mathbf{T}_{oi} + \Delta\mathbf{T} \tag{11} $$

$$ {}^{ci}\mathbf{R}_{od} = \Delta\mathbf{R} {}^{ci}\mathbf{R}_{oi} \tag{12} $$

- **the reference frame,** in that case we have:

$$ {}^{ci}\mathbf{T}_{od} = {}^{ci}\mathbf{T}_{oi} + {}^{ci}\mathbf{R}_{\mathcal{F}_m} \Delta\mathbf{T} \tag{13} $$

$$ {}^{ci}\mathbf{R}_{od} = {}^{ci}\mathbf{R}_{\mathcal{F}_m} \Delta\mathbf{R} {}^{\mathcal{F}_m}\mathbf{R}_{oi} \tag{14} $$

- **the initial object frame,** in that case we have:

$$ {}^{ci}\mathbf{M}_{od} = {}^{ci}\mathbf{M}_{oi} \Delta\mathbf{M} \tag{15} $$

  with $\Delta\mathbf{M} = (\Delta\mathbf{R}, \Delta T)$.

Dealing with the last case, the desired position is given as a position to be reached in the desired image. From this position, a pose computation algorithm is used to compute the transformation matrix ${}^{ci}\mathbf{M}_{od}$. From ${}^{ci}\mathbf{M}_{od}$, we then deduce ${}^{cc}\mathbf{M}_{od}$ using the odometry of the pan/tilt head.

2.4 Control of the manipulator velocity

A position-based control does not always appear to be the best way to control a robot. A velocity control is often more intuitive for a human operator especially if he has access to a 3D mouse or another similar device.

2.4.1 Overview of the new visual servoing control law

The control law is no longer specified as a position to reach but as a trajectory to follow. It is then necessary to determine the trajectory that the object, observed by the camera, has to follow and to build a control law that tracks this trajectory and, finally, minimizes the tracking errors.

The idea is to produce a trajectory $\mathbf{P_d}(t)$. As in the previous paragraph, we want to minimize the error:

$$\mathbf{e}(\mathbf{q}(t), t) = \mathbf{P}(\mathbf{q}(t)) - \mathbf{P_d}(t) \tag{16}$$

We got:

$$\dot{\mathbf{e}} = \mathbf{J}\dot{\mathbf{q}} - \frac{\partial \mathbf{P_d}}{\partial t} \qquad \text{with} \qquad \mathbf{J} = \frac{\partial \mathbf{P}}{\partial \mathbf{q}}$$

and the control law is then given by:

$$\dot{\mathbf{q}} = -\lambda \mathbf{J}^{+} \left(\mathbf{P}(\mathbf{q}(t)) - \mathbf{P_d}(t)\right) + \mathbf{J}^{+} \frac{\partial \mathbf{P_d}}{\partial t}$$

In this equation, the second term is used to minimize the tracking errors.

2.4.2 Velocity control law

Since we no longer specify a position to reach $\mathbf{P_d}$ but a trajectory $\mathbf{P_d}(t)$, the position ${}^{\mathcal{F}_m}\mathbf{M}_{of(t)}$ to reach in the reference frame is modified at each iteration. As in the previous case, the error is then defined by:

$$\mathbf{P} - \mathbf{P_d}(\mathbf{t}) = \begin{pmatrix} {}^{\mathcal{F}_m}\mathbf{T}_{oc} - {}^{\mathcal{F}_m}\mathbf{T}_{of(t)} \\ {}^{\mathcal{F}_m}\mathbf{R}_{oc}\theta\mathbf{u} \end{pmatrix} \tag{17}$$

where θ and $\mathbf{u}$ are the angle and the axis of the rotation ${}^{of(t)}\mathbf{R}_{oc}$ and where

$$ {}^{\mathcal{F}_m}\mathbf{M}_{of(t)} = {}^{\mathcal{F}_m}\mathbf{M}_{\mathcal{F}_{pt}} {}^{\mathcal{F}_{pt}}\mathbf{M}_{ci} {}^{ci}\mathbf{M}_{of(t)} \tag{18}$$

$$ {}^{of(t)}\mathbf{M}_{oc} = {}^{of(t)}\mathbf{M}_{ci} {}^{ci}\mathbf{M}_{\mathcal{F}_{pt}} {}^{\mathcal{F}_{pt}}\mathbf{M}_{cc} {}^{cc}\mathbf{M}_{oc} \tag{19}$$

The trajectory to follow is thus function of ${}^{ci}\mathbf{M}_{of(t)}$ (see (18) and (19)) that is updated at each iteration using the velocity specified by the operator. At the beginning, we have of course:

$$ {}^{ci}\mathbf{M}_{of(0)} = {}^{ci}\mathbf{M}_{oi} \tag{20}$$

Then, if the velocity is specified in

- the object frame, ${}^{ci}\mathbf{M}_{of(t+\Delta t)}$ is given by:

$$ {}^{ci}\mathbf{M}_{of(t+\Delta t)} = {}^{ci}\mathbf{M}_{of(t)}\,{}^{of(t)}\mathbf{M}_{of(t+\Delta t)} \tag{21} $$

  with

$$ {}^{of(t)}\mathbf{M}_{of(t+\Delta t)} = \Delta\mathbf{M} = \begin{pmatrix} \Delta\mathbf{R} & \Delta\mathbf{T} \\ \mathbf{0}_{1\times 3} & 1 \end{pmatrix} \tag{22} $$

- in the camera frame, ${}^{ci}\mathbf{M}_{of(t+\Delta t)}$ is given by:

$$ {}^{ci}\mathbf{T}_{of(t+\Delta t)} = {}^{ci}\mathbf{T}_{of(t)} + \Delta\mathbf{T} \tag{23} $$

$$ {}^{ci}\mathbf{R}_{of(t+\Delta t)} = \Delta\mathbf{R}\,{}^{ci}\mathbf{R}_{of(t)} \tag{24} $$

- in the manipulator reference frame, ${}^{ci}\mathbf{M}_{of(t+\Delta t)}$ is given by:

$$ {}^{ci}\mathbf{T}_{of(t+\Delta t)} = {}^{ci}\mathbf{T}_{of(t)} + {}^{ci}\mathbf{R}_{\mathcal{F}_m}\Delta\mathbf{T} \tag{25} $$

$$ {}^{ci}\mathbf{R}_{of(t+\Delta t)} = {}^{ci}\mathbf{R}_{\mathcal{F}_m}\Delta\mathbf{R}^{\mathcal{F}_m}\mathbf{R}_{of(t)} \tag{26} $$

In these equations we have

$$ \begin{aligned} \Delta\mathbf{T} &= \mathbf{V}\Delta t \\ \Delta\mathbf{R} &= \cos(\omega\Delta t)\mathbf{I}_{3\times 3} + (1-\cos(\omega\Delta t))\mathbf{v}.\mathbf{v}^T + \sin(\omega\Delta t)\tilde{\mathbf{v}} \end{aligned} $$

where $\mathbf{V}$ is the specified translation, $\mathbf{v}$ is the axis of the rotation and ω is the velocity of this rotation around $\mathbf{v}$. All these value are expressed in the frame specified by the operator. Δt is the rate of the closed-loop. In the reminder we note $\Omega = \omega\mathbf{v}$.

The final control law is given by:

$$ \dot{\mathbf{q}} = J^{-1}_{\mathcal{F}_m}(\mathbf{q}) \begin{pmatrix} \mathbb{I}_{3\times 3} & {}^{\mathcal{F}_m}\mathbf{R}_{ec}\,\widetilde{{}^{e}\mathbf{T}_{o}}\,{}^{ec}\mathbf{R}_{\mathcal{F}_m} \\ 0_{3\times 3} & \mathbb{I}_{3\times 3} \end{pmatrix} \cdot \left(-\lambda\left(\mathbf{P} - \mathbf{P_d}(\mathbf{t})\right) + \begin{pmatrix} \mathbf{V} \\ \mathbf{\Omega} \end{pmatrix}_{\mathcal{R}_{\mathcal{F}_m}} \right) \tag{27} $$

To use properly this equation, it must be noted that the term $(\mathbf{V}, \mathbf{\Omega})^T$ is expressed in the reference frame. Since the operator may specify the desired velocity in 3 different frames, a frame manipulation may be necessary to express $(\mathbf{V}, \mathbf{\Omega})^T$ in $\mathcal{R}_{\mathcal{F}_m}$. If the velocity is initially expressed in:

- the camera frame

$$\begin{pmatrix} \mathbf{V} \\ \mathbf{\Omega} \end{pmatrix}_{\mathcal{R}_{\mathcal{F}_m}} = \begin{pmatrix} {}^{\mathcal{F}_m}\mathbf{R}_{cc} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & {}^{\mathcal{F}_m}\mathbf{R}_{cc} \end{pmatrix} \begin{pmatrix} \mathbf{V} \\ \mathbf{\Omega} \end{pmatrix}_{\mathcal{R}_{cc}} \tag{28}$$

- in the object frame :

$$\begin{pmatrix} \mathbf{V} \\ \mathbf{\Omega} \end{pmatrix}_{\mathcal{R}_{\mathcal{F}_m}} = \begin{pmatrix} {}^{\mathcal{F}_m}\mathbf{R}_{oc} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & {}^{\mathcal{F}_m}\mathbf{R}_{oc} \end{pmatrix} \begin{pmatrix} \mathbf{V} \\ \mathbf{\Omega} \end{pmatrix}_{\mathcal{R}_{oc}} \tag{29}$$

- in the reference frame, we directly have $\begin{pmatrix} \mathbf{V} & \mathbf{\Omega} \end{pmatrix}^T_{\mathcal{R}_{\mathcal{F}_m}}$

2.5 Computing articular positions

In both position-based and velocity-based control, joints value $\mathbf{q}$ of the manipulator have to be estimated (see (9) and (27)). However, as already stated, we do not have a direct access to these values since the robot is not instrumented.

To compute the joints positions $\mathbf{q}$ we use the position of the effector in the manipulator reference frame ${}^{\mathcal{F}_m}\mathbf{M}_{ec}$ and the inverse geometrical model $f^{-1}(.)$ of the manipulator. We get:

$$\mathbf{q} = f^{-1}({}^{\mathcal{F}_m}\mathbf{M}_{ec}) \tag{30}$$

where ${}^{\mathcal{F}_m}\mathbf{M}_{ec}$ is estimated knowing the pose by:

$${}^{\mathcal{F}_m}\mathbf{M}_{ec} = {}^{\mathcal{F}_m}\mathbf{M}_{\mathcal{F}_{pt}} {}^{\mathcal{F}_{pt}}\mathbf{M}_{cc} {}^{cc}\mathbf{M}_{oc} {}^{oc}\mathbf{M}_{ec} \tag{31}$$

We have considered in this paper only the case of a non-redundant robot that ensures a finite number of solutions for the inverse robot geometrical model. However, more than one solution can be found for the inverse geometrical model. In the current implementation of our system, if this problem appears at the initialization, the user has to choose the closest to the real one. If this appears during the visual servoing control loop, the solution the closest from the previous one is automatically selected.

2.6 Pan/Tilt control

It is important to control the camera pan and tilt in order to ensure that the end effector of the manipulator remains in the camera field of view. To achieve this task we simply use the 2D visual servoing approach [Espiau 92]. We define as visual features the projection of the center of gravity of the target: $\mathbf{P} = (X, Y)^T$ and we control the camera in order to see it centered in the image: $\mathbf{P_d} = (X_d, Y_d)^T = (0, 0)^T$.

The image Jacobian related to the task is given by:

$$\mathbf{L} = \begin{pmatrix} XY & -(1+X^2) \\ 1+Y^2 & -XY \end{pmatrix} \tag{32}$$

We therefore have

$$\mathbf{e} = \mathbf{L}^+ (\mathbf{P} - \mathbf{P_d}) = \begin{pmatrix} \frac{Y}{1+X^2+Y^2} \\ -\frac{X}{1+X^2+Y^2} \end{pmatrix}$$

and the control law at each iteration k is given by:

$$\mathbf{T}_k = -\lambda_2 \mathbf{e}_k - \left(\widehat{\frac{\partial \mathbf{e}}{\partial t}}\right)_k \tag{33}$$

The second term is due to the target motion. In order to attenuate the tracking error, and assume a constant velocity for the manipulator end-effector, an adequate estimation of $\widehat{\frac{\partial \mathbf{e}}{\partial t}}$ is given by a simple integrator [Chaumette 91]:

$$\left(\widehat{\frac{\partial \mathbf{e}}{\partial t}}\right)_k = -\mu \sum_{j=0}^{k} \mathbf{e}_k \tag{34}$$

3 Experimental results

3.1 Image Processing and pose computation

The image processing algorithm is required to be fast and robust. To achieve these goals we propose a simple but efficient tracking algorithm that relies both on the tracking of 2D features and the estimation of the 3D position of the object in the camera frame. As the end-effector of the Victor 6000 manipulator is cylindrical, tracking the target (the object) raised many problems.

This target is made of white dots on a black background and we assume that the CAD model of this target is fully known (see for example Figure 3). Due to the end-effector cylindrical shape, all the landmarks cannot be seen at the same time. Appearance/disappearance of dots must then be handled by the algorithm.

We give here a brief description of this algorithm. One iteration of this algorithm includes the image acquisition and its processing.

Initialization in the very first image (iteration 0). In the current version of the system described in this paper, initialization of the tracking in the very first image of the sequence is performed partly manually. This means that the user has to click at least four points on both the initial image and the CAD model of the object. This is achieved within an interactive procedure ensuring also the matching between the selected model points and their corresponding projections in the images located by the user.

Description of iteration i. We suppose that at the end of the iteration $i-1$, a set P^{i-1} of N ($N \geq 4$) points are tracked in the images: $P^{i-1} = \{P_1^{i-1}, \dots P_N^{i-1}\}$ and that we know the 3D coordinates of each point in the target frame $p^{i-1} = \{p_1^{i-1}, \dots p_N^{i-1}\}$.

The first step consists in a 2D tracking of the point between frame $i-1$ and i. To achieve this task, we assume that the image target motion is small and we use a classical algorithm to compute the center of gravity of the dot. This can be done since tracking is performed in real time and the target is moving slowly. However, some points may be lost due to motion which is too large or an occlusion by other objects or by the target itself. We therefore get a list P^i of M points with $M \leq N$.

From these M points we compute the pose. A number of methods have been proposed to compute pose from points. We have used the method designed by Dementhon [Dementhon 95] completed by Lowe's non-linear method [Lowe 92]. Dementhon's method calculates the rigid transformation in an iterative way from the knowledge of the coordinates of at least four points in the object coordinate system, and of their corresponding projections in the image. Its principle consists of approximating perspective projection by scaled orthographic projec-

tion, and then iteratively modifying the scaled orthographic projection to converge upon the perspective projection. We then apply the method proposed by Lowe to improve the pose estimation: Lowe's approach is based on an iterative minimization of a residual using the non linear Levenberg-Marquardt minimization technique.

Once the pose ${}^{cc}\mathbf{M}_{oc}$ is available, we can easily determine visible and invisible points of the target and add new points in the list P^i on a prediction/verification basis.

3.2 Control experiments

Experiments have been carried out on a 6 dof cartesian robot at INRIA Rennes. Control and image processing are performed on a PC Linux (Pentium II, 366Mhz). Unlike the Victor 6000 manipulator, this robot is fully instrumented and the odometry is available. We will use this knowledge to compare the displacement achieved using measured $\mathbf{q}$ and using estimated $\mathbf{q}$. It will also be used to compare the specified displacement and the real one.

Figure 3 shows four images of the object mounted on the manipulator end-effector acquired in a typical run of our algorithm. Green lines represent the virtual links between the current and desired position of the landmark in the image. As can be seen the initial desired position is not (necessarily) in the image, however as the camera is controlled in pan and tilt to center the object, this desired position is moving in the images over time (see also Figure 4).

In all the reported experiments, in order to get a faster convergence of the control law, we considered for λ (see (2)) an adaptive gain function of the error $\mathbf{P} - \mathbf{P_d}$.

3.2.1 Displacement specified as an image to reach

It is possible to specify the desired position of the manipulator in the image. Using this image, the desired pose ${}^{ci}\mathbf{M}_{od}$ and consequently ${}^{\mathcal{F}_m}\mathbf{M}_{od}$ can be computed. This is a "learning" step. The camera and/or the manipulator is/are then moved to another location and then servo back to the learnt position.

If the camera calibration parameters used for the learning step and the servoing step are the same, then no errors in the positioning process are observed (see Table 1). Indeed

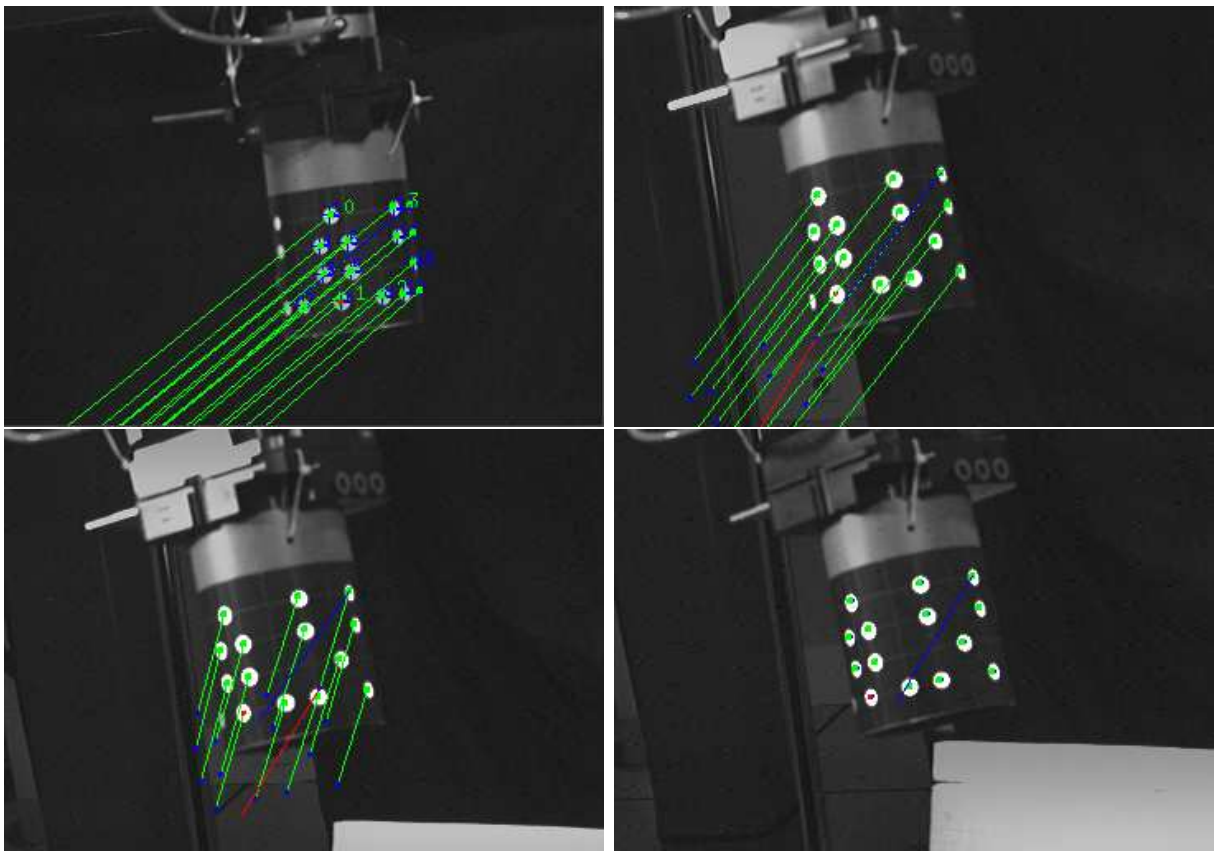

Figure 3: Target tracking in an image sequence and control of both the manipulator and the pan/tilt camera.

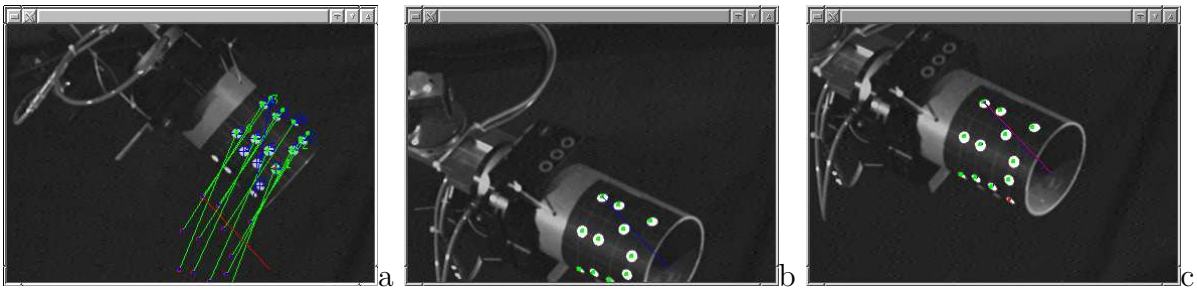

Figure 4: Effect of the pan/tilt control: the desired position of the object in the image is modified. (a) initial position, (b) desired position with no pan/tilt control, (c) desired position with pan/tilt control.

poses are computed using the same calibration parameters. If these parameters are wrong and although the poses are incorrect the relative displacement achieved is correct. Even with very bad camera calibration parameters, precision remains very good as long as the resolution of the inverse geometrical model remains possible. Errors are then less than 5mm in translation and one degree in rotation and are due to the rough calibration of the pan/tilt system and to the lack of precision of the pose computation.

Desired position in $\mathcal{R}_{\mathcal{F}_m}$	T_x	T_y	T_z	θ_x	θ_y	θ_z
desired	154	44	-98	29.3	6.9	33.6
actual	150	47	-100	29.3	6.1	33.2
actual (calibration -20%)	155	52	-103	29.8	3.2	33.0
actual (calibration +20%)	155	45	-100	29.3	6.7	33.6
actual (calibration +40%)	155	45	-99	29.3	6.9	33.5

Table 1: Displacement specified in the image (the desired and actual positions are computed using the robot odometry)

3.2.2 Position-based control

We now present results dealing with displacement within the various possible frames The examples reported in Table 2 feature simple displacements, specified in the object initial frame, by translation, another by rotation, as well as a complex displacement of every axes. Plots on Figure 6 show the behavior of the algorithm over time (error, control law, and achieved displacement).

In each case, a small bias between the desired displacement and the actual one can be observed (mainly in the translation displacement). This bias is due to calibration errors in the camera parameters (as can be seen the amplitude of the bias is correlated to the error introduced in the camera parameters), but also to errors in the initial estimation of transformation ${}^{\mathcal{F}_m}\mathbf{M}_{\mathcal{F}_{pt}}$ and ${}^{e}\mathbf{M}_{o}$, and measurement errors in the pose computation. Calibration of the full system is a problem of its own that is not in the scope of this paper.

Moreover if important errors are introduced in the camera parameters (typically over 40%), the resulting computed pose and therefore of the object position ${}^{\mathcal{F}_m}\mathbf{M}_{e}$ may be nonsensical (e.g., out of the joint-limits) and the inverse geometrical model may be unsolvable.

In that case, visual servoing will fail.

Dealing with the online estimation of the articular position $\mathbf{q}$, results show that the errors in this estimation (due to calibration and measure errors in ${}^{\mathcal{F}_m}\mathbf{M}_{\mathcal{F}_{pt}}$, ${}^{cc}\mathbf{M}_{oc}$ and ${}^{e}\mathbf{M}_{o}$) have no effect on the achieved displacement. Indeed, the achieved displacements considering measured $\mathbf{q}$ or estimated $\mathbf{q}$ are very similar (see table 2).

Figure 5 shows the effect of errors in the calibration of the camera on the projection of the desired position. We considered here errors of $\pm 40\%$ wrt. to the initial parameters (which are also certainly wrong since the camera has not been calibrated). The desired position is therefore very different, and the actual displacement will be therefore very dependent of these parameters as can be seen in the various tables.

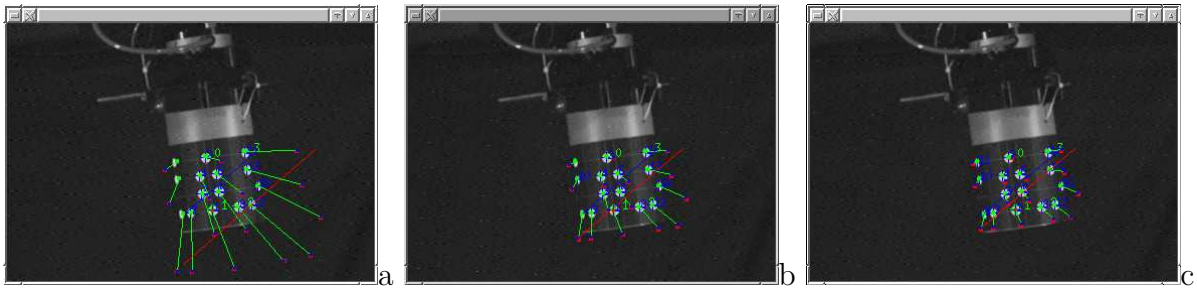

Figure 5: Effect of the calibration errors on the desired position. The specified displacement is a translation of -500mm along the camera optical axis. (a) initial camera parameters with a noise of -40 %, (b) initial camera parameters (c) initial camera parameters with a noise of +40 %

3.2.3 Velocity control

Finally, experiments dealing with velocity control have been carried out. In this experiment, the required displacement was specified in the reference frame. The specified trajectory is a rectangle (that is translation along x and y axes) followed by a rotation around the object axis (the object should not translate during this step). This trajectory is then iterated a few time.

Figure 7 shows the trajectory achieved by the object in the reference frame. As expected the trajectory is correctly achieved minus the small errors due to the system calibration errors (as stated in the previous paragraph, these errors are mainly due to the rough estimation

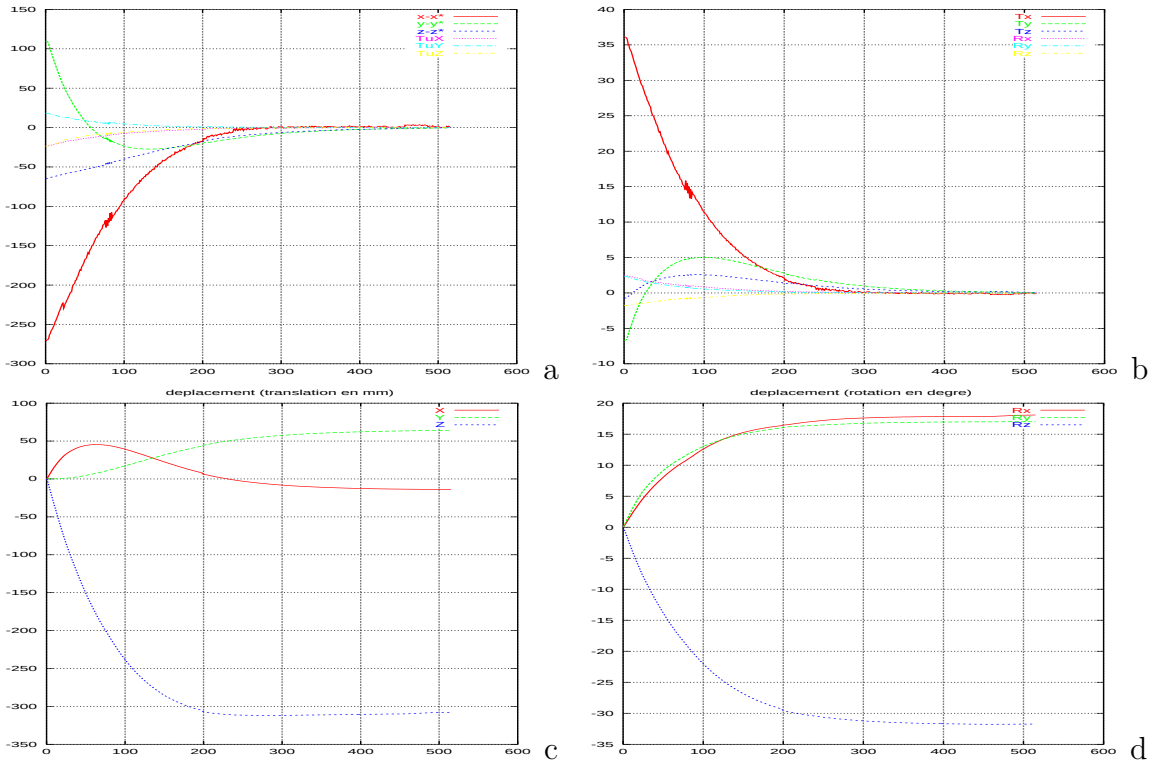

Figure 6: Results of a displacement specified in the initial camera frame $\Delta\mathbf{T} = (50, 50, -300)$, $\Delta\mathbf{R} = (20, 15, -30)$. All the plots are over time. (a) error $\mathbf{P} - \mathbf{P_d}$, (b) control law (c) displacement in translation (mm), (d) displacement in rotation (deg).

Δ in $\mathcal{R}_{oi}$	T_x	T_y	T_z	θ_x	θ_y	θ_z
desired	0	0	0	0	0	30
actual	-8.8	-0.5	-1.7	-0.1	-0.3	31.6
actual with measured $\mathbf{q}$	-8.7	-0.9	-1.6	-0.1	-0.3	31.8
actual (calibration +20%)	0.9	0.0	1.2	0.2	1.3	30.8
actual (calibration +40%)	3.7	-0.1	1.0	0.2	1.1	30.8
actual calibration -20%	-3.5	-0.4	1.4	0.2	2.0	31.7
desired	300	0	0	0	0	0
actual	291.7	48.8	-1.0	0.0	0.3	1.9
actual with measured $\mathbf{q}$	290.6	49.0	0.5	0.1	0.1	2.0
actual calibration +20%	253.8	41.2	-5.5	0.5	2.1	-0.8
actual calibration +40%	220.4	39.6	-4.6	0.4	1.7	-1.1
actual calibration -20%	371.6	42.9	-6.6	0.5	3.6	0.5
desired	300	50	50	20	20	60
actual	281.0	102.7	56.9	15.4	23.4	64.3
actual with measured $\mathbf{q}$	281.1	102.3	57.1	15.4	23.5	64.1
actual calibration +20%	255.0	97.9	51.3	17.3	27.1	59.2
actual calibration -20%	362.3	96.0	49.9	15.5	28.9	66.9
actual (PT)	283.0	94.7	69.1	15.6	24.4	64.3
actual calibration +20% (PT)	254.6	78.9	41.3	16.9	26.8	59.8
actual calibration -20% (PT)	373.4	97.9	78.9	16.2	29.9	61.5

Table 2: Displacement in the initial object frame

of the ${}^{\mathcal{F}_{pt}}\mathbf{M}_{\mathcal{F}_m}$ matrix). The rotation achieved at the end of the rectangle trajectory is correctly handled (i.e., no translational motion at this point, see the bottom left corner of Figure 7.a). When a new velocity is given, due to tracking errors, a few iterations are necessary to correctly achieve the newly specified velocity. The term $\frac{\partial \mathbf{P_d}}{\partial t}$ introduced in the control law allows to suppress quickly these tracking errors (see Figure 7.c for the computed velocity and 7.d for the actual measured velocity).

3.3 Peg-in-hole experiments

In this last experiment we consider both position-based and velocity control. The goal is to insert the cylinder of the robot end-effector on another cylinder in order to achieve a "peg-in-hole" task. A position-based control task is first considered to move the target on the second cylinder axis. A velocity control (motion along the z axis in the target frame) is then performed to achieve the insertion. The cylinder on the end effector has a radius of 5

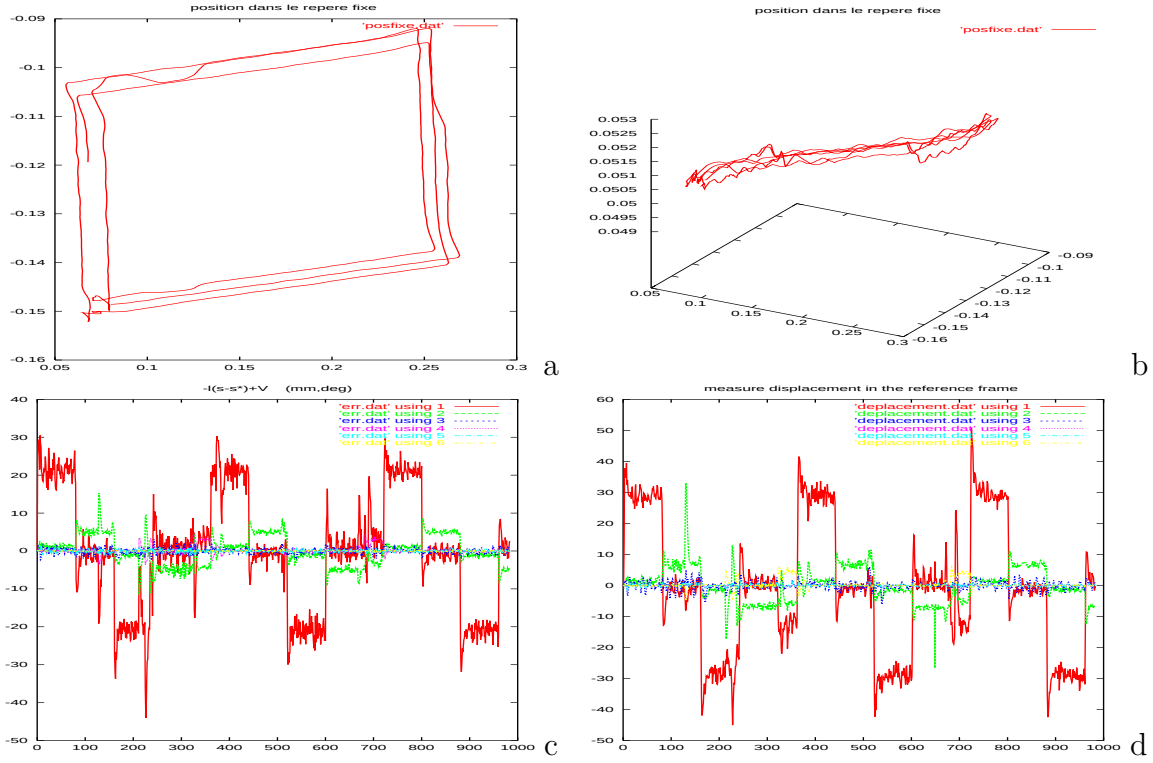

Figure 7: Velocity control : (a-b) Position of the object in the reference frame ((a) X-Y view (b) X-Y-Z view) (c) Velocity sent to the robot controller (d) Measured velocity

cm while the other has a radius of 4 cm. The experiment is reported on Figure 8.

3.4 Remark

Currently this algorithm has not been tested in underwater condition on the Victor 6000. Indeed to achieve this goal, more robust image processing has to be developed to take into account the effects of the water medium. Particularly, we have to deal with specific optical laws when using underwater cameras. Specific camera calibration and pose computation algorithms may have to be considered as in [Lavest 00]. However, classical perspective models have proved to be an effective solution to robot control in an underwater environment [Rives 97, Lots 00, Lots 01, Van Der Zwaan 01]. Furthermore, since visual servoing is robust wrt modeling errors, we have not considered this point.

4 Conclusion

In this paper we proposed a complete framework to control in position and in velocity the effector of a manipulator using a vision-based control approach. On the practical side, the primary interest is to use the visual information provided by the camera to efficiently control a manipulator that was not instrumented by proprioceptive sensors (no odometry). A second interest lies in the independence of the position reached by the arm compared to the various modeling errors and calibration errors, of the sensor as well as the arm if it is possible to express the task assigned with the manipulator directly in the space of measurement of the sensor (in fact the image plane of the camera). To achieve the manipulator control, the value of the manipulator joints are estimated on-line and the specified displacement is achieved using a 3D visual servoing control law. The orientation of the camera is also controlled by 2D visual servoing so that the effector always remains in the camera field of view. Because of the camera motion, it is appropriate to measure this motion (measured by odometry) and to compute at each iteration of the control the position to reach or the adequate velocity to follow. Experiments have been carried out on a 6 dof robot and show the validity and the efficiency of our approach.

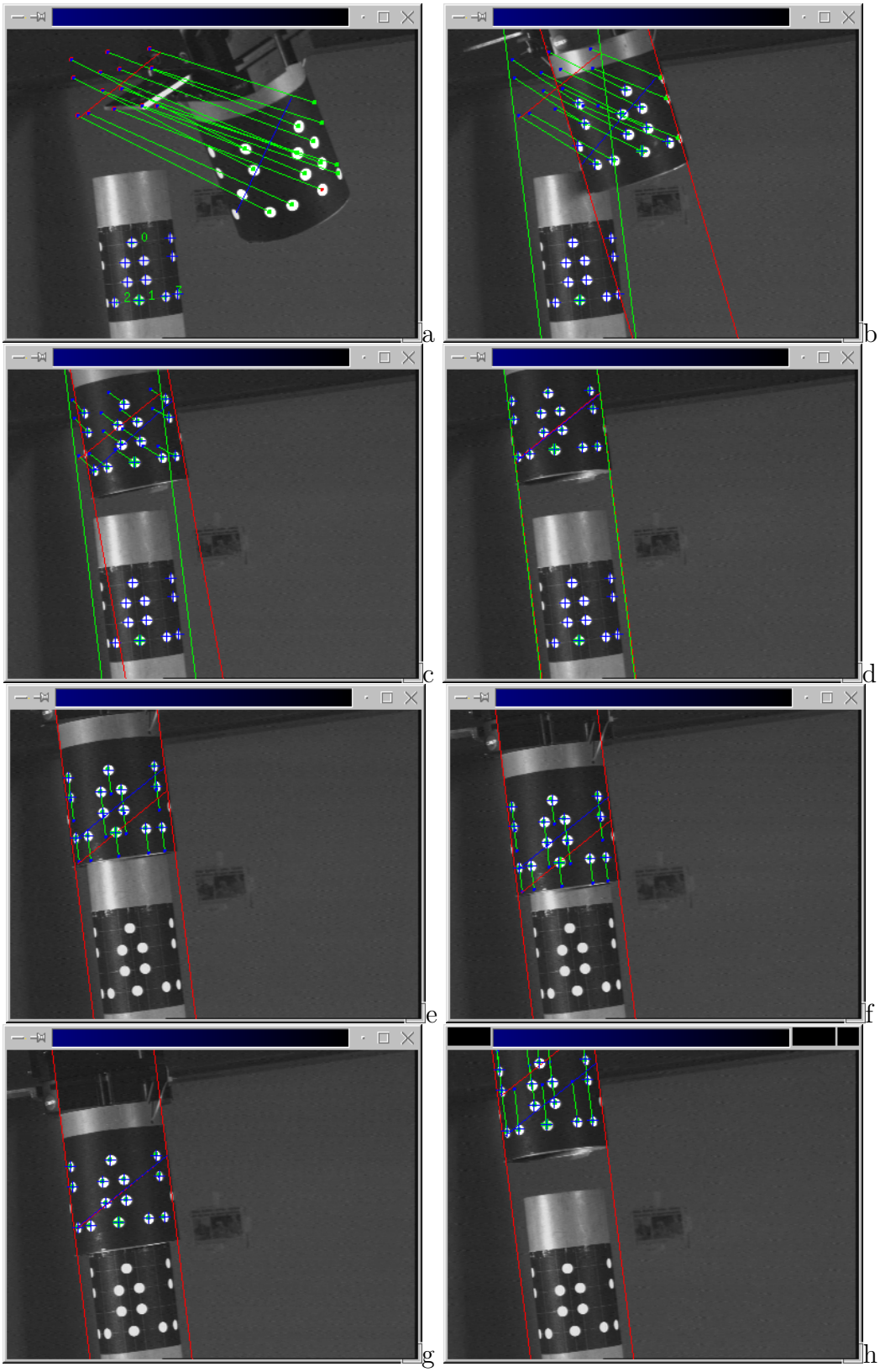

Figure 8: Insertion task: the goal is to insert the cylinder on robot end-effector onto another cylinder. (a-d) Alignment of the cylinders axes (position-based control), (e-g) cylinder insertion (velocity control), (g) removing the cylinder

Acknowledgment

This work has been supported by Ifremer under contract INRIA/Ifremer 1.00.C.001 and INRIA/Ifremer 1.00.C.606.

References

[Allen 93] P.K. Allen, A. Timcenko, B. Yoshimi, P. Michelman. – Automated tracking and grasping of a moving object with a robotic hand-eye system. *IEEE Trans. on Robotics and Automation*, 9(2):152–165, April 1993.

[Chaumette 91] F. Chaumette, P. Rives, B. Espiau. – Positioning of a robot with respect to an object, tracking it and estimating its velocity by visual servoing. – *IEEE Int. Conf. on Robotics and Automation*, vol. 3, pp. 2248–2253, Sacramento, California, USA, April 1991.

[Chaumette 00] F. Chaumette, E. Malis. – 2 1/2 d visual servoing: a possible solution to improve image-based and position-based visual servoings. – *IEEE Int. Conf. on Robotics and Automation*, vol. 1, pp. 630–635, San Francisco, CA, April 2000.

[Dementhon 95] D. Dementhon, L. Davis. – Model-based object pose in 25 lines of codes. *Int. J. of Computer Vision*, 15:123–141, 1995.

[Espiau 92] B. Espiau, F. Chaumette, P. Rives. – A new approach to visual servoing in robotics. *IEEE Trans. on Robotics and Automation*, 8(3):313–326, June 1992.

[Hager 95] G. Hager, W. Chang, A.S. Morse. – Robot feedback control based on stereo vision: Towards calibration-free hand-eye coordination. *IEEE Control Systems Magazine*, 15(1):30–39, February 1995.

[Hashimoto 93] K. Hashimoto. – *Visual Servoing : Real Time Control of Robot Manipulators Based on Visual Sensory Feedback.* – World Scientific Series in Robotics and Automated Systems, Vol 7, World Scientific Press, Singapor, 1993.

[Horaud 98] R. Horaud, F. Dornaika, B. Espiau. – Visually guided object grasping. *IEEE Trans. on Robotics and Automation*, 14(4):525–532, August 1998.

[Hutchinson 96] S. Hutchinson, G. Hager, P. Corke. – A tutorial on visual servo control. *IEEE Trans. on Robotics and Automation*, 12(5):651–670, October 1996.

[Lavest 00] J.-M. Lavest, G. Rives, J.-T. Laprest. – Underwater camera calibration. – *European Conference on Computer Vision, ECCV'00*, vol. 2, pp. 654–668, Dublin, Ireland, June 2000.

[Lots 00] J.-F. Lots, D. Lane, E. Trucco. – Application of 2 1/2 d visual servoing to underwater vehicle station-keeping. – *MTS/IEEE Oceans 2000 Conference proceedings*, volume 2, pp. 1257-1264, Providence, Rhode Island, September 2000.

[Lots 01] J.-F. Lots, D. Lane, E. Trucco, F. Chaumette. – A 2-d visual servoing for underwater vehicle station keeping. – *IEEE Int. Conf. on Robotics and Automation*, Seoul, Korea, May 2001.

[Lowe 92] D.G. Lowe. – Robust model-based motion tracking through the integration of search and estimation. *Int. J. of Computer Vision*, 8(2):113–122, 1992.

[Malis 99] E. Malis, F. Chaumette, S. Boudet. – 2 1/2 D Visual servoing. – *IEEE Trans. on Robotics and Automation*, 15(2):238-250, apil 1999.

[Martinet 96] P. Martinet, J. Gallice, D. Khadraoui. – Robot control using 3d visual features. – *Word Automation Congress, WAC'96*, vol. 3, pp. 497–502, Montpellier, May 1996.

[Nokin 97] M. Nokin. – Victor 6000, a deep teleoperated system for scientific research. – *MTS/IEEE OCEANS conference proceedings*, volume 1, pp.167-171 Halifax, Canada, October 1997.

[Papanikolopoulos 93] N.P. Papanikolopoulos, P.K. Khosla, T. Kanade. – Visual tracking of a moving target by a camera mounted on a robot: A combination of control and vision. *IEEE Trans. on Robotics and Automation*, 9(1):14–35, February 1993.

[Rives 97] P. Rives, J.-J. Borrelly. – Visual servoing techniques applied to an underwater vehicle. – *IEEE Int. Conf. on Robotics ans Automation*, vol. 3, pp. 1851–1856, Albuquerque, USA, April 1997.

[Samson 91] C. Samson, M. Le Borgne, B. Espiau. – *Robot Control: the Task Function Approach*. – Clarendon Press, Oxford, United Kingdom, 1991.

[Van Der Zwaan 01] S. Van Der Zwaan, J. Santos-Victor. – Real-time vision-based station keeping for underwater robots. – *MTS/IEEE OCEANS conference proceedings*, volume 2, pp. 1058-1065, Honolulu, November 2001.

[Wilson 96] W. Wilson, C. Hulls, G. Bell. – Relative end-effector control using cartesian position-based visual servoing. *IEEE Trans. on Robotics and Automation*, 12(5):684–696, October 1996.