MTS/IEEE Oceans 2001, Honolulu, Hawaii, November 2001

Controlling an uninstrumented ROV 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 vision-based method to control the displacement of robot arm mounted on an underwater ROV. A closed-loop system based on an eye-to-hand visual servoing approach has been designed to achieve this task. We show that, using such an approach, measuring the manipulator motion with proprioceptive sensors is not required to precisely control the end-effector motion. To maintain the end effector in the field of view, the camera orientation is also controlled. Presented results show the validity of the approach.**

I. Introduction

In this paper we present a vision-based method to control the manipulator of the Victor 6000 ROV. Victor 6000 [11] is a deep underwater 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 and to carry out local assignments for imagery, implementing instrumentation and sampling water, sediments or rocks.

Fig. 1. The Ifremer Victor 6000 underwater ROV (© Ifremer)

Victor 6000 is equipped with with two manipulators: a 6 dof manipulator called Maestro and a 4 dof manipulator called Sherpa. The Sherpa manipulator is not instrumented and is open-loop controlled with a joystick. Due to the lack of proprioceptive sensors, the odometry, and in particular the joints positions $\mathbf{q}$ are not available. Therefore there is no way to measure the manipulator motion and any control will be imprecise if no external sensor is used to provide a closed-loop system.

To cope with this problem we consider the control of the manipulator within the visual servoing framework. Visual servoing has proved to be a very efficient method to control manipulator in hostile environments. Dealing with underwater robotics, eye-in-hand visual servoing has been used to control Remote Operated Vehicle (e.g., [12], [8], [7]). Our goal is not to control the ROV itself, but to control the motion of its non-instrumented manipulator using informations provided by a camera mounted on a pan/tilt head mounted on the ROV and observing the end-effector of the manipulator. In this paper we show that the measurement using proprioceptive sensors is not required to precisely control the end-effector motion and that the approach is quite robust to calibration errors wrt. to the camera and the system. Furthermore, from the end-user point of view it is not realistic to consider a static camera. Indeed the defined manipulator motions may allow the end-effector to move outside the image and, in that case, control will fail. It is therefore 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.

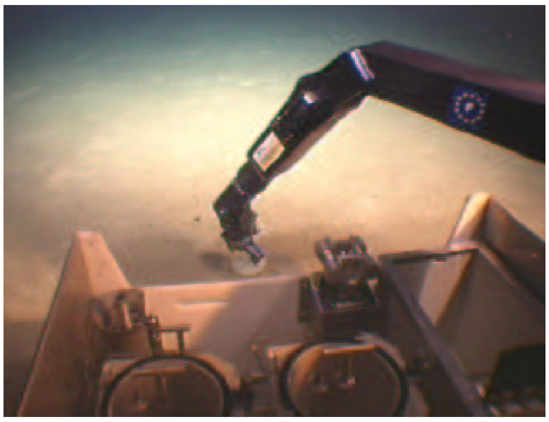

Fig. 2. The Victor 6000 six dof manipulator Maestro (© Ifremer)

The reminder of this paper is organized as follows: in a first section we present 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 we describe the image processing algorithm ; finally we show on various experimental results the efficiency of our approach.

II. Image-based control

A. Notation

Let us define 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.

More precisely, the frames used in this paper are represented on Figure 3. The first letter represents the origin of the frame (c for camera, e for effector and o for object) 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 head. For example ${}^{ci}\mathbf{M}_{od}$ defines the desired position of the object in the initial camera frame.

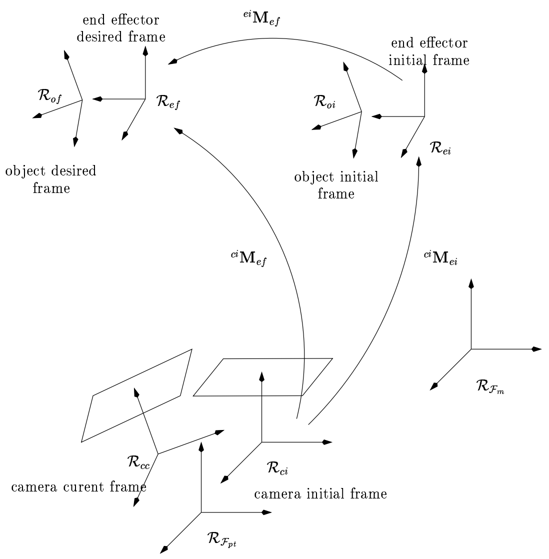

Fig. 3. Overview of the various frames

B. Estimating various transformation

In the reminder of this paper, we will refer to many frame transformations. Prior to a good behavior of the proposed algorithms, some transformations 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 hand-measured ;
- ${}^{ei}\mathbf{M}_{oi} = {}^{ec}\mathbf{M}_{oc} = {}^{ed}\mathbf{M}_{od} = {}^{e}\mathbf{M}_o$ is hand-measured ;
- ${}^{\mathcal{F}_{pt}}\mathbf{M}_c$ is measured using the pan/tilt head odometry.

C. Visual servoing: overview

Visual servoing technics [4], [6], [5] allow to automatically position a robot with respect to its environment using visual data. It consists in specifying a task as the regulation of a set of informations extracted from the images [4], [5].

In our case, a vision-based task $\mathbf{e}$ is defined by [4], [13]:

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

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.

D. Control manipulator position

D.1 Overview of the algorithm

The goal for the manipulator is to achieve the displacement specified by the ROV operator. Two methods are available to specify this desired displacement:
- a direct definition of the desired manipulator displacement ΔT, ΔR in, possibly, three different frames $\mathcal{R}_{ci}, \mathcal{R}_{oi}, \mathcal{R}_{\mathcal{F}_m}$.
- a definition in the image space. It corresponds to reach again a position that has been learned in an off-line learning step.

We present in this section an overview of the control algorithm that allows to achieve this task.

1. In a first time we have to compute the initial position of the object in the initial camera frame (i.e., the initial pose ${}^{ci}\mathbf{M}_{oi}$). This is done using the proposed image processing algorithm (see Section III-A). The joint position $\mathbf{q}$ is then computed (see Section II-F). If more than one solution are found for the inverse geometrical model, the user has to choose the most convincing one.
2. From the specified displacement, we determine the desired object position in the initial camera frame ${}^{ci}\mathbf{M}_{od}$ (see Section II-D.3).
3. A visual servoing closed loop is then used to reach the desired position:
 – Acquire the image and track the object (see Section III-A) ;
 – Compute the current pose ${}^{cc}\mathbf{M}_{oc}$ and desired pose ${}^{cc}\mathbf{M}_{od}$ (see Section 3) ;
 – Compute the articular joint positions $\mathbf{q}$ (see Section 2.3.3) ;
 – Compute the control law for the manipulator (see Section 2.3.2) ;
 – Compute the control law for the pan/tilt camera (see Section II-G).

This process is now described in details.

D.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 [4], [6] (say, coordinates of points extracted from the images) or 3D data obtained after a pose computation (for example, coordinates of 3D points [10] or the six parameters that represent the displacement to achieve [14]). Finally, it is possible to combine 2D and 3D visual features: this is 2D 1/2 visual servoing [1].

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 and position-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:

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

with

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

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

We finally get the following control law :

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

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:

$$\begin{aligned} {}^{oc}\mathbf{M}_{\mathcal{F}_m} &= {}^{oc}\mathbf{M}_{cc}{}^{cc}\mathbf{M}_{\mathcal{F}_{pt}}{}^{\mathcal{F}_{pt}}\mathbf{M}_{\mathcal{F}_m} & (6) \\ {}^{\mathcal{F}_m}\mathbf{M}_{od} &= {}^{\mathcal{F}_m}\mathbf{M}_{\mathcal{F}_{pt}}{}^{\mathcal{F}_{pt}}\mathbf{M}_{ci}{}^{ci}\mathbf{M}_{od} & (7) \\ {}^{od}\mathbf{M}_{oc} &= {}^{od}\mathbf{M}_{cc}{}^{cc}\mathbf{M}_{oc} & (8) \end{aligned}$$

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

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

$$\dot{\mathbf{q}} = J_{\mathcal{F}_m}^{-1}(\mathbf{q}) \begin{pmatrix} \mathbb{I}_{3\times3} & {}^{\mathcal{F}_m}\mathbf{R}_{ec}\widetilde{{}^{e}\mathbf{T}_o}{}^{ec}\mathbf{R}_{\mathcal{F}_m} \\ 0_{3\times3} & \mathbb{I}_{3\times3} \end{pmatrix} \begin{pmatrix} \mathbf{V} \\ \mathbf{\Omega} \end{pmatrix}_{\mathcal{R}_{\mathcal{F}_m}} \tag{9}$$

where $J_{\mathcal{F}_m}^{-1}$ 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 $\mathbf{T}$.

D.3 Reaching the desired position

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

Let us first examine the three former cases. We define by $\Delta\mathbf{T}, \Delta\mathbf{R}$ the required displacement in, respectively, 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 ${}^{ci}\mathbf{M}_{od}$. The displacement is given in:

- **the initial camera frame.** In that case we have:

$$\begin{aligned} {}^{ci}\mathbf{T}_{od} &= {}^{ci}\mathbf{T}_{oi} + \Delta\mathbf{T} & (11) \\ {}^{ci}\mathbf{R}_{od} &= \Delta\mathbf{R}{}^{ci}\mathbf{R}_{oi} & (12) \end{aligned}$$

- **the reference frame.** In that case we have:

$$\begin{aligned} {}^{ci}\mathbf{T}_{od} &= {}^{ci}\mathbf{T}_{oi} + {}^{ci}\mathbf{R}_{\mathcal{F}_m}\Delta\mathbf{T} & (13) \\ {}^{ci}\mathbf{R}_{od} &= {}^{ci}\mathbf{R}_{\mathcal{F}_m}\Delta\mathbf{R}{}^{\mathcal{F}_m}\mathbf{R}_{oi} & (14) \end{aligned}$$

- **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, it is possible using a pose computation algorithm to compute the transformation matrix ${}^{ci}\mathbf{M}_{od}$. We then deduce ${}^{ci}\mathbf{M}_{od}$ using the odometry of the pan/tilt head.

E. Control manipulator velocity

E.1 Overview

The control law is no longer specified as a position to reach but as a velocity to follow. It is then necessary to determine the trajectory in the image space that the effector, observed by the camera, has to follow and to build a control law that tracks this trajectory and 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:

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

E.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

$$\begin{aligned} {}^{\mathcal{F}_m}\mathbf{M}_{of(t)} &= {}^{\mathcal{F}_m}\mathbf{M}_{\mathcal{F}_{pt}}{}^{\mathcal{F}_{pt}}\mathbf{M}_{ci}{}^{ci}\mathbf{M}_{of(t)} & (18) \\ {}^{of(t)}\mathbf{M}_{oc} &= {}^{of(t)}\mathbf{M}_{ci}{}^{ci}\mathbf{M}_{\mathcal{F}_{pt}}{}^{\mathcal{F}_{pt}}\mathbf{M}_{cc}{}^{cc}\mathbf{M}_{oc} & (19) \end{aligned}$$

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

$$ {}^{of(t)}\mathbf{M}_{of(t+\Delta t)} = \Delta\mathbf{M} = \begin{pmatrix} \Delta\mathbf{R} & \Delta\mathbf{T} \\ \mathbf{0}_{1\times3} & 1 \end{pmatrix} \tag{22}$$

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

$$\begin{aligned} {}^{ci}\mathbf{T}_{of(t+\Delta t)} &= {}^{ci}\mathbf{T}_{of(t)} + \Delta\mathbf{T} & (23)\\ {}^{ci}\mathbf{R}_{of(t+\Delta t)} &= \Delta\mathbf{R}{}^{ci}\mathbf{R}_{of(t)} & (24)\end{aligned}$$

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

$$\begin{aligned} {}^{ci}\mathbf{T}_{of(t+\Delta t)} &= {}^{ci}\mathbf{T}_{of(t)} + {}^{ci}\mathbf{R}_{\mathcal{F}_m}\Delta\mathbf{T} & (25)\\ {}^{ci}\mathbf{R}_{of(t+\Delta t)} &= {}^{ci}\mathbf{R}_{\mathcal{F}_m}\Delta\mathbf{R}{}^{\mathcal{F}_m}\mathbf{R}_{of(t)} & (26)\end{aligned}$$

In these equations we have

$$\begin{aligned} \Delta\mathbf{T} &= \mathbf{V}\Delta t \\ \Delta\mathbf{R} &= \cos(\omega\Delta t)\mathbf{I}_{3\times3} + (1-\cos(\omega\Delta t))\mathbf{v}.\mathbf{v}^T + \sin(\omega\Delta t)\tilde{\mathbf{v}}\end{aligned}$$

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:

$$\dot{\mathbf{q}} = J^{-1}_{\mathcal{F}_m}(\mathbf{q}) \begin{pmatrix} \mathbb{I}_{3\times3} & {}^{\mathcal{F}_m}\mathbf{R}_{ec}\widetilde{{}^{e}\mathbf{T}_o}{}^{ec}\mathbf{R}_{\mathcal{F}_m} \\ 0_{3\times3} & \mathbb{I}_{3\times3}\end{pmatrix}.$$

$$\left(-\lambda\left(\mathbf{P}-\mathbf{P_d}(\mathbf{t})\right) + \begin{pmatrix}\mathbf{V}\\ \mathbf{\Omega}\end{pmatrix}_{\mathcal{R}_{\mathcal{F}_m}}\right) \tag{27}$$

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 $\begin{pmatrix}\mathbf{V} & \mathbf{\Omega}\end{pmatrix}^T_{\mathcal{R}_{\mathcal{F}_m}}$

F. Computing articular positions

As already stated, we do not have a direct access to the joint position $\mathbf{q}$ of the manipulator. To compute the joints positions 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:

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

G. 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 [4]. 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:

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

and the resulting control law is simply given by:

$$\begin{pmatrix}\Omega_x \\ \Omega_y\end{pmatrix} = -\lambda_2\begin{pmatrix} \frac{Y}{1+X^2+Y^2} \\ -\frac{X}{1+X^2+Y^2}\end{pmatrix} + \mu I \tag{33}$$

where I is an integral term introduced to attenuate the tracking errors [2].

III. Experimental results

A. Image Processing

The image processing algorithm has 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 4). 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. N points are tracked over the image sequence. Knowing the position of these dots in the image and their 3D position in the object frame we are able to compute the pose ${}^{cc}\mathbf{M}_{oc}$. A number of methods have been proposed to compute pose from points. We have used the method designed by Dementhon [3] completed by Lowe's non-linear method [9]. 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 in approximating perspective projection by scaled orthographic projection, and then in iteratively modifying the scaled orthographic projection to converge to 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.

B. Control experiments

Experiments have been carried out on a 6 dof cartesian robot at Inria Rennes. Control and image processing are performed on a Sun Ultra Sparc 1. Unlike the Victor 6000 manipulator, our 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 actual one.

Figure 4 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 5).

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 $\mathbf{P} - \mathbf{P_d}$.

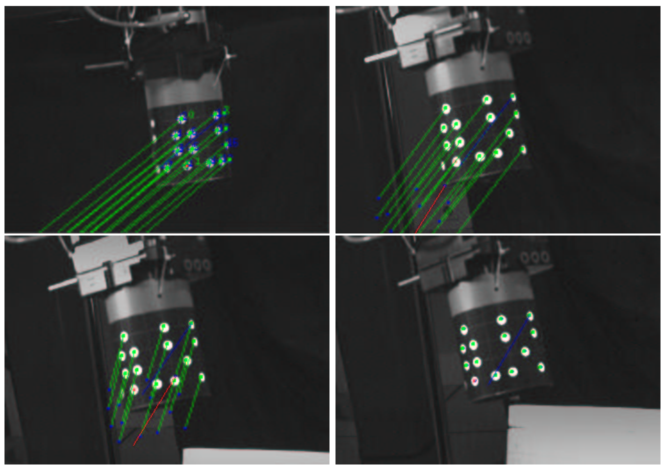
Fig. 4. Target tracking in an image sequence and control of both the manipulator and the pan/tilt camera.

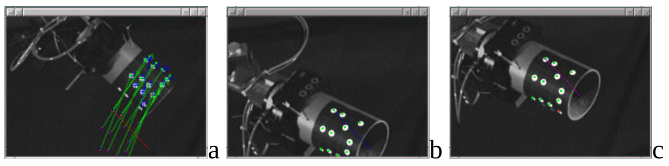

Fig. 5. Effect of the pan/tilt control: the desired position of the object in the image is modified. (a) initial position, (b) desired position with no pan/tilt control, (c) desired position with pan/tilt control.

B.1 Displacement specified as an image to reach

If the desired position of the manipulator is specified as an image and if the camera calibration parameters used for the learning step and servo step are the same, then no error in the positioning process are observed (see Table I). Even with very bad camera calibration parameters, precision remains very good as long as the resolution of the inverse geometrical model remain possible. Errors are then lesser 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	29.33	6.93	33.57
actual	150	47	-100	29.33	6.07	33.17
actual (calibration -20%)	155	52	-103	29.79	3.19	33.05
actual (calibration +20%)	155	45	-100	29.33	6.70	33.57
actual (calibration +40%)	155	45	-99	29.27	6.93	33.51

TABLE I

DISPLACEMENT SPECIFIED IN THE IMAGE (THE DESIRED AND ACTUAL POSITIONS ARE COMPUTED USING THE ROBOT ODOMETRY)

B.2 Position-based control

We now present results dealing with displacement in the various possible frames This example reported in table II features simple displacements ,specified in the object initial frame, in translation and another in rotation as well as a complex displacement of every axes. Plots on Figure 7 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 measurements 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 II).

Figure 6 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.

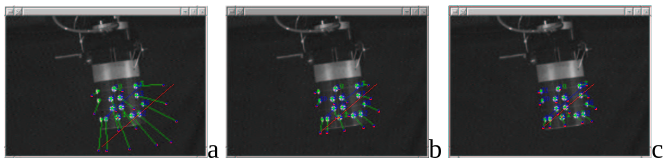

Fig. 6. Effect of the calibration errors on the desired position. The specified displacement is a translation of -500mm along the camera optical axis. (a) initial camera parameters with a noise of -40 % (b) initial camera parameters (c)initial camera parameters with a noise of +40 %

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	-8.81	-0.545	-1.66	-0.12	-0.31	31.64
actual with measured $\mathbf{q}$	-8.71	-0.89	-1.64	-0.07	-0.33	31.776
actual (calibration +20%)	0.94	0.03	1.22	0.23	1.26	30.81
actual (calibration +40%)	3.75	-0.06	1.04	0.17	1.08	30.84
actual calibration -20%	-3.49	-0.42	1.41	0.20	2.00	31.67
desired	300	0	0	0	0	0
actual	291.71	48.78	-1.01	0.00	0.35	1.93
actual with measured $\mathbf{q}$	290.62	49.04	0.49	0.07	0.1	2.04
actual calibration +20%	253.84	41.16	-5.51	0.5	2.07	-0.85
actual calibration +40%	220.39	39.64	-4.58	0.37	1.74	-1.15
actual calibration -20%	371.57	42.91	-6.59	0.52	3.60	0.55
desired	300	50	50	20	20	60
actual	281.07	102.7	56.87	15.42	23.40	64.27
actual with measured $\mathbf{q}$	281.06	102.28	57.15	15.37	23.49	64.15
actual calibration +20%	255.01	97.88	51.27	17.35	27.13	59.20
actual calibration -20%	362.34	96.02	49.87	15.48	28.92	66.87
actual (PT)	283.02	94.75	69.08	15.58	24.43	64.26
actual calibration +20% (PT)	254.6	78.88	41.33	16.97	26.79	59.77
actual calibration -20% (PT)	373.47	97.87	78.94	16.19	29.91	61.47

TABLE II

DISPLACEMENT IN THE INITIAL OBJECT FRAME

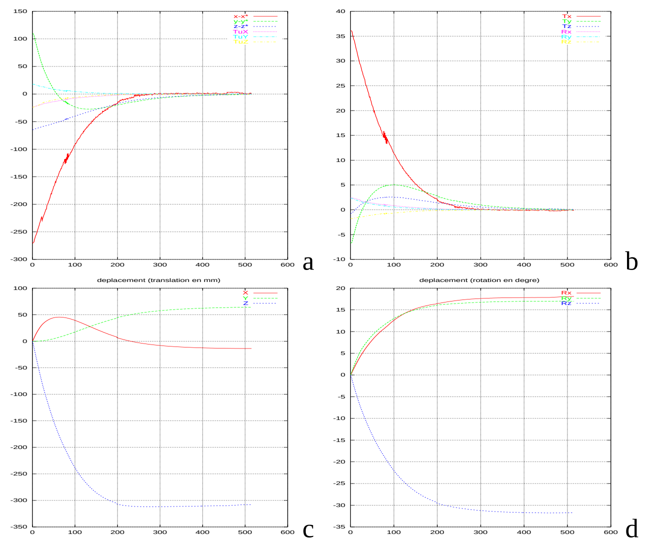

Fig. 7. Results of a displacement specified in the initial camera frame $\Delta\mathbf{T} = (50, 50, -300)$, $\Delta\mathbf{R} = (20, 15, -30)$. All the plots are over time. (a) error $\mathbf{P} - \mathbf{P_d}$, (b) control law (c) displacement in translation (mm), (d) displacement in rotation (deg).

B.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 8 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 of the ${}^{\mathcal{F}_{pt}}\mathbf{M}_{\mathcal{F}_m}$ matrix). The rotation achieved at the end of the rectangle trajectory is correctly handled (no translational motion at this point, see the bottom left corner of Figure 8.a). When a new velocity is given, due to tracking errors, a few iterations are necessary to achieve correctly the new specified velocity (see Figure 8.c for the computed velocity and 8.d for the actual measured velocity).

IV. CONCLUSION

In this paper we proposed a complete framework to control a non-instrumented and roughly calibrated and non-instrumented manipulator using a vision-based approach. To allow the control, we compute on-line the articular position of the manipulator and we achieve the specified displacement using a visual servoing control law. Experiments have been carried out on a 6 dof robot and shows the validity and the efficiency of our approach. Though not reported in this paper, other experiments have carried out on the Victor 6000 arm at ifremer where similar results have been obtained.

Acknowledgment

This work has been supported by Ifremer under contract INRIA/Ifremer1.00.C.001 and INRIA/Ifremer1.00.C.606.

REFERENCES

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

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

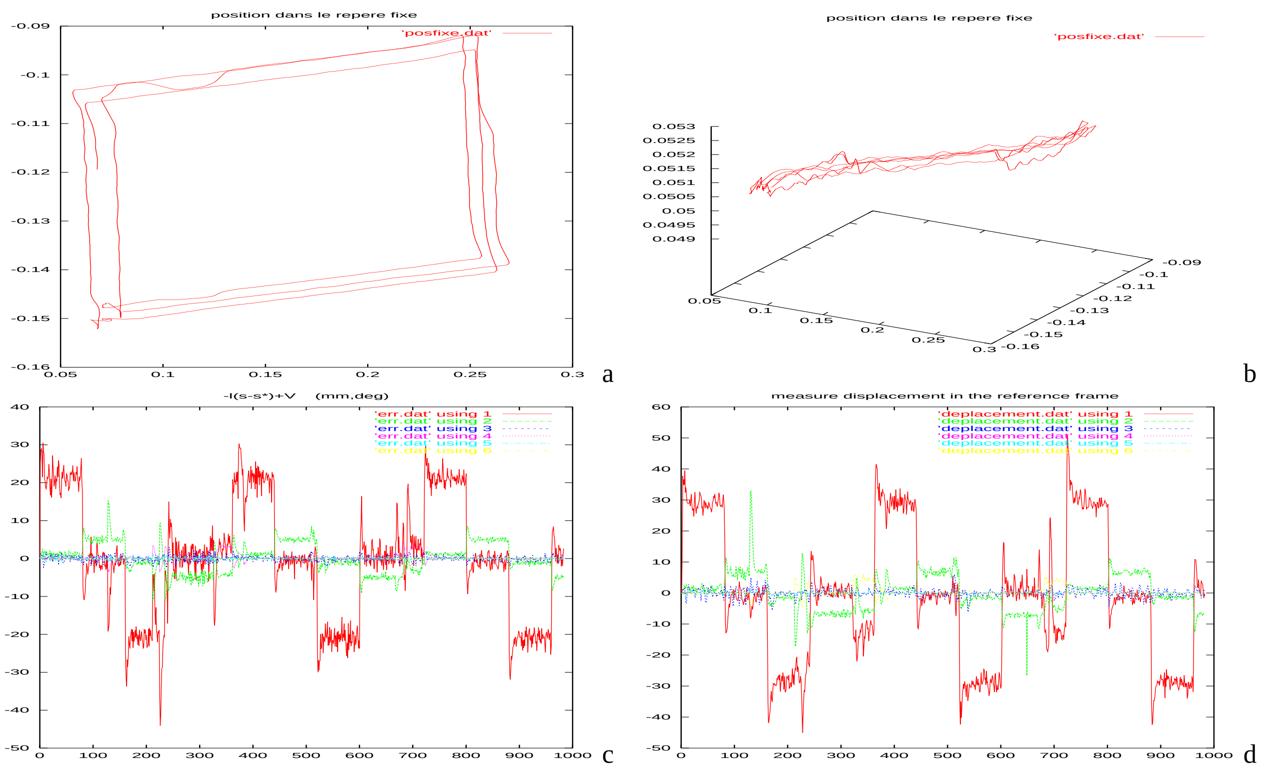

Fig. 8. Velocity control : (a-b) Position of the object in the reference frame ((a) X-Y view (b) X-Y-Z view) (c) Velocity sent to the robot controller (d) Measured velocity

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

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

[5] 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.

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

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

[8] J.-F. Lots, D.M. Lane, and E. Trucco. Application of 2 1/2 d visual servoing to underwater vehicle station-keeping. In *IEEE/OES Oceans 2000 Conference*, Providence, Rhode Island, USA, September 2000.

[9] 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.

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

[11] M. Nokin. Victor 6000, a deep teleoperated system for scientific research. In *MTS/IEEE Oceans'97*, Halifax, 1997.

[12] P. Rives and J.J Borrelly. Visual servoing techniques applied to an underwater vehicle. In *IEEE Int. Conf. on Robotics ans Automation*, Albuquerque, USA, april 1997.

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

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