Robot Control using Monocular Pose Estimation

Philippe Martinet, Nadine Daucher, Jean Gallice, Michel Dhome

Robot Control using Monocular Pose Estimation

P. Martinet, N. Daucher, J. Gallice, M. Dhome

Université Blaise Pascal de Clermont-Ferrand,

Laboratoire des Sciences et Matériaux pour l'Electronique, et d'Automatique.

U.M.R. 6602 du C.N.R.S., F-63177 Aubière Cedex, France.

E-Mail : Philippe.Martinet@lasmea.univ-bpclermont.fr

Abstract

Since the last years, we are interested in visual servoing and we have developed many visual servoing applications using 2D visual features. The aim of this paper is to study how we can introduce 3D visual features in a robot control loop. We consider a camera mounted on the end effector of the manipulator robot to estimate the pose of the target object. The required positioning task is to reach a specific pose between the sensor frame and a target object frame. Knowing the target object model, we can localize the object in the 3D visual sensor frame and obtain the pose between the camera and the target object at each iteration. To insure the visual servoing task, we have developed two approaches. The first approach is based on the task function approach introduced by Samson and Espiau at the end of 80's. We considered a 3D visual sensor which gives the 3D visual pose of an object in the sensor frame. In this case, the sensor signal (dimension 6) is composed of the position and the orientation of the frame object in the sensor frame. The second approach consists to split the trajectory between the current and the desired pose in a set of elementaries screwing tasks. At each iteration, the situation between the sensor and the object is computed, and then the trajectory is updated.

INTRODUCTION

Sanderson and Weiss in [25] introduced an important classification of visual servo structures based on two criterias: space control and presence of joint feedback. So, in this classification we distinguish two main approaches:

- *Position Based Control* : in this case, image features are extracted from the image and a model of the scene and the target is used to determine the pose of the target with respect to the frame attached to the camera.

- *Image Based Control* : in image based control, the pose estimation is omitted, and the control law is directly expressed in the sensor space (image space).

Figure 1 and 2 illustrated the both servoing scheme.

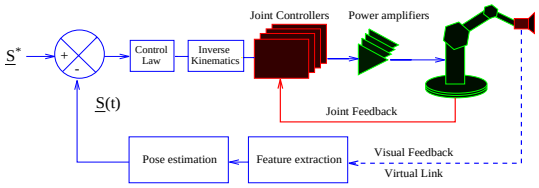

Figure 1: Dynamic position based look and move structure

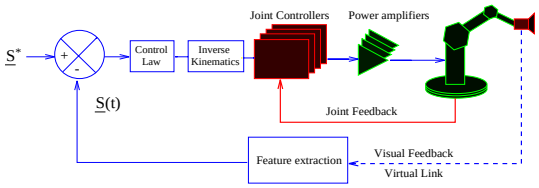

Figure 2: Dynamic imaged based look and move structure

A bibliographic review on Visual Servoing has been done in [13], and particularly developed by P. Corke in [5]. A tutorial on Visual Servo Control has been organised by G. Hager et *al* [12] in 1996. These state of the art in the field of visual servoing show that the Image Based Control has retained to be an alternative way to the Position Based Control approach. Generally, many authors consider that Image Based Control approach is better than the other one, according to the following criteria: camera calibration, hand-eye calibration, robot modelling, scene and target modelling, ... , and also regarding to the processing time to compute the sensor signal.

It is clear that Image Based Control approach do not need a precise calibration and modelling, because of the closed loop defined in the sensor space. Many

works [1], [4], [6], [9], [10], [12], [16], [19], [21], [22], [23] and many others, has been done in the case of the camera sensor and the $2D$ space. We recommand the reader to have a look on the review previously quoted and on [2] for an introduction on active perception.

The notion of *Task Function* introduced by Samson et *al* in [24], can be used to elaborate a control law in the sensor space. According to this concept, Martinet et *al* in [20] introduce the notion of $3D$ logical sensor which deliver a $3D$ sensor signal by monocular vision at video rate. Recent progress in pose estimation, localization and $3D$ modelling [7], [15] shows that it is not an utopic way to introduce $3D$ visual information in a closed loop control. Using this assumption, we can synthesize control laws using this kind of informations like we do directly with the camera sensor. Another way to obtained the estimation of the pose, is to compute a Kalman filter using several visual features like presented in [27]. In this case, the control law is defined to reach a particular pose between sensor and object frames with a PD controller.
In fact, few work [11], [20], [27] has been done using $3D$ sensor signal. However, we can remark that precise calibration and modelling are really usefull only in the case where the task to achieve is expressed in the cartesian space. If we learn the $3D$ reference signal in real conditions, like in Image Based Approach, we obtain the same good results in the logical sensor space.

The aim of this paper is to study how we can introduce 3D visual features in a robot control loop. In the first section of the paper, we discuss how to extract 3D visual features with a logical sensor. Knowing the object model, the logical sensor uses the De Menthon Algorithm [7] to localize the object in the 3D visual sensor frame. At each iteration we obtain the pose between the camera and the target object. In the second section, and the third section we present the two approaches developed in our laboratory. In the last section, we show the results obtained by simulation and experimentation with our robotic platform. We use a specific object composed of four illuminated points in real experimentation. Finally we conclude and present some perspectives for future development in visual servoing.

POSE ESTIMATION BY MONOCULAR VISION

During the last ten years, many methods to locate 3D objects from one monocular image was proposed in the litterature. All of these approaches need some prerequisites, as the knowledge of the geometrical model of the observed object and matches between 2D image primitives and 3D model elements, to compensate the loss of data due to the projection of the 3D world on the image plane. Using these informations, the methods look for the spatial attitudes of the 3D model such that the matched model elements are projeted on the corresponding image primitives.

All of these approaches can be classified in function of the manipulated image primitives (*points, straight lines, elliptical contours, limbs of curved surfaces ...*) and the assumption about the projection of the real world on the image plane (*orthographic, weak perspective, full perspective projection*). Some methods give closed form solutions of the addressed inverse perspective problem, the others uses iterative processes to reach the solution.

The reader will find in the following references an overview of the different kinds of approach [3], [7], [8], [14], [15], [17], [18], [26].

If up to a recent time, the computing time of such algorithms was prohibity to implement them during a real time closed loop control : it is no more the case. For example, the approach proposed by D. Dementhon [7] allows to compute the spatial attitude of a 3D object in a fraction of the image acquisition cycle.

FIRST APPROACH

Introduction

In the last ten years, many people have become interested in how to introduce external sensor signals directly into the control loop. Near the end of the 1980's, C. Samson and B. Espiau proposed a new approach, called "Sensor Based Control" [9], [24]. In this approach, any robot task can be characterized as a regulation to zero of a task function, where it defines the interaction between the sensors and the robot environment. It is expressed with the following relation:

$$\underline{e}(\underline{r},t) = [\underline{s}(\underline{r},t) - \underline{s}^{\star}] \tag{1}$$

where $\underline{s}^{\star}$ is considered as a reference target sensor feature to be reached in the sensor frame, and $\underline{s}(\underline{r},t)$ is the value of sensor information currently observed by the sensors.
Figure 3 shows the $3D$ sensor based servoing scheme.

The interaction between sensor and scene is given in matrix form by:

$$\frac{\partial \underline{s}}{\partial \underline{r}} = L^{t}_{\underline{s}} \tag{2}$$

This is called the interaction matrix which links the interaction between the robot and its environment. We

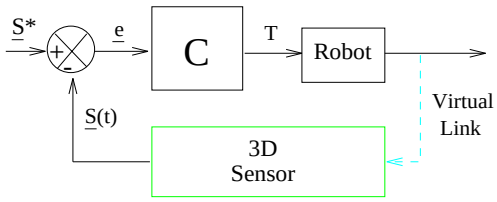

Figure 3: 3D servoing scheme

have the following relation:

$$\dot{\underline{s}} = L_{\underline{s}}^{t} T \tag{3}$$

where $\dot{\underline{s}}$ is the time variation of $\underline{s}$ and $T = (\underline{V}, \underline{\Omega})^t$ is the velocity screw of the camera which englobes three translations and three rotations.

Considering an exponential convergence (for a task positioning) of all components of the task function (i.e. $\dot{e} = -\lambda e$, where λ is a positive scalar constant), we obtain the expression of the control law:

$$C = \lambda L_{\underline{s}}^{t+} \tag{4}$$

where $L_{\underline{s}}^{t+}$ represents the pseudo-inverse matrix of $L_{\underline{s}}^{t}$.

The stability depends of the positivity of the product term $L_{\underline{s}}^{t}.C$. If $L_{\underline{s}}^{t^{-1}}$ exists in all robot configurations, this control law realize a perfect decoupling servoing scheme, and it is equivalent to consider the system as a juxtaposition of a set of closed integrator. Figure 4 represents the servoing scheme for each component S_i of the sensor signal $\underline{S}$.

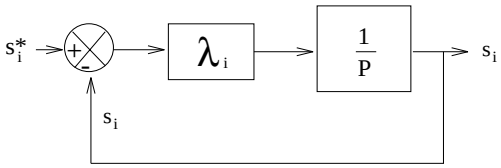

Figure 4: Servoing scheme in ideal case

We choose the positif gain λ to insure the stability. The sensor signal $\underline{S}$ decreases as a first order system with a constant of time $\tau = \lambda^{-1}$. In this case, the first order steady state error is null ($\underline{e}_{\infty} = 0$).

Principle

We consider a scene with a 3D object and a wrist 3D sensory apparatus mounted on the end effector of the robot. We define three homogeneous matrices transformation as follows:

- $\underline{M}_o$ is the homogeneous matrix transformation between an absolute frame attached to the scene, and the object frame $\mathcal{R}_o$
- $\underline{M}_{c_t}$ is the homogeneous matrix transformation between an absolute frame attached to the scene, and the sensor frame computed at each iteration $\mathcal{R}_{c_t}$
- $\underline{M}_c^*$ is the homogeneous matrix transformation between an absolute frame attached to the scene, and the sensor frame desired at the equilibrium $\mathcal{R}_c^*$.

Figure 5 represents the scene with the different frames.

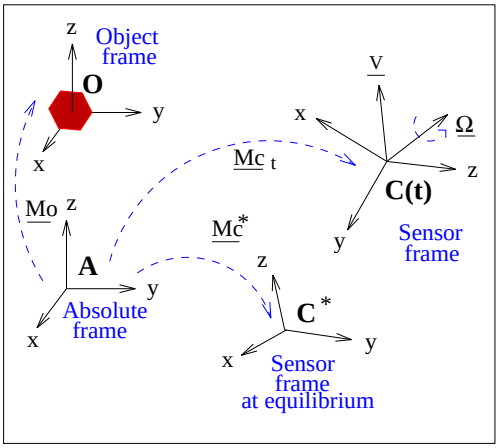

Figure 5: Different frames used in modelling

The sensor signal $\underline{S}$ is a vector of dimension 6 and is composed of:

- $\underline{X} = [\underline{AC}]_c$ the position of the absolute frame expressed in the sensor frame
- $\underline{\theta}$ the orientation of the sensor frame relative to the absolute frame attached to the scene.

We have: $\underline{S} = (\underline{X}, \underline{\theta})^t$.

In the absolute frame, we have the relation:

$$[\underline{AC}]_a = R_c [\underline{AC}]_c \tag{5}$$

where R_c is the rotation matrix extracted from $\underline{M}_{c_t}$.

We use the exponential representation for the expression of the rotation. So, we write:

$$R_c = exp(AS(\underline{\theta})) \tag{6}$$

where $AS(\underline{\theta})$ represents the antisymetric matrix associated with the vector $\underline{\theta}$.

In our case, we have $[\underline{\theta}]_a = R_c [\underline{\theta}]_c = [\underline{\theta}]_c$ because the product $AS(\underline{\theta}) * \underline{\theta} = 0$ (cross product of two identical vectors).

Derivating relation 5, we obtain:

$$\frac{d[\underline{AC}]_a}{dt} = \frac{dR_c}{dt} * [\underline{AC}]_c + R_c * \frac{d[\underline{AC}]_c}{dt} \tag{7}$$

In this equation, the first term represents the translation velocity $\underline{V}$ of the robot expressed in the absolute frame. Using the definition of R_c, we have: $\frac{dR_c}{dt} = AS(\underline{\Omega}) * R_c$. In this expression, $\underline{\Omega}$ represents the rotation velocity vector. Using the transpose matrix of R_c, we can write the following relation:

$$R_c{}^t[\underline{V}]_a = R_c{}^t * AS(\underline{\Omega}) * R_c * [\underline{AC}]_c + \frac{d[\underline{AC}]_c}{dt} \quad (8)$$

where:

- the first term $R_c{}^t[\underline{V}]_a$ represents the translation velocity $\underline{V}$ of the sensor in the sensor frame
- the expression $R_c{}^t * AS(\underline{\Omega}) * R_c$ can be rewrite as $-AS(\underline{\Omega})$.

So, we have:

$$\frac{d\underline{X}}{dt} = \underline{V} + AS(\underline{\Omega}) * \underline{X} \quad (9)$$

with $\underline{X} = [\underline{AC}]_c$.

Using the relation $AS(\underline{\Omega}) * \underline{X} = -AS(\underline{X}) * \underline{\Omega}$, we can rewrite 9 as:

$$\frac{d\underline{X}}{dt} = [I_3, -AS(\underline{X})] * T \quad (10)$$

where $T = (\underline{V}, \underline{\Omega})^t$ is the kinematic screw expressed in the sensor frame. We have also the following relation:

$$\frac{d\underline{\theta}}{dt} = \underline{\Omega} \quad (11)$$

Derivating $\underline{S} = (\underline{X}, \underline{\theta})^t$, from relation 3, 10 and 11, we obtain:

$$\dot{\underline{S}} = L_{\underline{s}}^t * T = [[I_3, -AS(\underline{X})]; [O_3, I_3]] * T \quad (12)$$

We can give the expression of the inverse of $L_{\underline{s}}^t$:

$$L_{\underline{s}}^{t^{-1}} = [[I_3, AS(\underline{X})]; [O_3, I_3]] \quad (13)$$

The control matrix is given by $C = \lambda L_{\underline{s}}^{t^{-1}}$, and the system is equivalent to a set of decoupled integrators. In real case, if we consider an estimation of the sensor signal $\underline{S} = \underline{S}_{est}$, the control matrix becomes:

$$C = \lambda * [[I_3, AS(\underline{X}_{est})]; [O_3, I_3]] \quad (14)$$

and the product $L_{\underline{s}}^t * C$ is not equal to λ but to $\lambda * L_{\underline{s}}^t * L_{\underline{s}_{est}}^{t^{-1}}$. In fact, we have:

$$L_{\underline{s}}^t * C = \lambda * [[I_3, AS(\underline{X}_{est} - \underline{X})]; [O_3, I_3]] \quad (15)$$

The orientation is decoupled and we obtain an exponential decay of the rotation angles. It is not the same case for the coordinate of the end effector.

Convergence and stability of the control law

To insure the stability of the system, the product $L_{\underline{s}}^t * C$ must be positive. In other terms, for all vectors $\underline{Z} = (\underline{z}, \underline{a})^t$ of the state space (dimension 6) , we have to verify the condition:

$$\underline{Z}^t * L_{\underline{s}}^t * C * \underline{Z} > 0 \quad (16)$$

and then that the scalar product $< \underline{Z}, L_{\underline{s}}^t * C * \underline{Z} >$ is positive.

We can rewrite the scalar product $< \underline{Z}, L_{\underline{s}}^t * C * \underline{Z} >$ as:

$$< \underline{Z}, \underline{Z} > + < \underline{z}, AS(\underline{X}_{est} - \underline{X}) * \underline{z} > \quad (17)$$

In this new expression, the second term is null (triple product with two identical vectors) and the first term represents the norm of the vector $\underline{Z}$. Then, for all $\underline{Z}$ different to zero, we have $< \underline{Z}, L_{\underline{s}}^t * C * \underline{Z} > \; > 0$ and the matrix $L_{\underline{s}}^t * C$ is always positive which insure the stability of the system for all configurations of the robot.

The steady state error is $\underline{e} = \underline{S}^* - \underline{S_{est}}$ becomes null at the end of the servoing and insures a perfect positioning task. The reference $\underline{S}^*$ must be learned by training, and then it compensate the error of modelling.

Modelling the closed loop control

To model the closed loop control, we have to compute the evolution of the sensor signal $\underline{S}$ at each iteration. For that, we have to solve the following relation:

$$\dot{\underline{S}} = L_{\underline{s}}^t * T \quad (18)$$

for a given kinematic screw T. Considering that $\underline{S} = (\underline{X}, \underline{\theta})^t$, from relation 12 we deduce:

$$\begin{aligned} \dot{\underline{\theta}} &= \underline{\Omega} \\ \dot{\underline{X}} &= \underline{V} + AS(-\underline{X}) * \underline{\Omega} = \underline{V} + AS(\underline{\Omega}) * \underline{X} \end{aligned} \quad (19)$$

Then, we can solve the system of equation like:

$$\begin{aligned} \underline{\theta}(t) &= \underline{\theta}(t_0) + \int \underline{\Omega}(t')dt' \\ \underline{X}(t) &= exp(\int AS(\underline{\Omega}(t'))dt') * \underline{X}(t_0) + \\ & exp(\int AS(\underline{\Omega}(t'))dt') * \int exp(-\int AS(\underline{\Omega}(t''))dt'')\underline{V}(t')dt' \end{aligned} \quad (20)$$

If we consider a kinematic screw constant between time t_0 and t, we obtain the following relation:

$$\begin{aligned} \underline{\theta}(t) &= \underline{\theta}(t_0) + \underline{\Omega} * (t - t_0) \\ \underline{X}(t) &= exp(AS(\underline{\Omega}) * (t - t_0)) * \underline{X}(t_0) + \\ & \int_0^{t-t_0} exp(AS(\underline{\Omega}) * t')dt' * \underline{V} \end{aligned} \quad (21)$$

Using the Rodrigues formula, we have:

$$exp(AS(\underline{\Omega}) * (t - t_0)) = cos(\Omega(t - t_0)) * I_3 + sin(\Omega(t - t_0)) * AS(\underline{v}) + (1 - cos(\Omega(t - t_0))) * \underline{v} * \underline{v}^t \quad (22)$$

In the following expression, $AS(\underline{\Omega}) = \Omega * AS(\underline{v})$, Ω represents the norm of the vector $\underline{\Omega}$ and ($\underline{v}$ the unitary vector in the direction of $\underline{\Omega}$).

So, we can rewrite expression 21 like:

$$\underline{X}(t) = [cos(\Omega(t - t_0)) * I_3 + sin(\Omega(t - t_0)) * AS(\underline{v}) + (1 - cos(\Omega(t - t_0))) * \underline{v} * \underline{v}^t] * \underline{X}(t_0) + [sin(\Omega(t - t_0))/\Omega * I_3 + (1 - cos(\Omega(t - t_0))/\Omega) * AS(\underline{v}) + (t - t_0 - sin(\Omega(t - t_0))/\Omega) * \underline{v} * \underline{v}^t] * \underline{V} \quad (23)$$

SECOND APPROACH

Principle

The second approach is based on following assumption: the whole system including image acquisition and localization algorithm is taken as a black box delivering at video rate the pose parameters of an observed object.

Moreover, the proof of the convergence presented in the next paragraph is based on the hypothesis that the robot motion is perfect. The computation at each image shot of the target object pose permits to handle the behavior of the robot system.

The pose parameters can be indifferently seen as the object pose in the camera reference system or as the pose camera relative to the object. These informations can be expressed as rigid transformation matrices of the form:

$$M = \begin{bmatrix} R_{3X3} & T_{3X1} \\ 0 & 1 \end{bmatrix}$$

where R is a rotation matrix and T a translation vector.

If, (see figure 6) M_t denotes the object localization matrix in the reference system R_{ct} of the camera at time t, and M_* the object desired pose matrix in R_{c*}, the C_t command to be applied to the robot must transform R_{ct} into R_{c*}.

In this way the coordinates $[X]_m$ of a point X of the object defined in the R_m reference system are moved to $[X]_{ct}$ in the reference system R_{ct} following the formula:

$$[X]_{ct} = M_t[X]_m$$

In the same way,

$$[X]_{c*} = M_*[X]_m$$

where $[X]_{c*}$ are the coordinates of X in R_{c*} .

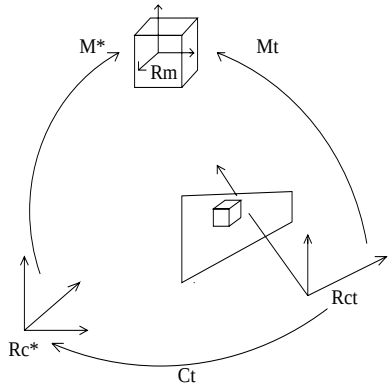

Figure 6: Control law computation

We deduce

$$\begin{cases} [X]_m = M_t^{-1}[X]_{ct} \\ [X]_m = M_*^{-1}[X]_{c*} \end{cases}$$

Thus

$$[X]_{c*} = [M_t M_*^{-1}]^{-1}[X]_{ct}$$

This relation expresses the fact that the matrix $[M_t M_*^{-1}]^{-1}$ is the pose matrix for reference system R_{ct} relative to R_{c*}.

Obviously the command transforming R_{ct} in R_{c*} is:

$$\boxed{C_t = M_t M_*^{-1}}$$

The command is given by a rigid transformation matrix and is numerically performed as a (4×4) matrix product, the inversion of M_* being computed once off line before running the feedback control process.

The task funtion can be expressed relative to the command law as:

$$[e]_t = M_* - M_t = [\mathbf{I} - C_t]M_*$$

where

$$\mathbf{I} = \begin{bmatrix} (I)_{3X3} & (0)_{3X1} \\ 0 & 1 \end{bmatrix}$$

On the other side, the pose to obtain expressed versus the current pose is:

$$M_* = C_t^{-1} M_t$$

From the matrix C_t, we can extract the parameter vector $\underline{\Delta X}$ of the rigid transformation between R_{c*} and R_{ct}. The inverse geometric model of the robot transforms this vector in an vector $\underline{\Delta q}$ of the corresponding joint variations. Last, a normalization relative to the sampling period Δt leads to the speed vector $\underline{\dot{q}}$ to be applied to the motors controling the joints.

If the observed object is not moving, the minimization of the task function will be completed when the vector $\underline{\dot{q}}$ will be zeroed (see figure 6), which corresponds to a command C_t:

$$C_t = \begin{bmatrix} (I)_{3X3} & (0)_{3X1} \\ 0 & 1 \end{bmatrix}$$

At this time $M_t = M_*$ and the regulation of the task function is done, the robot being located in the desired pose.

Theoretically, the form of the command law show that the task can be performed in one iteration, of course numerous factors prevent from sending the whole command directly to actuators of the joints; the most important being the robot dynamic.

The first factor naturally leads to consider only a fraction $\lambda < 1$ of the command:

$$\boxed{C_{t,\lambda} = (M_t.M_*^{-1})^{\lambda}}$$

The computation of the fractional power of a rigid transformation matrix λ^{th} is simple using a screwing representation (see figure 7 for more details).

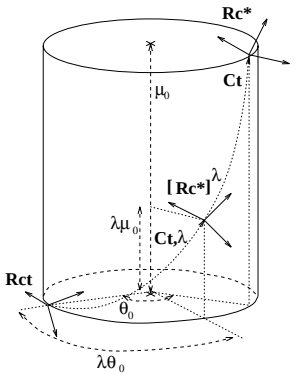

Figure 7: Screwing representation of a rigid transformation

At each iteration (image shot), we compute the current pose M_t from images primitives and the command $C_{t,\lambda}$ to send to perform the task. At each iteration, the M_t computation will also permits to take into account the two kinds of following errors:

- errors induced by visual primitives extraction;
- modelization errors in the robot and the acquisition system.

The choice of λ is important and its value depends of the state of completion of the task. We will give in the following paragraph, a necessary condition for task convergence.

Convergence and exponential decrease of the task function

Convergence

Let us suppose that at time t in the servo control process the object is located at M_t relative to the robot camera. The task function to regulate is:

$$[e]_t = M_* - M_t$$

The command $C_{t,\lambda}$ will now bring the robot in a new pose given by:

$$M_{t+1} = C_{t,\lambda}^{-1}.M_t$$

Thus:

$$M_{t+1} = (M_* M_t^{-1})^{\lambda} M_t$$

To prove the theoretical convergence of the task, it suffices to show that M_t tends to M_* as t tends to infinity.

It is easy to prove by recursion that:

$$M_t = (M_* M_0^{-1})^{1-(1-\lambda)^t} M_0 \qquad (24)$$

where M_0 is the initial pose of the object.

Thus, M_{t+1} tends to M_* if and only if:

$$\lim_{t\to\infty} 1 - (1-\lambda)^{t+1} = 1$$

This condition is satisfied if and only if:

$$\boxed{\lambda \in]0, 2[}$$

We find again that if $\lambda = 1$, (which is only to apply the whole command at the first step) the task function is immediately satisfied. Extremal values $\lambda = 0$ and $\lambda = 2$ respectively correspond to a static behavior of the robot ($M_t = M_0 \forall t$) and at a oscillating one ($M_t = M_0$ if t is even, $M_t = (M_* M_0^{-1})^2 M_0$ if t is odd).

Exponential decrease

We have shown that our command law insures the convergence of the task function. To be sure of good performance, it is quite necessary to know what is the rate of this convergence. We show here that the error is exponentially decreasing with time.

Let us remember that the task function is given at time t by:

$$[e]_t = M_* - M_t$$

If we replace M_t by its computed value (24), we obtain:

$$[e]_t = M_* - (M_* M_0^{-1})^{1-(1-\lambda)^t} M_0$$

Thus:

$$[e]_t = [I - (M_0 M_*^{-1})^{-1+(1-\lambda)^t} M_0 M_*^{-1}] M_*$$

And we have the following relation:

$$[e]_t = [I - (M_0 M_*^{-1})^{(1-\lambda)^t}]M_*$$

Time evolution of $[e]_t$ depends of $(M_0 M_*^{-1})^{(1-\lambda)^t}$. This expression is nothing but a fraction of the screwing representation $(M_0 M_*^{-1})$. If μ_0 and θ_0 respectively characterize the translation norm and the rotation angle of the screwing representation $(M_0 M_*^{-1})$, the desired fraction will be represented by a rotation angle θ_t, and a translation vector $\mathcal{T}_t$ whose components will be determined by the following relations:

$$\theta_t = (1-\lambda)^t \theta_0$$

and

$$\mathcal{T}_t = \begin{bmatrix} x_c(1-\cos\theta_t) + y_c \sin\theta_t \\ -x_c \sin\theta_t + y_c(1-\cos\theta_t) \\ (1-\lambda)^t \mu_0 \end{bmatrix}$$

where the coordinates x_c, y_c characterize a point of the screwing axis in a particular coordinates system.
The origin of this system corresponds to the origin of the reference system R_{ct} (see figure 7). The z-axis is colinear to the screwing axis, the x-axis is colinear to the projection of the translation in a plane orthogonal to the screwing axis and the y-axis is obtained by a cross product.
The screwing angle is now obviously exponentially decreasing if $\lambda \in]0,1[$. For the translation, we will show that its norm is non increasing along time and majored by a decreasing exponential.

Using elementary trigonometry, we can write:

$$\mathcal{T}_t = \begin{bmatrix} 2\sin\frac{\theta_t}{2}(x_c \sin\frac{\theta_t}{2} + y_c \cos\frac{\theta_t}{2}) \\ -2\sin\frac{\theta_t}{2}(x_c \cos\frac{\theta_t}{2} - y_c \sin\frac{\theta_t}{2}) \\ (1-\lambda)^t \mu_0 \end{bmatrix}$$

The square of the norm of vector $\mathcal{T}_t$ can be written:

$$||\mathcal{T}_t||^2 = 4\sin^2\frac{\theta_t}{2}(x_c^2 + y_c^2) + (1-\lambda)^{2t}\mu_0^2$$

The two conditions:

$$\begin{cases} \lambda \in]0,1[\\ \theta \in [0,\pi] \end{cases}$$

insure that the norm decrease to 0. Moreover as $\forall x |\sin x| < |x|$, we get:

$$||\mathcal{T}_t||^2 \leq \theta_t^2(x_c^2 + y_c^2) + (1-\lambda)^{2t}\mu_0^2$$

Thus:

$$||\mathcal{T}_t||^2 \leq (1-\lambda)^{2t}\theta_0^2(x_c^2 + y_c^2) + (1-\lambda)^{2t}\mu_0^2$$

$$||\mathcal{T}_t||^2 \leq (1-\lambda)^{2t}[\theta_0^2(x_c^2 + y_c^2) + \mu_0^2]$$

that prove the property previously stated.

It is interesting to notice that the servoing behavior is quite independant of the starting position (the object must be visible). The above computations prove the convergence and decrease of the task function.

RESULTS

Simulation results: First approach

To validate the first approach, we have developed a simulator in Matlab using the main relations previously developed. We represent the scene with two frames. The object frame is centered in the space, and materialized by a pyramid.

Figure 8 shows the $3D$ trajectory of the sensor frame during servoing. The initial $\underline{S_i}$ and the final $\underline{S_f} = \underline{S^*}$ value of the sensor signal are:

$$\begin{aligned} \underline{S_i} &= (-1.691, -1.978, -0.691, 1.047, 0, -1.047)^T \\ \underline{S_f} &= (-0.150, -2.589, -0.076, -1.034, 1.561, -0.007)^T \end{aligned} \quad (25)$$

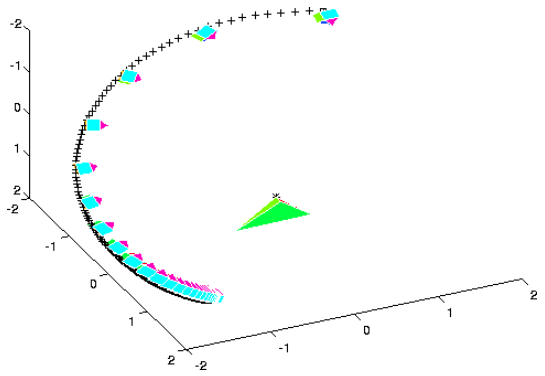

Figure 8: 3D trajectory of the sensor

Figure 9 and 10 show the evolution of sensor signal.

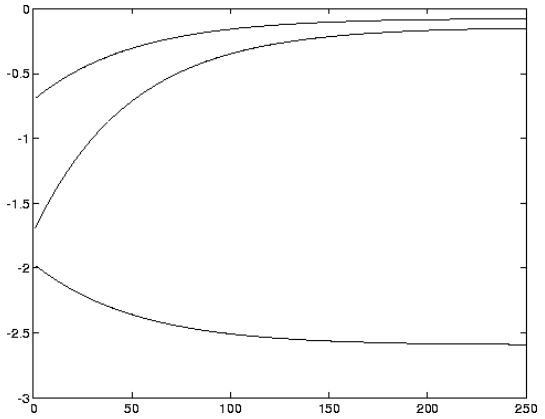

Figure 9: Position of the sensor

If we introduce a noise of 30% (proportional to the magnitude of the sensor signal) directly on the sensor signal, we obtain the behavior represented on the figure 11. We can observe a little disturbance during servoing, but the convergence of the control is insured.

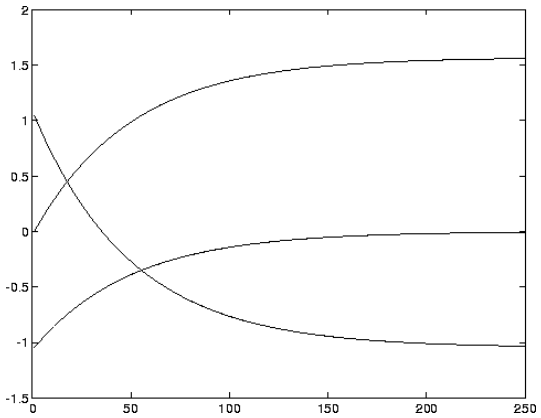

Figure 10: Orientation of the sensor

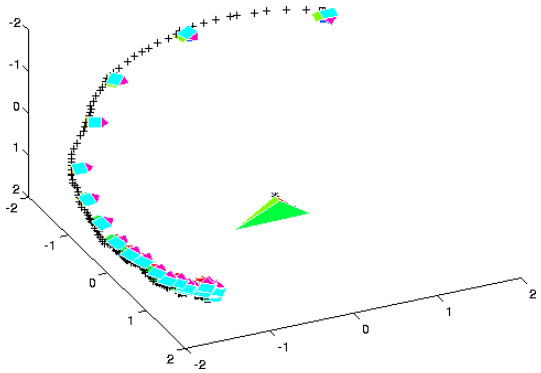

Figure 11: 3D trajectory of the sensor with noise

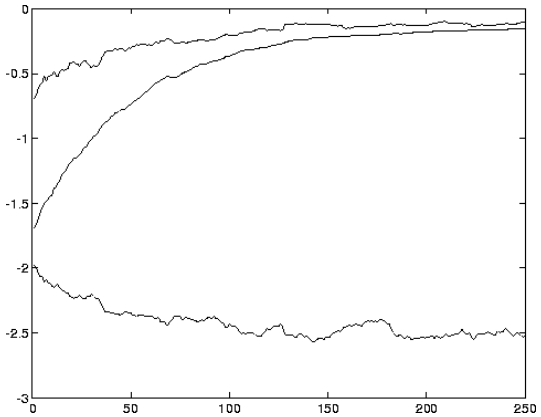

Figure 12: Position of the sensor with noise

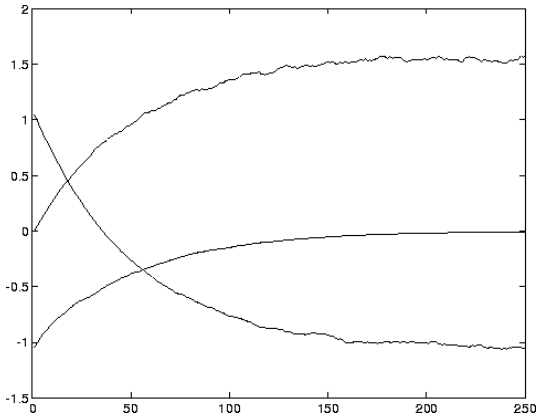

Figure 13: Orientation of the sensor with noise

Figure 12 and 13 show the evolution of sensor signal in presence of noise. As we can see, the servoing is always insured despite of the presence of noise

Experimental results: both approaches

We have tested the two control laws by tracking a regular tetrahedron object materialized by four leds, with a camera mounted on the end effector of a cartesian robot.

The low level image processing consists of a simple barycenter computation for each of the leds. From these four detected points, the Dementhon's algorithm [7] is used to locate the modelled object and thus the control laws can be computed. The sampling period is given by the image acquisition time, i.e. 40 ms.

First approach

We define three frames as follows: $\mathcal{R}_S$ is an absolute frame attached to the scene, $\mathcal{R}_C$ is a frame attached to the 3D sensor, and $\mathcal{R}_O$ is the object frame.

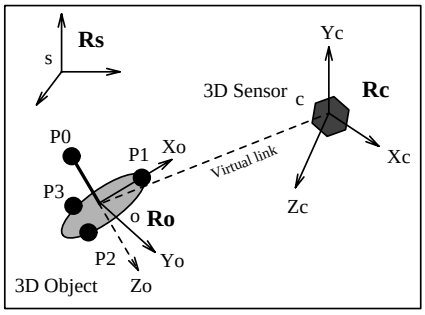

Figure 14: Object and sensor in the scene.

Figure 14 represents the scene with the different frames used in this experimentation.

To test our approach, we implemented a $3D$ logical

sensor on the "Windis" parallel vision system. This architecture, developed in collaboration with INRIA [1] in Sophia Antipolis (France), implements the concept of active windows. Figure 15 shows this architecture, where several active windows (of varying size and position) are extracted from the image at video rate. Low level processing ($3*3$ or $5*5$ convolution) is then executed, and results are sent to the DSP modules. The DSP modules provide a geometric description of the required primitive in each window. The window manager controls all the architecture and computes the "$3D$ logical sensor" features resulting from the geometric description. We use a six degrees of freedom cartesian robot, where the camera is embedded on the end effector. The control law was evaluated and implemented on our experimental site (see Figure 16). All software was written in C language and we use the VxWorks real time system environment.

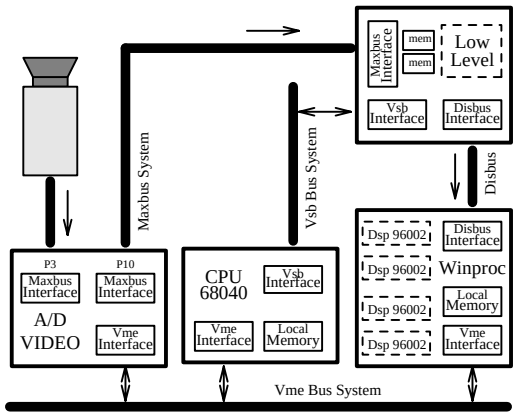

Figure 15: Overview of Windis architecture

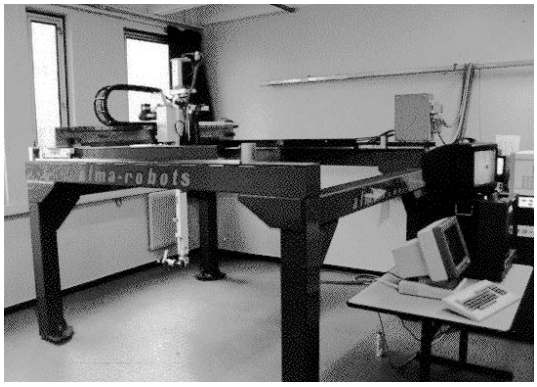

Figure 16: Overview of the robotic platform

We have tested the first approach with various initial and final situations of the visual sensor frame relating to the $3D$ object frame. The initial position of the sensor is always $\underline{X} = (-253, -101, 1500)^t$ (mm) in front of the object and the orientation $\underline{\theta} = (0, 0, 0)^t$ (deg). For the final pose $\underline{S}^* = \underline{S}_f = (\underline{X}, \underline{\theta})^t$ we choose a specific position $\underline{X} = (0, 0, 750)^t$ (mm) and different orientation $\underline{\theta} = (\theta_X, 0, 0)^t$ depending on the θ_X angle $(10, -10, 15, -15, -20)$ (deg). In each case, we estimate the interaction matrix at the equilibrium situation ($L^t_{\underline{s}} = L^t_{\underline{s}=\underline{s}^*}$). We obtained an exponential decay of the error signal and a convergence at around 200 iterations during all tests. We present the results for $\theta_X = 15$ deg.
Figure 17 shows the $3D$ points trajectories during the visual servoing (which traduces the object trajectory in the camera frame).

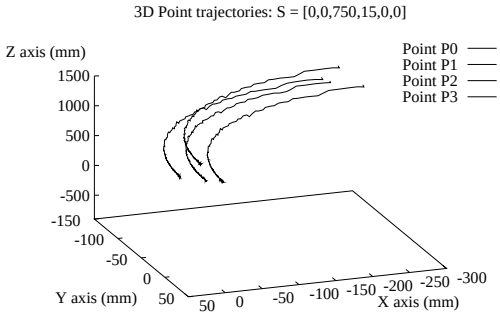

Figure 17: Object trajectory

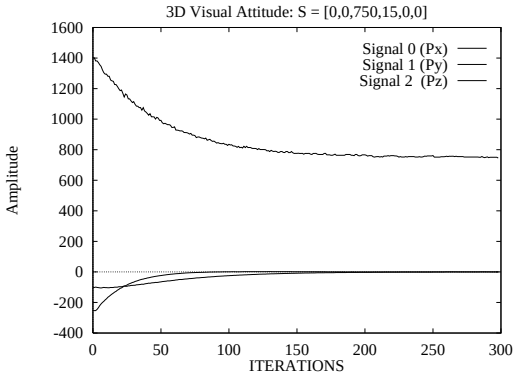

Figure 18: Sensor signal: position

In Figures 18 and 19, the position and orientation of the object are presented. The behavior is like a first order system. We noticed a little disturbance on the curves, due to the $3D$ logical sensor sensibility (less than 1 degree per second on the rotation velocities). This disturbance is due to the difficulty to extract the Z position and the orientation $\underline{\theta}$ of the object in the sensor frame in a dynamic sequence. To improve the results of the $3D$ logical sensor we think that we have

[1] Institut National de Recherche en Informatique et Automatique

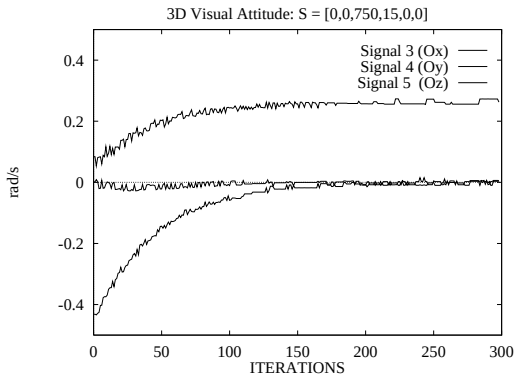

Figure 19: Sensor signal: orientation

to use a filter on the pose parameters like Wilson propose in [27].

Second approach

In the second approach, we have chosen to control the robot such as the desired pose of the tetrahedron (M_*) gives a symmetric centered image.

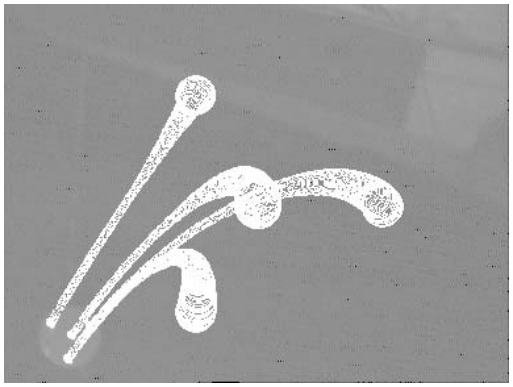

Figure 20: The four leds evolution

The first scheme (see figure 20) shows the evolution in the image of the four barycentres relative to a 3D camera displacement around 1.2m.

Figure 22 and figure 23 show the robot configurations at the beginning and at the end of the process. Figure 24 and figure 25 characterize the evolution of the command applied to the camera and figure 21 shows the displacement of the optical center in the space during servoing. We can note the exponential decrease of these different values and a convergence before 200 iterations.

The last curves (see figure 26 and figure 27) are relative to the speed values applied to the robot articulations. They are to be compared to the extremal possible values that are 1 m/s for the translations and 114 deg/s for the rotations and depend of the λ value ($\lambda = 0.5$ in this experiment).

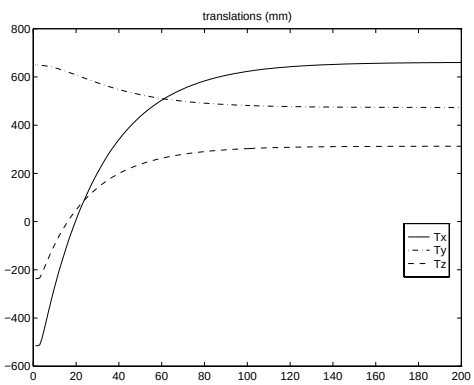

Figure 21: Camera optical center positions evolution

Figure 22: Initial robot arm configuration

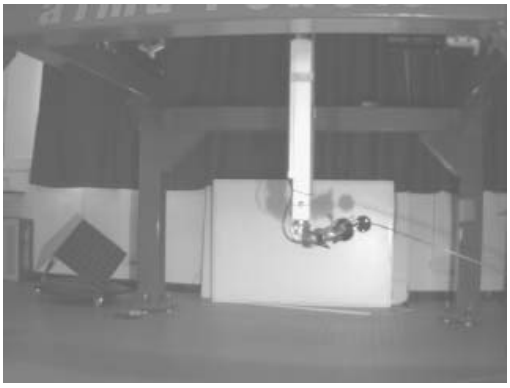

Figure 23: Final robot arm configuration

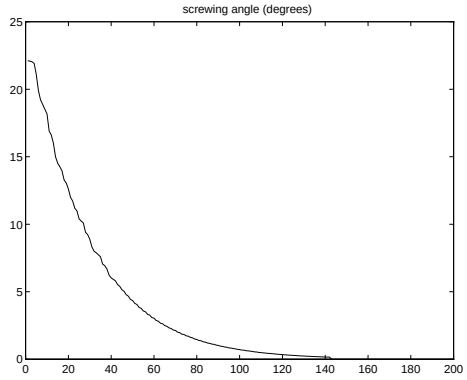

Figure 24: Screwing angle evolution

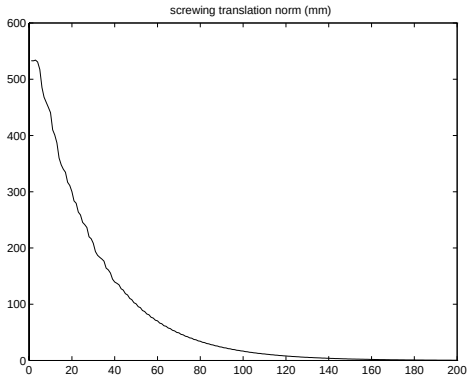

Figure 25: Screwing translation norm evolution

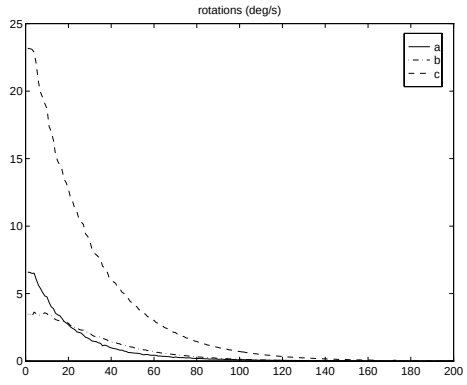

Figure 26: Rotation parameters evolution

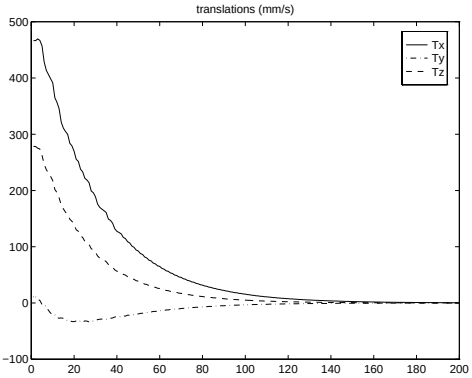

Figure 27: Translation parameters evolution

CONCLUSION

Many people are interested in visual servoing. Until now, only $2D$ visual features has been considered. In this paper, we show that 3D visual sensor elaborating 3D features at video rate can be used. We have developed theorically two control laws and validated their effectiveness in real conditions.

Results seem to be satifactory in regard with the primitive extraction (image led barycenter computation) and pose estimation (De Menthon algorithm) algorithms.

Actually, we have not realize a complete comparison between these both approaches, and we present no results in this way.

The particularity of this kind of method appears in the simplicity of the formalism; the control laws depend only on the desired and current situations of the observed object. Then, from one application to another, only the pose algorithm has to be modified.

For the future, we want to compare the two $3D$ approaches developed in this paper.

REFERENCES

1. Allen P.K., B. Yoshimi, A. Timcenko, "Real time visual servoing", Proc. of the Int. IEEE Conference on Robotics and Automation, 1991.
2. Aloimonos Y., "Active Perception", Lawrence Erlbaum Associates Publichers, 1993.
3. Barnard S.T., "Choosing a Basis for Perceptual Space", Computer Vision Graphics and Image Processing, vol. 29, n.1, pp 87-99, 1985.
4. Chaumette F., "La relation vision-commande: thérorie et applications à des tâches robotiques", Phd Thesis, IRISA, 1990.
5. Corke P., "Visual control of robot manipulators - A review", in "Visual Servoing", Hashimoto K., World Scientific, pp. 1-31, 1993.
6. Debain C., "Lois de Commande pour le Controle

et la Mobilité de Machines Agricoles", Phd Thesis, LASMEA, 1996.
7. Dementhon D.F., L.S. Davis, "Model-Based Object Pose in 25 Lines of Code", International Journal of Computer Vison, vol. 15, n. 1-2, pp.123-141, June 1995.
8. Dhome M., M. Richetin, J.T. Lapresté, G. Rives, "Determination of the Attitude of 3D Objects from a Single Perspective Image", IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol 11, n.12, pp 1265-1278, December 1989.
9. Espiau B., F. Chaumette, P Rives, "A new approach to visual servoing in robotics", IEEE Trans. on Robotics and Automation, vol. 8, n.3, pp.313-326, June 1992.
10. Feddema J.T., O.R. Mitchell, "Vision-guided servoing with feature-based trajectory generation", IEEE Trans. on Robotics and Automation, vol. 5, n.5, pp.691-700, October 1989.
11. Grosso E., M. Meta, A. Andrea, G. Sandini, "Robust Visual Servoing in 3-D Reaching Tasks", IEEE Trans. on Robotics and Automation, vol. 12, n.5, pp.732-742, October 1996.
12. Hager G.D., S. Hutchinson, P. Corke, "Tutorial on Visual Servo Control", IEEE Int. Conf. on Robotics and Automation, Minneapolis, USA, 22-28 April, 1996.
13. Hashimoto K., "Visual Servoing", World Scientific, 1993.
14. Horaud R., B. Conio, O. Leboulleux, B. Lacoll, "An Analytic Solution For the Perspective 4 Point Problem", CVGIP, vol. 47, n. 1, pp. 33-44, 1989.
15. Horaud R., S. Christy, F. Dornaika, "Object Pose : the Link between Weak Perspective, Para-Perspective, and Full Perspective", Int. Journal of Computer Vision, vol. 22, n.2, pp 173-189, 1997.
16. Khadraoui D., G. Motyl, P. Martinet, J. Gallice , F. Chaumette, "Visual Servoing in Robotics Scheme Using a Camera/Laser-Stripe Sensor", IEEE Trans. on Robotics and Automation, vol. 12, n.5, pp.743-749, October 1996.
17. Lowe D.G., "Perceptual Organization and Visual Recognition", Kluwer Publisher, Boston, 1985.
18. Lowe D.G., "Robust Model-based Motion Tracking through the Integration of Search and Estimation", Int. Journal of Computer Vision, vol. 8, n. 2, pp 113-122, 1992.
19. Martinet P., F. Berry, J. Gallice. "Use of first derivative of geometric features in Visual Servoing", IEEE International Conference on Robotics and Automation, ICRA'96, Minneapolis, USA, April 1996.
20. Martinet P., D. Khadraoui, J. Gallice. "Vision Based Control Law using 3D Visual Features", World Automation Congress, Vol. 3, pp.497-502, Montpellier, France, May 1996.
21. Michel H., P. Rives, "Singularities in the determination of the situation of a robot effector from the perspective view of 3 points", Technical report n. 1850 INRIA, February 1993.
22. Motyl G., "Couplage d'une caméra et d'un faisceau laser en Commande Référencée Vision", Phd Thesis, LASMEA, Clermont-Ferrand, 1992.
23. Papanikolopoulos N., 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, vol. 9, n.1, pp.14-35, February 1993.
24. Samson C., M. Le Borgne, B. Espiau. "Robot Control : The Task Function Approach", Oxford University Press, 1991.
25. Sanderson A.C., L.E. Weiss. "Image-based visual servo control using relational graph error signals", Proc. IEEE Int. Conf. on Robotics and Automation, pp. 1074-1077, 1980.
26. Shakunaga T., H. Kaneko, "Perspective Angle Transform and Its Application to 3-D Configuration Recovery", Proc. of Int. Conf. on Computer Vision and Pattern Recognition, Miami Beach, Florida, pp 594-601, June 1986.
27. Wilson W.J., C. C. Williams Hulls, G.S. Bell. "Relative End-Effector Control Using Cartesian Position Based Visual Servoing", IEEE Trans. on Robotics and Automation, vol. 12, n.5, pp.684-696, October 1996.