Visual Servoing Based on Image Motion

Armel Crétual, François Chaumette

Visual servoing based on image motion.

Armel Crétual and François Chaumette

IRISA / INRIA Rennes

Campus de Beaulieu

35042 Rennes cedex, France

E-mail: {armel.cretual@irisa.fr, francois.chaumette@irisa.fr}

Abstract

The general aim of visual servoing is to control the motion of a robot in order that visual features acquired by a camera become superimposed with a desired visual pattern. Visual servoing based upon geometrical features such as image points coordinates is now well set on. Nevertheless, this approach has the drawback that it usually needs visual marks on the observed object to retrieve geometric features. The idea developed in this paper is to use motion in the image as input of the control scheme, since it can be estimated without any a priori knowledge of the observed scene. Thus, more realistic scenes or objects can be considered.

Two different methods are presented. In the first one, geometric features are retrieved by integration of motion,which allows to use classical control laws. This method is applied to a 6 d.o.f. positioning task. We show that, in such a case, an affine model of 2D motion is insufficient to ensure convergence and that a quadratic one is needed. In the second method, the principle is to try to obtain a desired 2D motion field in the image sequence. In usual image-based visual servoing, variations of visual features are linearly linked to the camera velocity. In our case, the corresponding relation is more complex and we describe how it is possible to use this relation. This approach is illustrated with two tasks: positioning a camera parallel to a plane, and trajectory following.

1 Introduction

Image-based visual servoing consists in designing closed-loop control schemes using visual features measured in the images acquired by a camera [12, 17, 18]. This approach has been applied for numerous tasks such as positioning with respect to an object [12], and target tracking [1, 13]. In many cases, the visual data involved in the control scheme correspond to geometric features. It usually imposes to use marked objects to extract these features at video rate. To avoid using such marked objects, a first improvement is to track points of interest. The success of the tracking needs these points to be significant, meaning corners where the spatial gradient is high in two orthoganal directions. This technique has been used in [15]. The main limitation is the very high sensitivity to occlusion of the points of interest. In [5], the visual features used are parameters of a model of the object contour. In that case, a partial and limited occlusion of the object is admitted. A last improvement is to add an estimation of the contour position using the motion in the image. This allows to robustify the previous approach. It has been done in [10] and [19], where a CAD model of the 3D object is needed. Two other studies, close one to the other, are presented in [3, 11]. They are based on the deformations in the image of the object of interest in order to position a camera with respect to this object. The 6 parameters of the affine deformation of a planar object is used, which reduces the complexity of the representation, compared to B-splines or a snakes. Even if this model is generally sufficient, there exist several limitations. For example, a singularity occurs when the object is parallel to the camera plane. Furthermore, the knowledge of 3D data, such as the orientation and the depth of the object, are needed in [3].

In this paper, we present a new approach whose principle is to rely on image motion measurement. More precisely, we will use as visual features the parameters of a polynomial model of motion. Numerous tasks can be defined using such kind of dynamic visual features, some of them being impossible to perform using geometric visual features. They can be divided into three groups depending on the aim of the considered task.

The first one is related to mobile target tracking. It is described in [7]. The second class of robotics tasks achieved using the motion in the image is the navigation of mobile robots. First of all, the image motion allows to compute the time-to-contact, as presented for example in [2]. In [9], the time-to-contact is used to avoid frontal collision. Moreover, several articles [4, 25] have dealt with navigation in a corridor. They are based on the idea that the perceived image motion from a robot moving between two parallel walls is symmetric on the two sides of the image, as soon as its trajectory is exactly on a line parallel to the walls, and equidistant from them. Finally, a third group can be built from constraints defined on the image motion. In [14], the aim of the task is to bring a robot manipulator in a 3D position defined by a target thanks to a camera observing both of them. In this case, the trajectory in the image is constrained to be rectilinear controlling the projected speed of the manipulator. In [28], the aim is to bring the optical axis of a moving camera parallel to its direction of translation. It is done by constraining the constant parameters of the apparent speed.

Nevertheless, all these previous works deal with a particular task. In the present article, we present a general method applicable to a large class of tasks. After the presentation of the image motion model in Section 2, two different methods are developed. The first one uses the idea that geometric features can be retrieved by integrating 2D motion. It is then possible to apply classical visual servoing techniques. This principle is presented in Section 3. An application of this method, related to the positioning of a camera with respect to a complex scene is presented in Section 4. The principle of the second method is to introduce the motion parameters directly in the control loop. It necessitates to determine the relation between the derivative of the motion parameters and the camera motion. From this relation, different control schemes are proposed in Section 5. Finally, these ideas are applied in Section 6 for two different robotics tasks whose aim are, respectively, to position the camera parallel to a plane and to follow a trajectory to observe a surface.

2 Image motion model

Our aim is to control the motions of a robot from visual features without any a priori knowledge on the image content. Both methods presented below rely on motion in the image since it is not dependent on the image content. A motion model is used to approximate the apparent dense motion field in the image. This model is a simplified quadratic one with 8 parameters:

$$\left\{ \begin{array}{rcl} \dot{x} & = & c_1 + a_1 x + a_2 y + q_1 x^2 + q_2 xy \\ \dot{y} & = & c_2 + a_3 x + a_4 y + q_3 y^2 + q_4 xy \end{array} \right. \tag{1}$$

Let us note that the model presented above perfectly represents the motion field when rigid motions of a planar object are considered. In that case, we have [27, 21] :

$$\left\{ \begin{array}{ll} c_1 = -\frac{T_x}{Z_p} - \Omega_y & c_2 = -\frac{T_y}{Z_p} + \Omega_x \\ a_1 = \gamma_1 \frac{T_x}{Z_p} + \frac{T_z}{Z_p} & a_2 = \gamma_2 \frac{T_x}{Z_p} + \Omega_z \\ a_3 = \gamma_1 \frac{T_y}{Z_p} - \Omega_z & a_4 = \gamma_2 \frac{T_y}{Z_p} + \frac{T_z}{Z_p} \\ q_1 = q_4 = -\gamma_1 \frac{T_z}{Z_p} - \Omega_y & q_2 = q_3 = -\gamma_2 \frac{T_z}{Z_p} + \Omega_x \end{array} \right. \tag{2}$$

where $T = (T_x, T_y, T_z)$ and $\Omega = (\Omega_x, \Omega_y, \Omega_z)$ respectively represent the translational and the rotational components of the kinematic screw between the camera frame and the object frame, $Z = Z_p + \gamma_1 X + \gamma_2 Y$ being the equation that describes the 3D position and orientation of the plane to which belongs the object with respect to the camera frame.

In all other cases (*i.e.* non planar object, non rigid motion), the model given in (1) is only an approximation of the real motion field. Of course, other models could also be used, e.g. constant (the restriction of the presented one to terms c_i) or affine (the restriction to terms c_i and a_i). In fact, there is a necessary compromise to find between accuracy of the approximation and computation load.

Motion parameters are computed using the robust multi-resolution estimation method (RMR algorithm) described in [22]. A Gaussian image pyramid is constructed for each new image acquired by the camera. Let Θ be the vector of the motion model parameters at instant t. On the coarsest level of the pyramid, the first estimation of Θ consists in minimizing with respect to Θ the following criterion:

$$C(\Theta) = \sum_p \psi(r(p,\Theta)) \quad \text{with } r(p,\Theta) = \nabla I(p,t).w_\Theta(p) + I_t(p,t)$$

where $I(p,t)$ is the intensity level at pixel p, ∇I and I_t its spatial gradient and temporal derivative, $w_\Theta(p)$ is the velocity vector at pixel p provided by Θ, ψ is a robust estimator, typically Tukey's biweight function. This estimator allows to detect and reject outliers, i.e. pixels p whose spatio-temporal gradients do not correspond to the current estimation of Θ. Once an outlier is detected, a weight equal to zero is associated to it.

Then, a hierarchical and iterative strategy is used. Let Θ_i^k be the estimate of Θ at level i and k-th iteration at this level. We have $\Theta_i^k = \Theta + \Delta\Theta_i^k$. Successive incremental refinements $\Delta\Theta_i^k$ are given by:

$$\Delta\Theta_i^k = \arg\min_{\Delta\Theta_i^k} \sum_p \psi(r'(\Delta\Theta_i^k))$$

with $r'(\Delta\Theta_i^k) = \nabla I(p + w_{\Theta_i^k}(p), t+1).w_{\Delta\Theta_i^k}(p) + I(p + w_{\Theta_i^k}(p), t+1) - I(p,t)$.

Once the iterative estimation Θ_i of Θ is performed at level i using the incremental optimization process, the estimation at level $i+1$ is initialized by the projection of Θ_i on this finer resolution level, and this hierarchical scheme is driven up until reaching the finest resolution level.

If needed, the estimation can be performed only on a part of the image. In this case, one only has to set as outliers all the pixels of the image outside the considered part. It means in fact giving them an initial weight equal to zero.

3 Visual servoing from integration of 2D motion

A visual servoing scheme can be divided into two steps: the extraction of visual features and the control scheme. The main idea of this first method is to combine classical 2D visual servoing control laws and the estimation of geometric visual features (necessary as input of such control laws) from the integration along time of 2D motion.

3.1 Retrieval of the 2D features

Let us denote $s = (x, y)^T$ the 2D projection at time t of a 3D point M, and $\dot{s}$ its apparent speed in the image. s can obviously be recovered knowing its position s_0 at time 0 and the evolution of $\dot{s}$ over time, by:

$$\begin{aligned} s &= s_0 + \int_0^t \dot{s}\ dt \qquad \text{(in continuous form)} \\ s &= s_0 + \sum_{i=1}^{k} \dot{s}_i\ \delta t_i \qquad \text{(in discrete form)} \end{aligned} \tag{3}$$

where $\dot{s}_i$ is the i$^{\text{th}}$ measure of $\dot{s}$ and δt_i is the time duration between instants $(i-1)$ and i, provided by the computer clock. Of course, the motion estimation algorithm described in Section 2 is used to estimate $\dot{s}_i$.

Moreover, s can be, not only a single point, but a set of n points of the image. The real 2D-coordinates of these points on the i-th image will be denoted s_i and their estimations σ_i, obtained from (3), with:

$$\begin{aligned} s_i = &\quad (x_{1,i}, \ldots, x_{n,i}, y_{1,i}, \ldots, y_{n,i})^T \\ \sigma_i = &\quad (\xi_{1,i}, \ldots, \xi_{n,i}, \psi_{1,i}, \ldots, \psi_{n,i})^T \end{aligned} \tag{4}$$

Finally, we have to note that, since successive measurements are integrated along time, a potential drift may appear after a certain time if these measurements are biased. Experiments such as the ones presented in the following section, and those given in [7], show that the motion estimation we used is unbiased, which implies that no drift appears in practice.

3.2 Control law

Having an estimation of the position of one or several points in the image, it is possible to define a control law in order to bring them to a desired position.

Vector s is linked to the camera motions by the interaction relation [12, 18]:

$$\dot{s} = L\ V_c + \frac{\partial s}{\partial t} \tag{5}$$

where V_c is the vector of controlled camera motions (the whole set or a sub-set of the velocity screw components), L is the interaction matrix related to s (L is also called image Jacobian) and $\frac{\partial s}{\partial t}$ represents the variation of s due to a potential own motion of the target.

Let s^* be the desired positions of the points. A task function is defined as $e = C(s - s^*)$, where C is a constant chosen matrix. An exponential decay of e leads to the control law (see [12]):

$$V_c = -\lambda\ \left(C\ \widehat{L}\right)^{-1}\ e - \left(C\ \widehat{L}\right)^{-1}\ \widehat{\frac{\partial e}{\partial t}} \tag{6}$$

where $\widehat{L}$ is an approximation of L (see [12]) and $\widehat{\frac{\partial e}{\partial t}}$ is an estimation of $\frac{\partial e}{\partial t}$. C is usually chosen equal to $\widehat{L}^+$ where $\widehat{L}^+$ is the pseudo-inverse of $\widehat{L}$. The control law is thus reduced to:

$$V_c = -\lambda\ e - \widehat{\frac{\partial e}{\partial t}}$$

Replacing this value into equation (5) allows to describe the closed-loop behaviour of the system. It leads to the relation:

$$\dot{e} = -\lambda \left(\widehat{L}^+ L\right) e - \underbrace{\left(\left(\widehat{L}^+ L\right) \widehat{\frac{\partial e}{\partial t}} - \frac{\partial e}{\partial t}\right)}_{b} \tag{7}$$

The second term of equation (7), denoted b, represents the amplitude of the lag involved by a wrong estimation of $\frac{\partial e}{\partial t}$ in case of a mobile target tracking task. Furthermore, the global asymptotic stability is ensured under the well-known sufficient condition [12]:

$$\widehat{L}^+ L > 0 \tag{8}$$

Under that condition, $\|e\|$ will decrease towards $\left\|\frac{1}{\lambda}\left(\widehat{L}^+ L\right)^{-1} b\right\|$. Therefore, if the estimation error b is bounded, the error e will also be bounded, and the closer b is to zero, the smallest is the error e. We can finally note that, if we are interested in positioning task, the error e decreases to 0 since we have $\widehat{\frac{\partial e}{\partial t}} = \frac{\partial e}{\partial t} = 0$.

4 Application to positioning

In this section, we present a task whose aim is to position the camera with respect to an observed scene, without a priori knowledge upon the scene appearance. We will see that even if it seems that 6 motion parameters are sufficient to ensure the convergence, 8 motion parameters are in fact necessary.

4.1 Control law

To define a control law related to the positioning of camera using a visual servoing scheme, a set of three points is not sufficient to control 6 d.o.f. (3 rotations and 3 translations). Indeed, it has been shown in [20] that the system can reach singularities, in particular when the optical center lies on the surface of a particular cylinder (defined by the circle circumscribed to the points, and with a direction orthogonal to the plane of the triangle). Moreover, to one position of three points in the image correspond four positions of the camera. Therefore, a redundant information is necessary, for example based on a fourth point.

There are only two conditions for the choice of the points of interest: they should appear both in the desired and in the initial images and the corresponding interaction matrix has to be of full rank 6. In practice, to ensure a good observation of the scene deformations, the points are chosen sufficiently far one from the other. Moreover, the initial matching between points on the desired and initial images is performed semi-automatically. This means that an extraction of several characteristic points is made in both images. This extraction is performed using the classical Harris and Stephens method [16]. Then, the operator chooses four of them (see Figure 1). Let us note here that the precision of the extraction is only around one pixel. We have also to note that these points are used only to compute the initial error. Since their position is then estimated from the global motion of the scene, there is no need to track them along the realization of the task. Therefore, they can be occluded without disturbing the control, even at convergence. They can even go out of the camera field of view during the servoing.

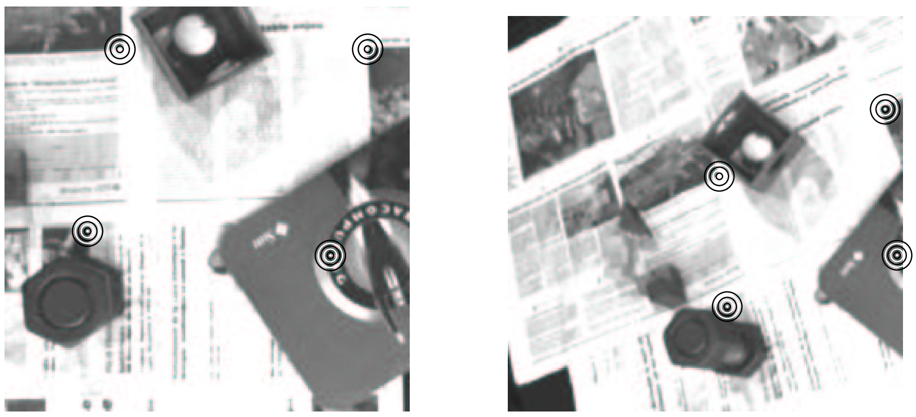


Figure 1: Observed scene at desired position (a) and at initial one (b)

Let s be the vector composed of the coordinates of the four points of interest, and s^* be their desired position. Since the scene is assumed to be motionless, the control law resulting from (6) is given by:

$$V_c = -\lambda \, e = -\lambda \, \widehat{L}^+ \, (s - s^*) \tag{9}$$

where the well-known form of the interaction matrix related to s is given by [12, 18]:

$$L = \begin{bmatrix} -1/Z_1 & 0 & x_1/Z_1 & x_1 y_1 & -1 - x_1^2 & y_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -1/Z_4 & 0 & x_4/Z_4 & x_4 y_4 & -1 - x_4^2 & y_4 \\ 0 & -1/Z_1 & y_1/Z_1 & 1 + y_1^2 & -x_1 y_1 & -x_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & -1/Z_4 & y_4/Z_4 & 1 + y_4^2 & -x_4 y_4 & -x_4 \end{bmatrix}$$

In [12], it is shown that, if the initial error is not too large, the estimation of L by its value at convergence using a coarse estimation $\widehat{Z}_i$ of the depth Z_i of each point provides satisfactory results. That is why we have used $\widehat{L} = L_{s=s^*, Z_i = Z_i^*}$ where values Z_i^* are given off line.

However, in our case, the error is in fact not equal to $(s - s^*)$ but to $(\sigma - s^*)$ (refer to equation (4)). This difference has an influence on the true interaction relation between used measurements and the camera motion, and therefore on the stability. If we write $\dot{\sigma} = L_\sigma V_c$, the stability condition is in fact:

$$\widehat{L}^+ L_\sigma > 0 \tag{10}$$

We now demonstrate that this condition explains why an affine model is inadequate in some cases.

4.2 Motion model: affine vs. quadratic

As specified in Section 3.1, the image motion is approximated using a polynomial model to retrieve the position of each point. An important question is to define which model should be used since there is a compromise to find between the swiftness of the estimation and its accuracy. In our case, six d.o.f. of the robot are constrained. Therefore, a constant model of motion is here heavily insufficient.

The affine model includes six parameters. Nevertheless, it is not enough to ensure the correct positioning. Actually, using such a model to estimate the points positions does not allow to make the distinction between a translation along $\overrightarrow{x}$ (resp. along $\overrightarrow{y}$) and a rotation around $\overrightarrow{y}$ (resp. around $\overrightarrow{x}$). More precisely, in that case the link between $\dot{\sigma}$ and the camera motion is given by the following matrix $L_{\sigma a}$:

$$L_{\sigma a} = \begin{bmatrix} -1/Z_1 & 0 & \xi_1/Z_1 & 0 & -1 & \psi_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -1/Z_4 & 0 & \xi_4/Z_4 & 0 & -1 & \psi_4 \\ 0 & -1/Z_1 & \psi_1/Z_1 & 1 & 0 & -\xi_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & -1/Z_4 & \psi_4/Z_4 & 1 & 0 & -\xi_4 \end{bmatrix}$$

It can never be ensured that the rank of this matrix is always 6. Therefore, if the rank is inferior to 6, the stability condition (10) cannot be respected, since the product $L_\sigma \widehat{L}^+$ must be srictly positive. In particular, when all the Z_i values are equal, meaning when the camera is parallel to the scene, the rank of this matrix is only 4. In such a case, combinations of T_x and Ω_y (resp. T_y and Ω_x) appear in the null space of $L_{\sigma a}$. This explains the singularity encountered in [3, 11].

On the contrary, if the considered model of motion is the quadratic one, the corresponding matrix $L_{\sigma q}$

is equal to:

$$L_{\sigma q} = \begin{bmatrix} -1/Z_1 & 0 & \xi_1/Z_1 & \xi_1\psi_1 & -1-\xi_1^2 & \psi_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -1/Z_4 & 0 & \xi_4/Z_4 & \xi_4\psi_4 & -1-\xi_4^2 & \psi_4 \\ 0 & -1/Z_1 & \psi_1/Z_1 & 1+\psi_1^2 & -\xi_1\psi_1 & -\xi_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & -1/Z_4 & \psi_4/Z_4 & 1+\psi_4^2 & -\xi_4\psi_4 & -\xi_4 \end{bmatrix}$$

The rank of this matrix is always 6. Therefore, we can conclude that it is necessary to use this model.

4.3 Results

The considered scene is composed of a main plane with several 3D objects laying on it (see Figure 1). It is a "real" one in the sense that no dedicated object has been added. The desired position is such that the main plane is parallel to the image plane, which corresponds to a singularity for the affine model.

Images were acquired with a SunVideo board and all the computations was made on a 170 MHz Ultra-Sparc station. The size of the images was 256×256 pixels which leads to 500 ms processing at each iteration using an affine model and 800 ms using a quadratic one.

4.3.1 Affine vs. quadratic model

The aim of these first experiments is to prove the necessity of using the quadratic model of motion. Different initial positions have been considered corresponding to simple motions of the camera, meaning a displacement on only one d.o.f.. Results concerning these experiments are displayed on Table 1. In the first column, the displacement between the initial position and the desired one is given in a fixed frame. The desired position corresponds to the position $\tau = (0,0,0)$ and the orientation $\omega = (0,0,0)$ in this frame. The second and third column gives the final position and orientation reached by the camera using respectively the affine and the quadratic model of motion. Positions are given in mm and orientation in deg.

We first can notice that the quadratic model gives strongly better results than the affine one. In the first case, the average error is around 10 mm in position and less than 1 deg in orientation. On the contrary, with the affine model, these errors reach more than 170 mm in position and 12 deg in rotation, and a divergence is even encountered in case of a rotational motion around $\overrightarrow{x}$. One can also notice that, when the affine model is used, an error in translation along $\overrightarrow{x}$ is always combined with an error in rotation around $\overrightarrow{y}$ and vice versa. This proves experimentally the necessity of using the quadratic model of motion. The small

Initial motion	Final position (affine model)	Final position (quadratic model)
T_x = + 100 mm	τ = (-36, +133, +20) ω = (+9.2, +2.1, 0.0)	τ = (-9, +15, +7) ω = (+1.3, +1.0, +0.2)
Ω_x = + 10 deg	divergence (see note) τ = (+64, -358, +276) ω = (-25.6, -4.6, +1.0)	τ = (-5, +12, +5) ω = (+0.9, +0.4, -0.1)
T_z = - 150 mm	τ = (-98, -50, +11) ω = (-3.7, +6.8, +0.6)	τ = (+13, -8, 0) ω = (-0.6, +0.8, +0.4)
Ω_z = + 30 deg	τ = (-84, +178, +34) ω = (+12.5, +5.8, -0.1)	τ = (-22, +6, -2) ω = (0.0, +1.4, +0.5)

Table 1: Comparison of positions at convergence using the affine or the quadratic model of motion. Note : *In case of divergence, the given values correspond to the position of the camera when a joint limit is reached*

residual errors can be easily explained by the weak precision of the initial extraction of the points (about 1 pixel whereas it is about a tenth of a pixel with dedicated objects).

4.3.2 Results for a large displacement

The aim of the following experiment is to show the accuracy of our method, even when the initial error is large. More precisely, the difference between the initial and desired positions of the camera was T = (300, 350, -150) (in mm), Ω = (25, -20, 25) (in deg). It corresponds to the images given in Figure 1. In that case, only the quadratic model has been used.

The obtained results are displayed on Figure 2. First, one can notice that despite the large error at the beginning (nearly a hundred pixels), the convergence is obtained. It is performed without oscillations and it remains stable once convergence is reached. One image upon ten of the sequence acquired during the task is displayed on Figure 3. On each image of the sequence, the position of the points used in the control scheme are designated by a target sign. It can be seen that they do not all correspond to corners with a high spatial gradient. Moreover, on several of these images, some of them have been occluded. For example, in the image at the upper right, three of the four points are occluded. This only leaves one point visible, which would be highly insufficient for techniques based on points of interest tracking. Comparing the last image of the sequence, meaning the one obtained at convergence, to the desired one presented on Figure 1, one can obviously notice that they are very close.

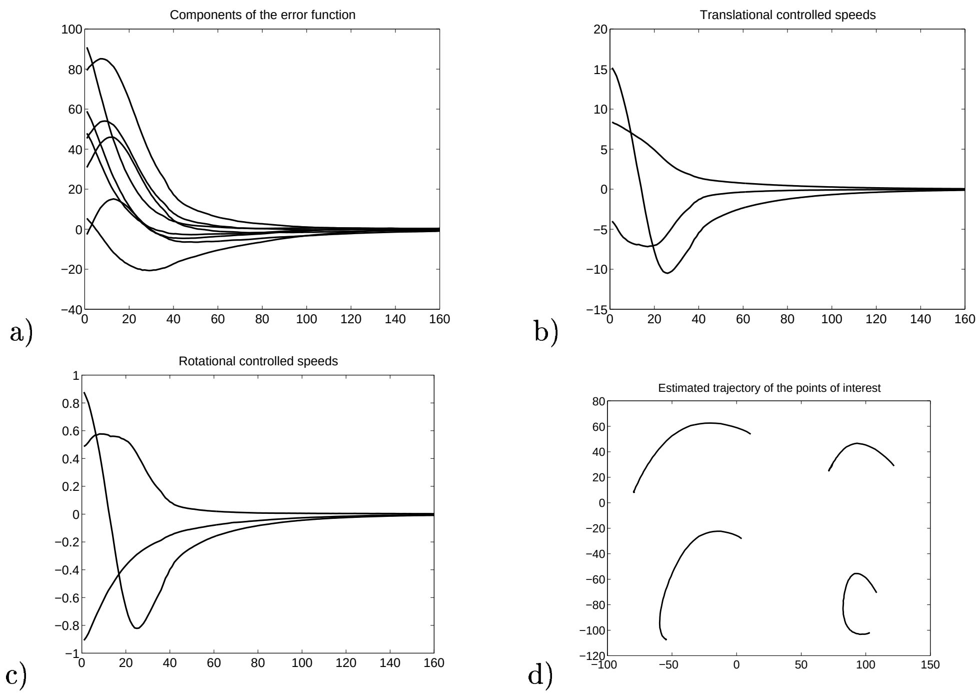


Figure 2: Positionning result: a) $s - s^*$ (in pixel) versus iteration number, b) T_x, T_y, T_z (in mm/s), c) $\Omega_x, \Omega_y, \Omega_z$ (in deg/s), d) trajectory of the four points of interest in the image

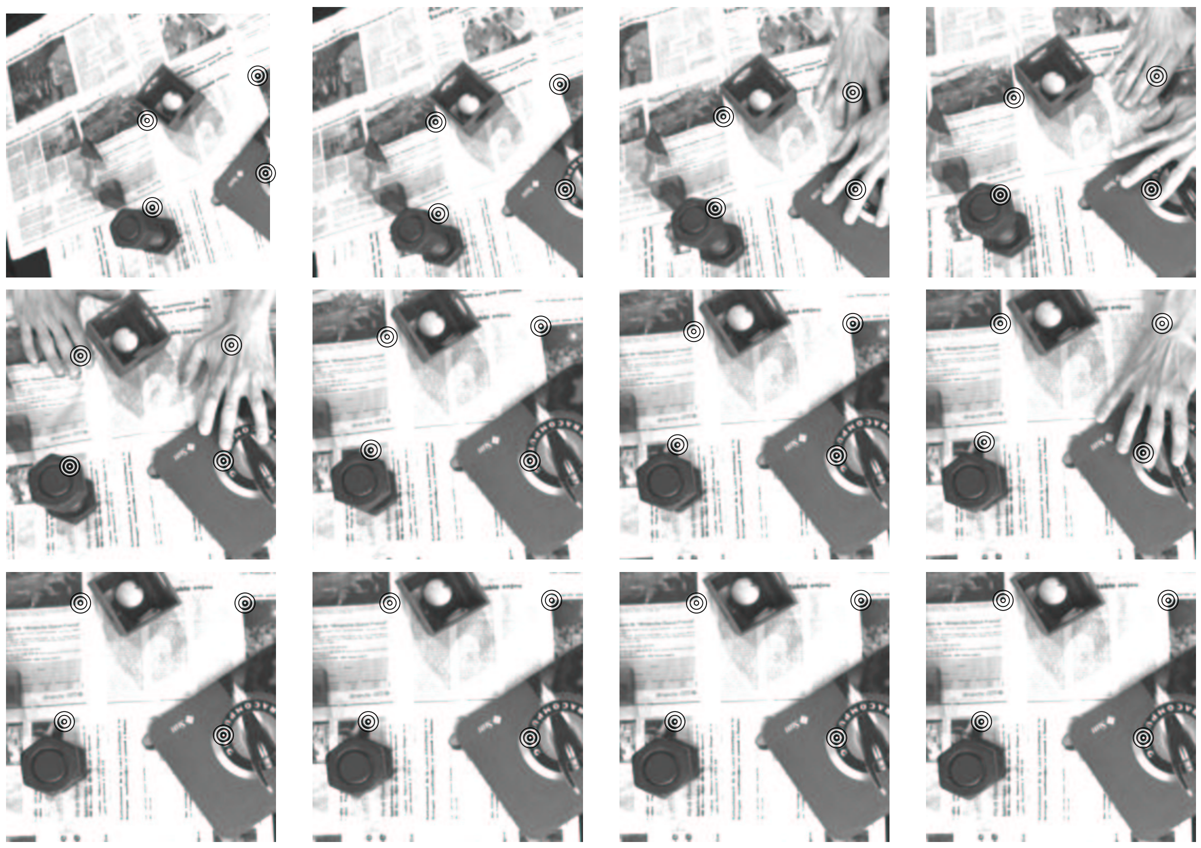


Figure 3: One image upon ten of the sequence with the estimated position of the points of interest marked by a target symbol

The main advantage of our method is that the matching between the current points and the desired ones has to be performed only once, at the beginning of the servoing. Therefore, there is no need to track the points of interest in the image at each iteration. Failure mode of our method is when an occlusion of the major part of the scene occurs. Indeed, the RMR algorithm can only support approximately 50% outliers. However, in that case, all the existing approaches would also fail.

5 2D motion-based visual servoing

In this second approach, the visual specification of the desired configuration is no more done with geometrical constraints, but with dynamic criteria, i.e. homogeneous to speed in the image. More precisely, we wish to control the camera motions in order that the current motion field in the image, such as the one presented in Figure 4.a, becomes equal to a desired one, such as, for example, the divergent field of Figure 4.b.

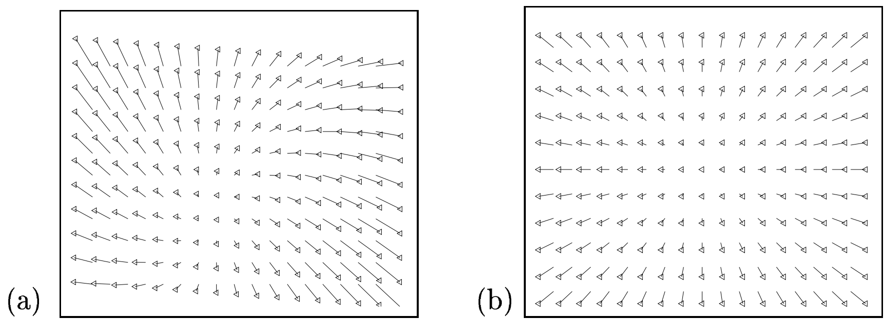


Figure 4: Current (a) and desired (b) 2D motion fields

To control the system, we must quantify its state to be able to compare it to a desired configuration. Concerning the dense field of motion (i.e. the displacement of each pixel between 2 images), even for a small image, the vector of visual features is too important to be used in practice. Moreover, computation of such field is often impossible in a real-time context. On the contrary, a binary map separating pixels in motion from motionless ones does not bring enough information on the state of the system, in particular because this information is not continuous.

Therefore, we use the polynomial model of motion defined by (1) to describe and to specify the behavior of the system. This choice is emphasized by the strong relation between the parameters of this model and particular motions of the camera.

5.1 Interaction relations associated to motion parameters

From (2), we can note that each term p_i of the quadratic model is a linear function of the kinematic screw $V = (T^T, \Omega^T)^T$:

$$p_i = L_{p_i}(r)\ V \tag{11}$$

respectively given by:

$$\begin{array}{ll}
L_{c_1} = \left[\frac{1}{Z_p},\ 0,\ 0,\ 0,\ 1,\ 0\right] & L_{c_2} = \left[0,\ \frac{1}{Z_p},\ 0,\ -1,\ 0,\ 0\right] \\
L_{a_1} = \left[-\frac{\gamma_1}{Z_p},\ 0,\ -\frac{1}{Z_p},\ 0,\ 0,\ 0\right] & L_{a_2} = \left[-\frac{\gamma_2}{Z_p},\ 0,\ 0,\ 0,\ 0,\ -1,\ \right] \\
L_{a_3} = \left[0,\ -\frac{\gamma_1}{Z_p},\ 0,\ 0,\ 0,\ 1\right] & L_{a_4} = \left[0,\ -\frac{\gamma_2}{Z_p},\ -\frac{1}{Z_p},\ 0,\ 0,\ 0\right] \\
L_{q_1} = \left[0,\ 0,\ \frac{\gamma_1}{Z_p},\ 0,\ 1,\ 0\right] & L_{q_2} = \left[0,\ 0,\ \frac{\gamma_2}{Z_p},\ -1,\ 0,\ 0\right]
\end{array} \tag{12}$$

Using these linear relations, the time variation of parameter p_i can be written as:

$$\dot{p}_i = L_{p_i}(r)\ \dot{V} + \left(\frac{\partial L_{p_i}(r)}{\partial r}\dot{r}\right)^T V$$

Noting that $\dot{r} = V$, the previous relation is thus the sum of a linear relation with respect to acceleration and a quadratic form of the kinematic screw:

$$\dot{p}_i = L_{p_i}\Gamma + V^T\ Q_{p_i}\ V \tag{13}$$

where:

- Γ is the relative acceleration between the camera and the scene, expressed in the camera frame, i.e. $\Gamma = (\dot{T}_x,\ \dot{T}_y,\ \dot{T}_z,\ \dot{\Omega}_x,\ \dot{\Omega}_y,\ \dot{\Omega}_z)^T$;
- L_{p_i} is given by (12);

- Q_{p_i} is given by (see Appendix or [6]):

$$Q_{c_1} = \begin{bmatrix} \frac{\gamma_1}{Z_p^2} & \frac{\gamma_2}{Z_p^2} & \frac{-1}{Z_p^2} & \frac{-\gamma_2}{Z_p} & \frac{\gamma_1}{Z_p} & 0 \\ 0 & 0 & \cdots & & \cdots & 0 \\ \vdots & \ddots & \ddots & & & \vdots \\ \vdots & & \ddots & \ddots & & \vdots \\ \vdots & & & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & 0 & 0 \end{bmatrix} \qquad Q_{c_2} = \begin{bmatrix} 0 & \frac{\gamma_1}{Z_p^2} & 0 & 0 & 0 & 0 \\ 0 & \frac{\gamma_2}{Z_p^2} & \frac{-1}{Z_p^2} & \frac{-\gamma_2}{Z_p} & \frac{\gamma_1}{Z_p} & 0 \\ \vdots & \ddots & 0 & \cdots & \cdots & 0 \\ \vdots & & \ddots & \ddots & & \vdots \\ \vdots & & & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & 0 & 0 \end{bmatrix} \tag{14}$$

$$Q_{a_1} = \begin{bmatrix} \frac{-\gamma_1^2}{Z_p^2} & \frac{-\gamma_1\gamma_2}{Z_p^2} & 0 & 0 & \frac{1}{Z_p} & \frac{\gamma_2}{Z_p} \\ 0 & 0 & \frac{-\gamma_2}{Z_p^2} & 0 & 0 & 0 \\ \vdots & 0 & \frac{1}{Z_p^2} & \frac{\gamma_2}{Z_p} & \frac{-\gamma_1}{Z_p} & 0 \\ \vdots & & \ddots & 0 & \cdots & 0 \\ \vdots & & & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & 0 & 0 \end{bmatrix} \qquad Q_{a_2} = \begin{bmatrix} \frac{-\gamma_1\gamma_2}{Z_p^2} & \frac{-\gamma_2^2}{Z_p^2} & \frac{\gamma_2}{Z_p^2} & \frac{-1}{Z_p} & 0 & \frac{-\gamma_1}{Z_p} \\ 0 & 0 & \cdots & \cdots & \cdots & 0 \\ \vdots & \ddots & \ddots & & & \vdots \\ \vdots & & \ddots & \ddots & & \vdots \\ \vdots & & & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & 0 & 0 \end{bmatrix}$$

$$Q_{a_3} = \begin{bmatrix} 0 & \frac{-\gamma_1^2}{Z_p^2} & 0 & 0 & 0 & 0 \\ 0 & \frac{-\gamma_1\gamma_2}{Z_p^2} & \frac{\gamma_1}{Z_p^2} & 0 & \frac{1}{Z_p} & \frac{\gamma_2}{Z_p} \\ \vdots & 0 & 0 & \cdots & \cdots & 0 \\ \vdots & & \ddots & 0 & \cdots & 0 \\ \vdots & & & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & 0 & 0 \end{bmatrix} \qquad Q_{a_4} = \begin{bmatrix} 0 & \frac{-\gamma_1\gamma_2}{Z_p^2} & \frac{-\gamma_1}{Z_p^2} & 0 & 0 & 0 \\ 0 & \frac{-\gamma_2^2}{Z_p^2} & 0 & \frac{-1}{Z_p} & 0 & \frac{-\gamma_1}{Z_p} \\ \vdots & 0 & \frac{1}{Z_p^2} & \frac{\gamma_2}{Z_p} & \frac{-\gamma_1}{Z_p} & 0 \\ \vdots & & \ddots & 0 & \cdots & 0 \\ \vdots & & & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & 0 & 0 \end{bmatrix}$$

$$Q_{q_1} = \begin{bmatrix} 0 & 0 & \frac{\gamma_1^2}{Z_p^2} & 0 & 0 & 0 \\ 0 & 0 & \frac{\gamma_1\gamma_2}{Z_p^2} & 0 & 0 & 0 \\ \vdots & 0 & \frac{-\gamma_1}{Z_p^2} & 0 & \frac{-1}{Z_p} & \frac{-\gamma_2}{Z_p} \\ \vdots & & \ddots & 0 & \dots & 0 \\ \vdots & & & \ddots & \ddots & \vdots \\ 0 & \dots & \dots & \dots & 0 & 0 \end{bmatrix} \quad Q_{q_2} = \begin{bmatrix} 0 & 0 & \frac{\gamma_1\gamma_2}{Z_p^2} & 0 & 0 & 0 \\ 0 & 0 & \frac{\gamma_2^2}{Z_p^2} & 0 & 0 & 0 \\ \vdots & 0 & \frac{-\gamma_2}{Z_p^2} & \frac{1}{Z_p} & 0 & \frac{\gamma_1}{Z_p} \\ \vdots & & \ddots & 0 & \dots & 0 \\ \vdots & & & \ddots & \ddots & \vdots \\ 0 & \dots & \dots & \dots & 0 & 0 \end{bmatrix}$$

5.2 Control schemes

The principle of 2D motion-based visual servoing is sum up on Figure 5. Unlike the geometric case, the measure s as well as its desired value s^* are now functions of parameters p_i of the 2D motion model. Let us note that the variations of these parameters depend now not only on the kinematic screw V but also on the acceleration Γ.

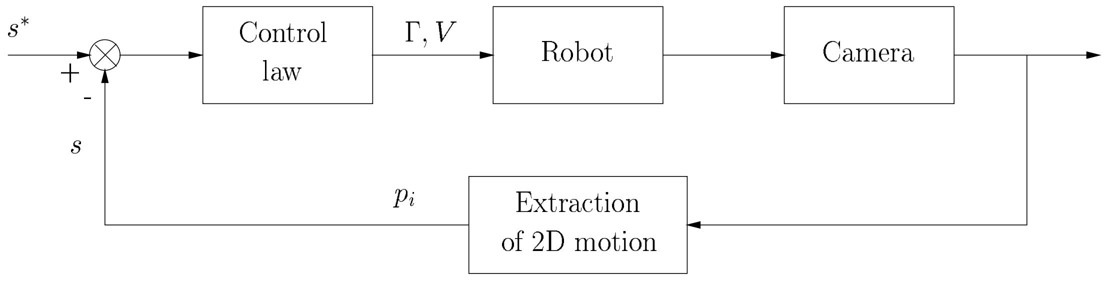


Figure 5: Block diagram of 2D motion-based visual servoing

Let us consider a vector s composed of n dynamic visual features. An element of s can be either one of the parameters of the quadratic model of motion, or a term obtained by difference between one of these parameters and a component of the camera velocity (assumed to be measurable by odometry). We will see further the interest of such combinations. We consider a task whose aim is to bring this vector to a desired value s^*. Using the relations (13), we can write:

$$\dot{s} = L\Gamma + f(V, r) \tag{15}$$

where:

- the matrix L of dimension $n \times 6$ is composed of each of the row matrices linking each element s_i of s to Γ and is called the interaction matrix associated to s;

- $f(r, V)$ is a vectorial function whose each component is a quadratic form associated to the corresponding element of s by the relations (13).

If the number of controlled axes is less than 6, Γ and V can be replaced by the sub-vectors corresponding to the considered axes. In the following, we will always consider that 6 axes are controlled without loss of generality.

To control the evolution of the system, an action is done on the camera motions. Since we have:$\Gamma = \Gamma_o - \Gamma_c$ and $V = V_o - V_c$ where indices o and c respectively stand for motions of the object and of the camera with respect to a static frame, the interaction relation can be written:

$$\dot{s} = -L\Gamma_c - f(V_c, r) + g_o(\Gamma_o, V_o, r) \tag{16}$$

where $g_o(\Gamma_o, V_o, r)$ is a function of the object own motion. We can also write this relation by grouping the terms depending on the object own motion under the form $\frac{\partial s}{\partial t}$. We obtain:

$$\dot{s} = -L\Gamma_c - f(V_c, r) + \frac{\partial s}{\partial t} \tag{17}$$

Once again, the previous relation is written in the case where the 6 camera d.o.f. are controlled. In any other case where Γ_c and V_c are only sub-vectors of the camera acceleration and speed, the influence of the non-constrained components also appear in $\frac{\partial s}{\partial t}$.

Let $e = C(s - s^*)$ be the error function associated to the task, with C a constant matrix of size $6 \times n$ to be defined. The objective of the control is to bring each component of e to a zero value. If we apply a gradient-like control law, imposing an exponential decay of e with gain λ ($\dot{e} = -\lambda e$), we have:

$$\lambda e = CL\;\Gamma_c + C\;f(V_c, r) - \frac{\partial e}{\partial t} \quad \text{with } \frac{\partial e}{\partial t} = C\frac{\partial s}{\partial t} \tag{18}$$

We thus have to solve the **integro-differential equation** (18), i.e. from measurements of e we have to determine the acceleration or the speed of the camera which will allow to obtain the desired behavior. This equation is said to be integro-differential because it depends on the choice to control in acceleration or speed. In the first case, the equation is integral as the speed is, by definition, the integral function of the acceleration, and in the other case, we have to solve a differential equation. Nevertheless, this equation is strongly non-linear, as the function f is quadratic. Therefore, instead of trying to solve such an equation, we propose to linearize it to determine the control vector. Even if equation (18) is non-linear, we can note that there is a linear relation between $\dot{e}$ and Γ. We now consider the different possible cases that may occur in function of the selected visual features.

5.3 Case of a full rank interaction matrix

Control law

In this paragraph, we consider that L is of full rank 6. In that case, C can be chosen such that CL is invertible. More precisely, if L is a 6×6 matrix, C is equal to $\mathbb{I}_6$. Otherwise, C is chosen equal to $\widehat{L}^+$, the pseudo-inverse of an estimation of L. A control law in acceleration can then be designed, and we obtain:

$$\Gamma_c = (CL)^{-1}\left(\lambda\, e - Cf(V_c, r) + C\frac{\partial e}{\partial t}\right) \tag{19}$$

In this equation, V_c is either measured by odometry, either considered as a tracking error which has to be estimated. Furthermorem, it is often difficult to obtain a precise measure of L. That is why we use an estimation $\widehat{L}$. Similarly functions f and $\frac{\partial e}{\partial t}$ are estimated respectively by $\widehat{f}$ and $\widehat{\frac{\partial e}{\partial t}}$. The previous relation can then be written:

$$\Gamma_c = \left(C\widehat{L}\right)^{-1}\left(\lambda\, \widehat{e} - C\widehat{f}(V_c, r) + C\widehat{\frac{\partial e}{\partial t}}\right) \tag{20}$$

where $\widehat{e}$ is the measurement of e given by the motion estimation algorithm.

Stability analysis

Let K be the matrix $(CL)(C\widehat{L})^{-1}$. The closed loop behavior of the system is described by:

$$\dot{e} = -\lambda K\widehat{e} + KC\left(\widehat{f} - \widehat{\frac{\partial e}{\partial t}}\right) - C\left(f - \frac{\partial e}{\partial t}\right) \tag{21}$$

With such a relation, it is not obvious to determine conditions ensuring the error decay. Nevertheless, by supposing each of the estimations made to design the control law is accurate enough, the behavior of the system can be at least brought close to the desired one. More precisely, if we denote:

$$\widehat{e} = e + b_e, \qquad \widehat{f} = f + b_f \quad \text{and} \quad \widehat{\frac{\partial e}{\partial t}} = \frac{\partial e}{\partial t} + b_g$$

where b_e, b_f and b_g are noise functions respectively on e, f and $\frac{\partial e}{\partial t}$, the relation (21) can be written:

$$\dot{e} = -\lambda K e - \lambda K b_e + KC(b_f - b_g) + (K - \mathbb{I})C\left(f - \frac{\partial e}{\partial t}\right) = -\lambda K e + b$$

where b stands for the global noise on the system.

If we admit the noise b is bounded, it is possible to prove, as in [24], that the positivity of K (here close to $\mathbb{I}$) ensures the decay of the error $\|e\|$ to a neighborhood, also bounded, around the $\left\|\frac{K^{-1}b}{\lambda}\right\|$ value.

Estimation of f and $\frac{\partial e}{\partial t}$

We are not interested here in estimating the two functions f and $\frac{\partial e}{\partial t}$ separately, but directly the term $f - \frac{\partial e}{\partial t}$ that appears in control law (19). Equation (17) can be written:

$$\dot{s} = -L\Gamma_c - f + \frac{\partial e}{\partial t} = -L\Gamma_c + \frac{\partial s}{\partial t}$$

where the last term represents the influence of the object motion. By considering three successive images, two successive values of s denoted $s(t)$ and $s(t + \delta t)$ can be computed, and then an approximation $\widehat{\frac{ds}{dt}}$ of $\dot{s}$ can be expressed under the form:

$$\widehat{\frac{ds}{dt}} = \frac{s(t + \delta t) - s(t)}{\delta t} \tag{22}$$

Therefore, thanks to a measure of the acceleration $\widehat{\Gamma}_c$ and to an estimation $\widehat{L}$ of L, $\frac{\partial s}{\partial t} = f - \frac{\partial e}{\partial t}$ can be estimated by:

$$\widehat{\frac{\partial s}{\partial t}} = \widehat{\frac{ds}{dt}} + \widehat{L}\widehat{\Gamma} \tag{23}$$

In practice, to avoid problems due to measurement noises, this estimation may then be smoothed using a Kalman filter.

5.4 Case of a non full rank interaction matrix

Let us consider now the case when the matrix L is not of full rank. Using $e = C(s - s^*)$ and (17), we obtain:

$$\dot{e} = -CL\Gamma_c - Cf(V_c, r) - \frac{\partial e}{\partial t} \tag{24}$$

However, the acceleration vector allowing to obtain an exponential decay of the error can no more be determined using the previous method since matrix CL is not invertible. Let m be its rank ($m < 6$) and p the dimension of its null space ($p = 6 - m$). The idea of the control scheme is to express $\dot{e}$ as a function of p terms of speed and m terms of acceleration. We thus discretize p terms of acceleration under the form:

$$\Gamma_{c,i}(t) = \frac{V_{c,i}(t) - V_{c,i}(t - \delta t)}{\delta t}$$

where $\Gamma_{c,i}$ and $V_{c,i}$ are respectively the i-th component of the acceleration vector and of the kinematic screw of the camera. The choice of the acceleration terms to be discretized is not obvious if the null space of CL is not generated by p components of Γ. However, this choice is generally possible after an analysis of the considered task. The main idea is to obtain the simpler expression for the new interaction relation. Another one is to take into account intuitive constraints, such as controlling the orientation by an action on rotational axes, as we will see in Section 6.2.

To be as clear as possible, we consider that the discretization is made for the p first terms of Γ. There is no loss of generality, since it is always possible to come down to this case with a simple permutation of Γ components. Let us denote Σ the control vector $(V_{c,1}, \ldots, V_{c,p}, \Gamma_{c,p+1}, \cdots, \Gamma_{c,6})^T$. We can write:

$$CL\Gamma = CL\left(\frac{V_{c,1}(t)}{\delta t}, \ldots, \frac{V_{c,p}(t)}{\delta t}, \Gamma_{p+1}, \ldots, \Gamma_6\right)^T - CL\left(\frac{V_{c,1}(t-\delta t)}{\delta t}, \ldots, \frac{V_{c,p}(t-\delta t)}{\delta t}, 0, \ldots, 0\right)^T$$

Writing down:

$$CL = \begin{bmatrix} (CL)_1 & (CL)_3 \\ (CL)_2 & (CL)_4 \end{bmatrix}$$

where $(CL)_1$ is of dimension $p \times p$, $(CL)_2$ $m \times p$, $(CL)_3$ $p \times m$ and $(CL)_4$ $m \times m$, we have:

$$CL\Gamma = \begin{bmatrix} \frac{(CL)_1}{\delta t} & (CL)_3 \\ \frac{(CL)_2}{\delta t} & (CL)_4 \end{bmatrix} \Sigma - \underbrace{\begin{bmatrix} \frac{(CL)_1}{\delta t} & 0 \\ \frac{(CL)_2}{\delta t} & 0 \end{bmatrix} V_c(t-\delta t)}_{\text{denoted } l(V_c)} \qquad (25)$$

Similarly, we can express the p terms involved in (18) as :

$$Cf(V_c, r) = P\,(V_{c,1}(t), \ldots, V_{c,p}(t), 0, \ldots, 0)^T + Ch(V_{c,p+1}, \ldots, V_{c,6}, r) \qquad (26)$$

where h is the part of f which only depends on the m last terms of V_c and of r.

Let us denote $P = \begin{bmatrix} P_1 & P_3 \\ P_2 & P_4 \end{bmatrix}$ where P_i has the same size as $(CL)_i$. Then, from (25) and (26), we obtain:

$$\dot{e} = M\Sigma - Ch - Cl + \frac{\partial e}{\partial t} \qquad (27)$$

with:

$$M = \begin{bmatrix} \frac{(CL)_1}{\delta t} + P_1 & (CL)_3 \\ \frac{(CL)_2}{\delta t} + P_2 & (CL)_4 \end{bmatrix}$$

Control law

We admit that there exists a set of p indices such that the discretization of the p corresponding components of Γ leads to an invertible matrix M. In the opposite case, the system will be considered as uncontrollable.

From the relation (27), when M is invertible, it is possible to establish a gradient-like control law:

$$\Sigma = -\widehat{M}^{-1}\left(\lambda\widehat{e} - C\left(\widehat{h} + \widehat{l} - \widehat{\frac{\partial e}{\partial t}}\right)\right) \tag{28}$$

where $\widehat{M}$ is an approximation of M and $\widehat{l}$ is an estimation of l. Finally, the estimation of $\left(h + l - \frac{\partial e}{\partial t}\right)$ can be made using a similar technique as the one presented in the case of a full-rank matrix CL.

The limit case of non-full rank matrix is to have a null matrix of interaction. In such a case, the control is performed only in speed. We will see an example of this limit case in Section 6.1.

6 Applications

Two applications are presented in this section. The first one, whose aim is to position a camera parallel to a plane, corresponds to the case of a null matrix of interaction. The second one deals with the follow-up of a specified trajectory. It corresponds to the case of a non-null matrix of interaction but whose rank is not full. The case of a full rank matrix of interaction is treated in [7] with the tracking of a moving object.

6.1 Positioning the camera parallel to a plane

The aim of this task is to position a camera parallel to a planar surface (see Figure 6). The control law associated to this task corresponds to the case of a null interaction matrix.

6.1.1 Control law

From the equation of the plane expressed in the camera frame ($Z = Z_p + \gamma_1 X + \gamma_2 Y$), the parallelism condition can be expressed as:

$$\left(\left(\frac{\partial Z}{\partial X}\right), \left(\frac{\partial Z}{\partial Y}\right)\right)^T = \begin{pmatrix} \gamma_1 \\ \gamma_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{29}$$

We can notice in (1) that the orientation terms γ_1 and γ_2 appear both in the affine and quadratic terms (resp. a_i and q_i). A first idea is to consider the case when v_x and v_y are not simultaneously equal to zero.

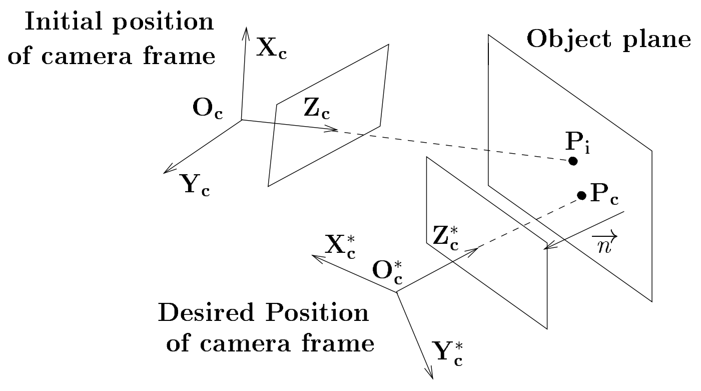


Figure 6: Context and expected result of the task

The condition (29) is then equivalent to (see (1)):

$$\begin{bmatrix} v_x & -v_y \\ v_y & v_x \end{bmatrix} \begin{pmatrix} \gamma_1 \\ \gamma_2 \end{pmatrix} = M \begin{pmatrix} \gamma_1 \\ \gamma_2 \end{pmatrix} = \begin{pmatrix} a_1 - a_4 \\ a_2 + a_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{30}$$

since in such a case, matrix M is always invertible. Using the derivative of these affine parameters, it is then possible to compute the link between their variations and the camera motion. This modelization has been used in [23]. Nevertheless, this approach, even if it is theoretically correct, is not applicable in practice. Indeed, it necessitates a non-null translational speed orthogonal to the optical axis. Such a motion does not seem to be relevant when the convergence is reached since it may imply the loss of the object from the field of view.

The terms γ_1 and γ_2 also appear in the quadratic parameters, where they are factor of the translational speed along the optical axis T_z. If value $v_z = T_z/Z$ is non-null, condition (29) is equivalent to:

$$\begin{pmatrix} s_1 \\ s_2 \end{pmatrix} = \begin{pmatrix} q_1 + \Omega_y \\ q_2 + \Omega_x \end{pmatrix} = - \begin{pmatrix} \gamma_1 \\ \gamma_2 \end{pmatrix} v_z = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

That choice seems more relevant than the previous one, even if a potential problem of collision appears. It can be solved by a control of translation T_z in order to maintain a constant time-to-contact [6]. We thus select $s = (s_1, s_2)^T$ whose desired value is $s^* = (0,0)^T$. The error e is chosen equal to s, meaning $C = \mathbb{I}_2$.

From (13), we obtain the following interaction relation related to s:

$$\dot{s} = v_z \begin{pmatrix} -\Omega_y + \gamma_1(\gamma_1 v_x + \gamma_2 v_y - v_z) - \gamma_2 \Omega_z + \gamma_1 \dfrac{\dot{T}_z}{Z_p} \\ \Omega_x + \gamma_2(\gamma_1 v_x + \gamma_2 v_y - v_z) + \gamma_1 \Omega_z + \gamma_2 \dfrac{\dot{T}_z}{Z_p} \end{pmatrix} \tag{31}$$

As T_z is not directly used to control the orientation, it can be fixed. We fix it to a constant non-null value. This implies that the acceleration $\dot{T}_z$ is zero. Therefore, the interaction matrix L is null since $\dot{s}$ does not depend on the angular accelerations. The control vector is thus $(\Omega_x, \Omega_y)^T$, and not $(\dot{\Omega}_x, \dot{\Omega}_y)^T$. Finally, equation (31) can be written:

$$\dot{s} = M \begin{pmatrix} \Omega_x \\ \Omega_y \end{pmatrix} + h \quad \text{with} \quad M = \begin{bmatrix} 0 & -v_z \\ v_z & 0 \end{bmatrix} \tag{32}$$

Let us denote $h = \delta_m + \delta_s$ with:

$$\delta_m = \begin{pmatrix} -\gamma_2 v_z \Omega_z \\ \gamma_1 v_z \Omega_z \end{pmatrix} = \Omega_z \begin{pmatrix} s_2 \\ -s_1 \end{pmatrix} \quad \text{and} \quad \delta_s = (v_z - \gamma_1 v_x - \gamma_2 v_y) s = \alpha s$$

As explained in the previous section, it is necessary to separate the speed linked to the controlled axes from the one linked to uncontrolled axes, which will be considered as measured data from the system. In this task, only the rotational speeds Ω_x and Ω_y are constrained by the control law. They only appear multiplied by the translation T_z previously fixed. Therefore, it is possible to write:

$$\begin{pmatrix} \Omega_x \\ \Omega_y \end{pmatrix} = -\widehat{M}^{-1} \left(\lambda s + \delta_s + \delta_m\right)$$

where the estimation $\widehat{M}$ of M is given by:

$$\widehat{M} = \begin{bmatrix} 0 & -\widehat{v_z} \\ \widehat{v_z} & 0 \end{bmatrix}$$

in which it is possible to approximate v_z by $\widehat{v_z} = \frac{a_1+a_4}{2}$ (see (1). Similarly, δ_s can be approximated by $\widehat{\delta_s} = \widehat{v_z}\widehat{s}$ Finally, the control law is given by:

$$\begin{pmatrix} \Omega_x \\ \Omega_y \end{pmatrix} = \frac{1}{\widehat{v_z}} \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \left((\lambda + \widehat{v_z}) \begin{pmatrix} \widehat{s_1} \\ \widehat{s_2} \end{pmatrix} + \widehat{\Omega_z} \begin{pmatrix} \widehat{s_2} \\ -\widehat{s_1} \end{pmatrix} \right) \tag{33}$$

Convergence analysis

The behavior of the closed loop system is given by:

$$\dot{s} = -M\widehat{M}^{-1} \left((\lambda + \widehat{v_z})\, \widehat{s} + \widehat{\Omega_z} \begin{pmatrix} \widehat{s_2} \\ -\widehat{s_1} \end{pmatrix} \right) + \alpha\, s + \Omega_z \begin{pmatrix} s_2 \\ -s_1 \end{pmatrix}$$

The positivity of matrix $K = M\widehat{M}^{-1}$ allows to ensure the convergence. In our case, we have $K = \frac{v_z}{\widehat{v_z}} \mathbb{I}_2$. The convergence will thus be ensured as soon as v_z is estimated with a correct sign.

6.1.2 Results

The control law presented previously has been implemented on a 6 d.o.f. robot. The images of the scene at the initial and final positions are respectively presented on Figure 7.a and Figure 7.b. One can notice that the planar object does not cover the totality of the field of view at the beginning. Moreover, non-planar objects are displayed on the plane. Nevertheless, due to the robustness of the RMR algorithm, we will see that the task is correctly performed.

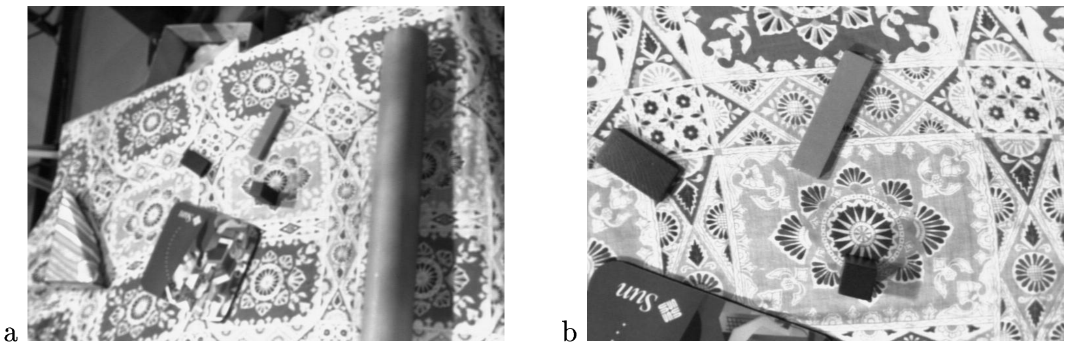
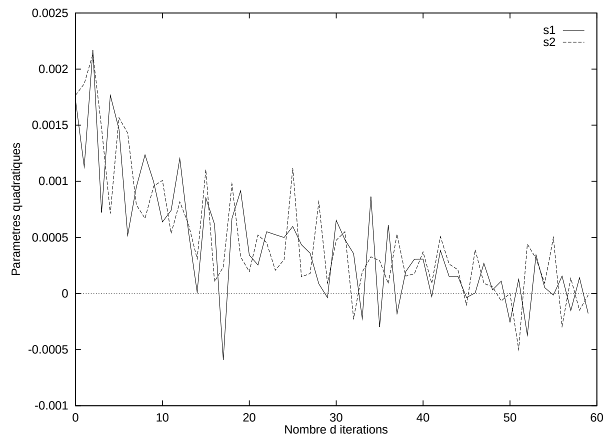


Figure 7: Observed scene at initial position (a), and at final one (b)

The alignment task has been tested using the following values: $\lambda = 0.1$ and $T_z = 10$ mm/s. Initial angular errors were about 12 degrees around $\overrightarrow{x}$ and 15 degrees around $\overrightarrow{y}$. Finally, non-constrained speeds, meaning T_x, T_y and Ω_z, have been set to zero. The obtained results are depicted on Figure 8. More precisely, the visual features s are given on Figure 8.a, the rotational speeds computed by the control law on Figure 8.b, and finally, the angular errors on Figure 9. Even if the estimation of the quadratic parameters is noisy, the convergence is obtained. It is also proved by the decrease of the angular errors. These errors are computed from an a priori knowledge of the plane orientation, and a measurement, by odometry, of the angular position of the camera. These features are obviously not used in the control loop.

In order to avoid the loss of the object from the field of view, a fixation task can be added to the positioning one. Its aim is to maintain the center of the initial image at the image center (see Figure 6). As explained in [6], it can be performed by maintaining its speed equal to zero, using T_x and T_y, since the pan and tilt are already constrained. This leads to a 4×4 matrix of interaction whose rank is equal to 2.

6.2 Follow-up of a trajectory to observe a surface

For this task, we suppose that the camera observes a 3D surface. The objective is to follow a trajectory parallel to this surface. It is controlled in such a way that the acquired sequence respects constraints given, a 

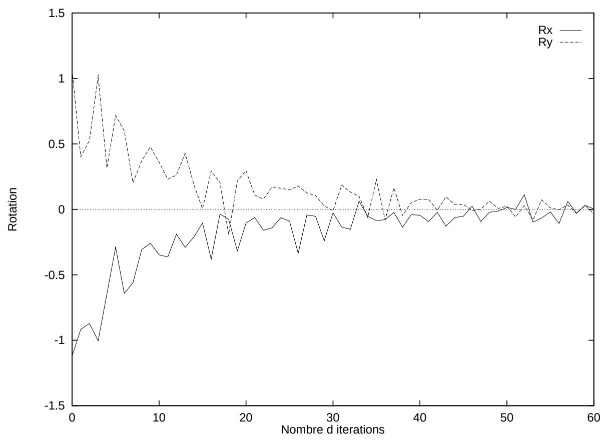
b 

Figure 8: Task function (a). Angular speeds given by the control law (in deg/s) (b)

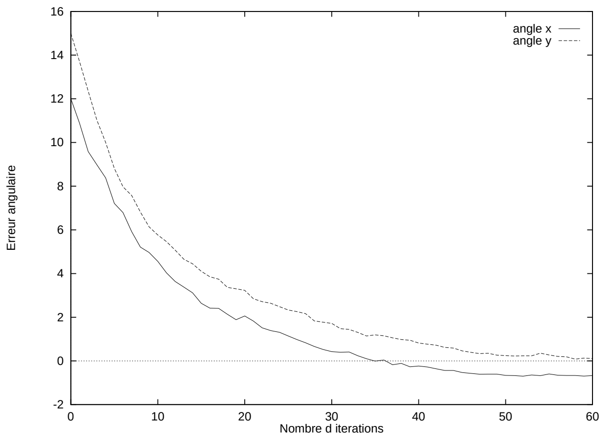


Figure 9: Angular errors (in deg)

for example, by a downstream image processing such as mosaicing.

As it will be shown now, this task corresponds to the intermediate case where the interaction matrix between the measurement vector and the acceleration is generally of full rank, but is not anymore of full rank in the neighborhood of convergence.

6.2.1 Control law

To perform this task, we choose as task vector s:

$$s = (hyp_1, hyp_2, div, c_1, c_2, rot)^T$$

where:

$$\begin{cases} hyp_1 &= a_1 - a_4 \qquad\qquad hyp_2 &= a_2 + a_3 \\ div &= a_1 + a_4 \qquad\qquad rot &= a_3 - a_2 \end{cases}$$

The desired value of s is given by the vector s^*:

$$s^* = (0, 0, 0, c_1^*, c_2^*, rot^*)^T$$

where c_1^*, c_2^* and rot^* are fixed in order to define a 2D trajectory in the image. Then, from equation (13), the interaction relation related to s is given by:

$$\dot{s} = L\ (\dot{T}_x, \dot{T}_y, \dot{T}_z, \dot{\Omega}_x, \dot{\Omega}_y, \dot{\Omega}_z)^T + f(V, r)$$

with:

$$L = \begin{bmatrix} \frac{\gamma_1}{Z_p} & -\frac{\gamma_2}{Z_p} & 0 & 0 & 0 & 0 \\ \frac{\gamma_2}{Z_p} & \frac{\gamma_1}{Z_p} & 0 & 0 & 0 & 0 \\ \frac{\gamma_1}{Z_p} & \frac{\gamma_2}{Z_p} & \frac{2}{Z_p} & 0 & 0 & 0 \\ -\frac{1}{Z_p} & 0 & 0 & 0 & -1 & 0 \\ 0 & -\frac{1}{Z_p} & 0 & 1 & 0 & 0 \\ -\frac{\gamma_2}{Z_p} & \frac{\gamma_1}{Z_p} & 0 & 0 & 0 & 2 \end{bmatrix}$$

and:

$$f(V,r) = \begin{pmatrix} (v_x\Omega_y + v_y\Omega_x - \gamma_1^2 v_x^2 + \gamma_2^2 v_y^2 + (\gamma_2 v_x + \gamma_1 v_y)\Omega_z + (\gamma_1 v_x - \gamma_2 v_y)v_z) \\ (-v_x\Omega_x + v_y\Omega_y + \gamma_1\gamma_2(v_x^2 - v_y^2) - (\gamma_1^2 + \gamma_2^2)v_x v_y + \dots \\ \dots + (\gamma_2 v_y - \gamma_1 v_x)\Omega_z + (\gamma_2 v_x + \gamma_1 v_y)v_z) \\ (v_x\Omega_y + v_y\Omega_x - \gamma_1^2 v_x^2 + \gamma_2^2 v_y^2 + (\gamma_2 v_x + \gamma_1 v_y)\Omega_z + (\gamma_1 v_x - \gamma_2 v_y)v_z) \\ \gamma_1 v_x^2 + \gamma_1 v_x\Omega_y + \gamma_2 v_x v_y - \gamma_2 v_x\Omega_x - v_x v_z \\ \gamma_1 v_x v_y + \gamma_1 v_y\Omega_y + \gamma_2 v_y^2 - \gamma_2 v_y\Omega_x - v_y v_z \\ (v_y\Omega_y + v_x\Omega_x + (\gamma_2^2 - \gamma_1^2)v_x v_y - \gamma_1\gamma_2(v_x^2 + v_y^2) + \dots \\ \dots + (\gamma_1 v_x + \gamma_2 v_y)\Omega_z + (\gamma_1 v_y - \gamma_2 v_x)v_z \end{pmatrix} \tag{34}$$

In the neighborhood of the convergence, γ_1 and γ_2 are close to zero. In that case, the interaction matrix L is no more of full rank 6, but only of rank 4. Therefore, as explained in Section 5.4, two components of the acceleration vector have to be discretized.

Another solution would be to consider the 8 parameters of the motion model (1). It would lead to a 8×6 matrix whose rank would always be 6. Nevertheless, the computation load would be increased in that case. Moreover, as the components of the null space of L can be controlled using speed, this alternative solution has not been considered.

At convergence, the two first components of $\dot{s}$ do not depend anymore on the acceleration. These two first terms are the derivative of the hyperbolic parameters hyp_1 and hyp_2. They are closely linked to the orientation of the camera. Therefore, we choose to discretize the angular accelerations around the $\overrightarrow{x}$ and $\overrightarrow{y}$ axes in order to obtain a link between the derivatives of the hyperbolic terms and Ω_x and Ω_y. This choice was preferred to the discretization of the translational accelerations $\dot{T_x}$ and $\dot{T_y}$ because it seems natural to control the orientation by rotational motions.

Using (34) and under the approximation $\gamma_1 = \gamma_2 = 0$, the interaction relation associated to s can be written:

$$\dot{s} = M\ (\dot{T_x}, \dot{T_y}, \dot{T_z}, \Omega_x, \Omega_y, \dot{\Omega}_z)^T + \delta_s \tag{35}$$

with:

$$M = \begin{bmatrix} 0 & 0 & 0 & v_y & v_x & 0 \\ 0 & 0 & 0 & -v_x & v_y & 0 \\ 0 & 0 & \frac{2}{Z_p} & -v_y & v_x & 0 \\ -\frac{1}{Z_p} & 0 & 0 & 0 & -\frac{1}{\delta t} & 0 \\ 0 & -\frac{1}{Z_p} & 0 & \frac{1}{\delta t} & 0 & 0 \\ 0 & 0 & 0 & v_x & v_y & 2 \end{bmatrix} \quad \text{and} \quad \delta_s = \left(0, 0, \frac{1}{\delta t}\widehat{\Omega_y}, -\frac{1}{\delta t}\widehat{\Omega_x}, 0\right)^T \tag{36}$$

where $\widehat{\Omega_x}$ and $\widehat{\Omega_y}$ are a measure of Ω_x and Ω_y obtained at the previous iteration. Using the developments presented in Section 5.4, we obtain the following control law:

$$(\dot{T_x}, \dot{T_y}, \dot{T_z}, \Omega_x, \Omega_y, \dot{\Omega}_z)^T = -\widehat{M}^{-1}(\lambda(\widehat{s} - s^*) + \widehat{\delta}_s) \tag{37}$$

where:

- $\widehat{M}$ is the approximation of M given in (36) in which:
  - the terms v_x and v_y are estimated from the computed value of the constant parameters of motion and a measure of the camera rotational velocity (see (1)):

$$\widehat{v_x} = -c_1 - \widehat{\Omega_y} \quad \text{and} \quad \widehat{v_y} = -c_2 + \widehat{\Omega_y}$$

– the depth Z_p is computed from previous estimations of v_x and v_y and a measure of the translational velocities T_x and T_y:

$$Z_p = \frac{\widehat{v_x} T_x + \widehat{v_y} T_y}{\widehat{v_x}^2 + \widehat{v_y}^2}$$

- $\widehat{\delta_s}$ is an estimation of δ_s obtained directly from $\widehat{\Omega_x}$ and $\widehat{\Omega_y}$.

Let us finally note that to avoid that matrix M is singular at the beginning of the servoing, it must start with a non-null translation orthogonal to the optical axis.

6.2.2 Results

The control law (37) has been validated on our robotics platform. The observed scene, called "newspaper", is presented on Figure 10.

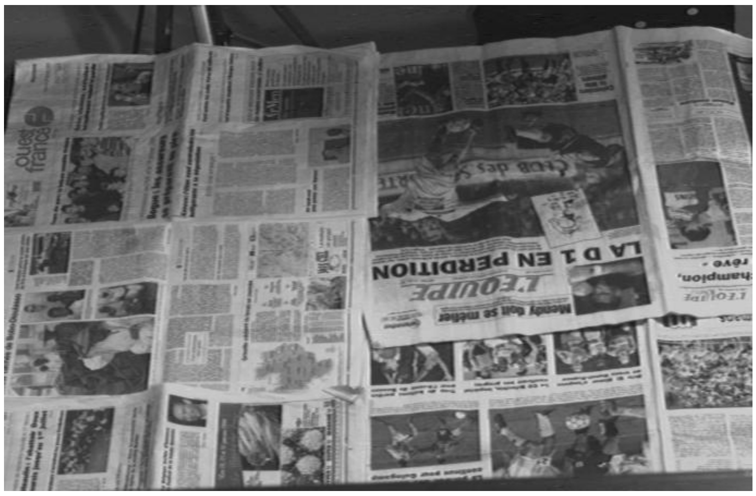


Figure 10: "Newspaper" scene

The reference plane is horizontal in the fixed frame, with a precision of one or two degrees. Initial angular errors were 15 degrees around $\overrightarrow{x}$ axis and 12 degrees around $\overrightarrow{y}$ one. Initial translational speeds were $T_x = 10$ mm/s and $T_y = 5$ mm/s. The images treated by the motion estimation algorithm are 128×128 pixels, which leads to 300 ms iterations. Which such a processing rate, gain λ is set to 0.2. Finally, successive values of s^* were $(0, 0, 0, -0.02, 0, 0)^T$ (from iteration 1 to iteration 140), $(0, 0, 0, 0, 0.015, 0)^T$ (it. 141 to 250), $(0, 0, 0, 0, 0.015, -0.157)^T$ (it. 251 to 320) and $(0, 0, 0, 0, 0.015, 0)^T$ (it. 321 to 500). It corresponds to a translation along $\overrightarrow{x}$ axis (till it. 140), followed by a translation along $\overrightarrow{y}$ axis (from it. 141 to 320) during which appears a rotation around the optical axis (from it. 251 to 320).

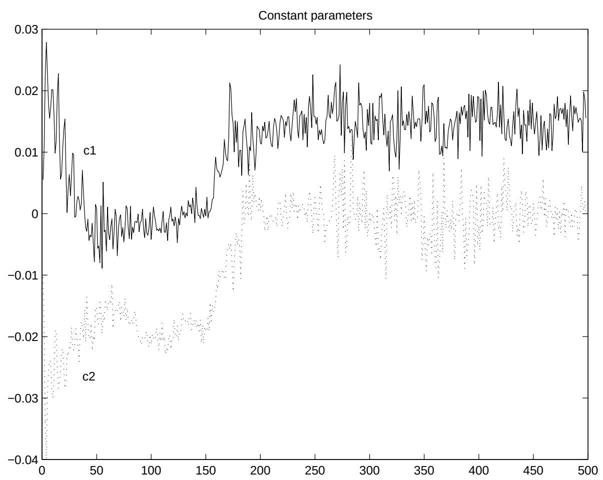
Eight curves are displayed about this experiment. As before, it is first curves concerning the visual features, successively constants parameters of the motion model on Figure 11.a, hyperbolic parameters on Figure 11.b, the rotational parameter on Figure 12.a and then the divergence on Figure 12.b. Then, are displayed the different values computed by the control law, i.e. translational accelerations orthogonal to the optical axis on Figure 13.a, translational accelerations along the optical axis on Figure 13.b, rotational speeds around the same first axes on Figure 14.a, and then, the angular acceleration around the optical axis on Figure 14.b. We can see that each of the visual features converges towards its desired values despite the important noise obtained. This noise is due to the fact that a part of the control is done in acceleration while the rate is not very high (just a few more than 3 Hz). On the contrary, the rotational parameter is highly less noisy because of its relative independence with regard to the other components of s.

Two other curves are also displayed on Figure 15 to show the system behaviour during the servoing. It is the 3D trajectory of the camera optical center, shown on Figure 15.a, and its projection on the reference plane on Figure 15.b. The initial position is marked with a circle, and the final one with a triangle. We can note that the translation along the optical axis is only involved at the beginning of the servoing till the image plane is parallel to the observed surface. Furthermore, the amplitude of this motion is not large (less than 10 cm), and after the orientation has converged, the camera trajectory remains planar. We can also note that the the noise that appeared in the computed control law has low effects on the 3D trajectory. Finally, we can see on the 2D trajectory of the optical center that the realized behaviour is close to the desired one.

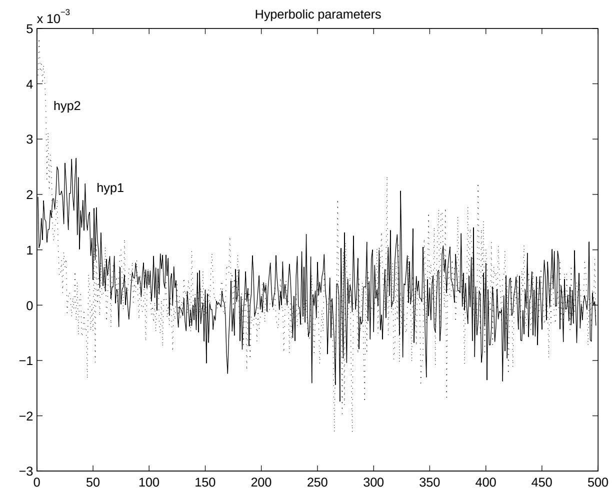
We finally present on Figure 16 a sequence of images acquired during the realization of the task. More precisely, this sequence is made with one image upon 20 from the set of all acquired images. The first translation, along the x direction in the image is from image 1 to 10, then the translation in the other direction is from image 10 to the end (image 25), while the rotation step around the optical axis is visible from image 16 to 21. If we refer to the global view of the scene, we can note that the trajectory showed by the sequence corresponds to the desired one.

7 Conclusion

Until now, motion in the image has only been used for particular robotics tasks. In this paper, we have proposed a general way to introduce such features in a robotic control scheme. Two approaches have been developed. The first one is based on the idea that position in the image can be retrieved by a simple integration of 2D motion. These reconstructed values can then be used in a classical 2D visual servoing

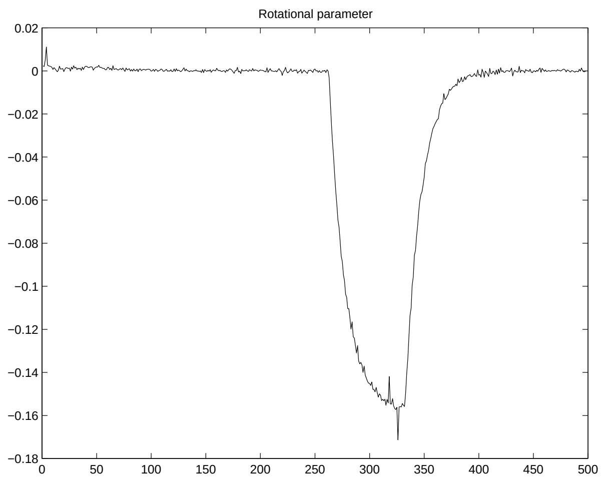


a

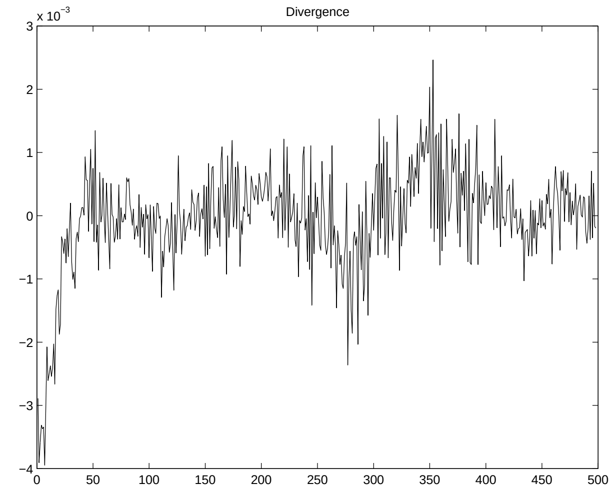


b

Figure 11: Task function: constant parameters (a), hyperbolic parameters(b)

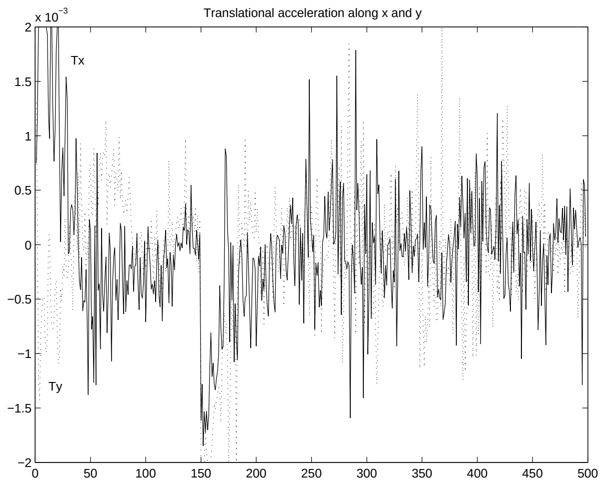


a

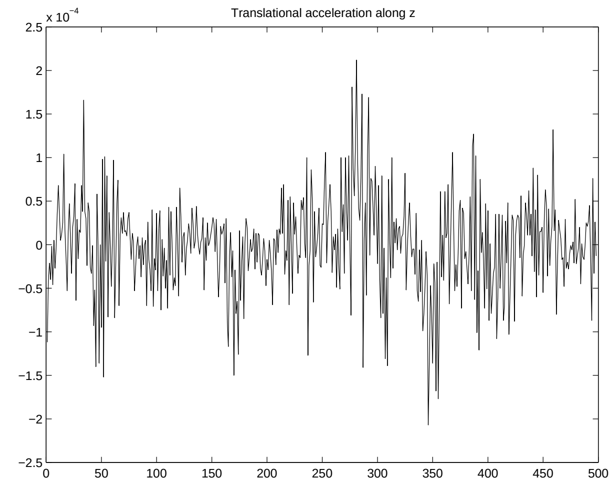


b

Figure 12: Task function: rotational parameter(a), divergence (b)

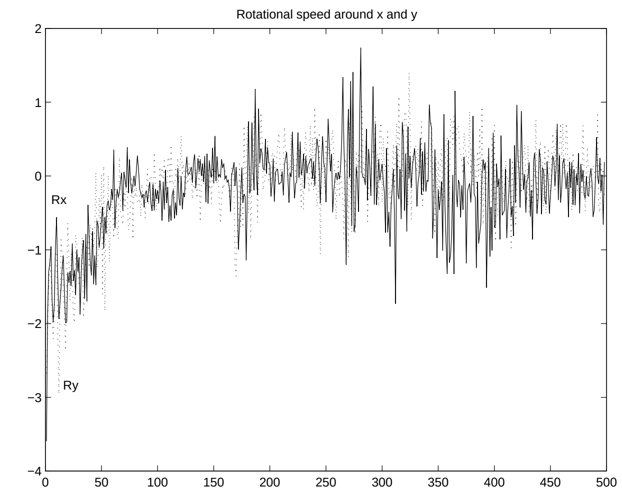


a



b

Figure 13: Control in translational acceleration: along x and y axes (a), along z axis (b)

a 

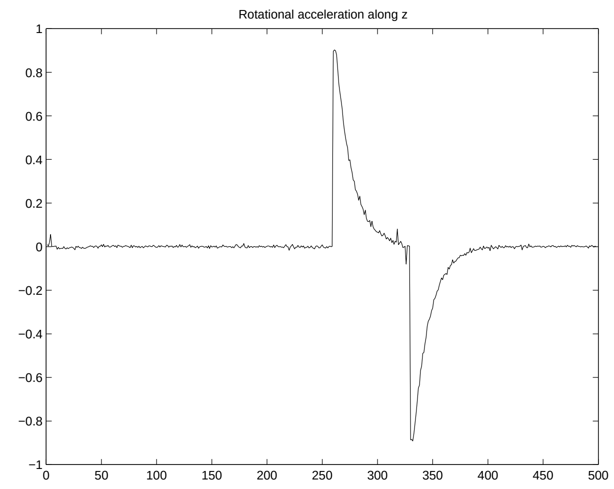
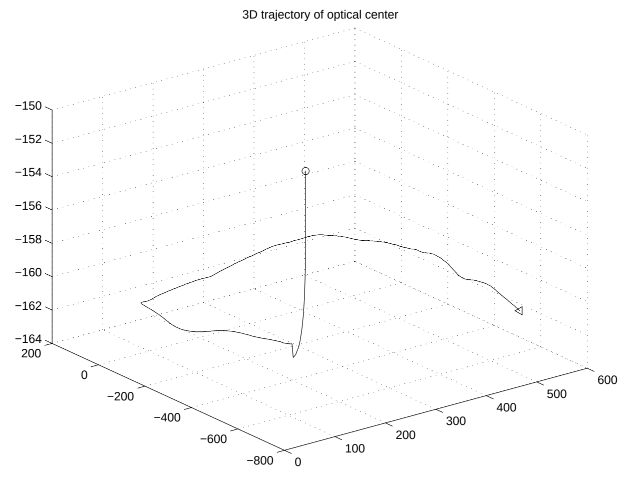
b 

Figure 14: Control in angular speed around axes x and y (a), acceleration around axis z (b)

a 

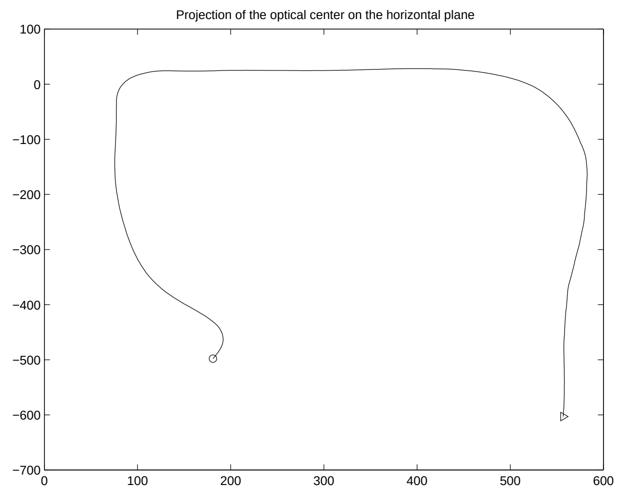
b 

Figure 15: 3D position of the optical center (a). Its projection on the reference plane (b)

scheme. The second method relies on an entirely new control scheme. From the visual servoing method developed in [12], we have followed the same way of reasoning by determining an analytical relation between the derivative of the used features and the camera motion. In the geometric case, this link was linear with respect to the speed. Using motion parameters makes appear at the same time the acceleration and the speed of the camera. Two main cases were distinguished whether the linear link with the acceleration is of full rank or not.

Several robotic tasks have been considered. Using the first approach, we have designed a task whose aim is to position a camera with respect to an unknown and complex scene. Points of interest are manually chosen and matched at the beginning. Then, their position is estimated using the motion in the image.

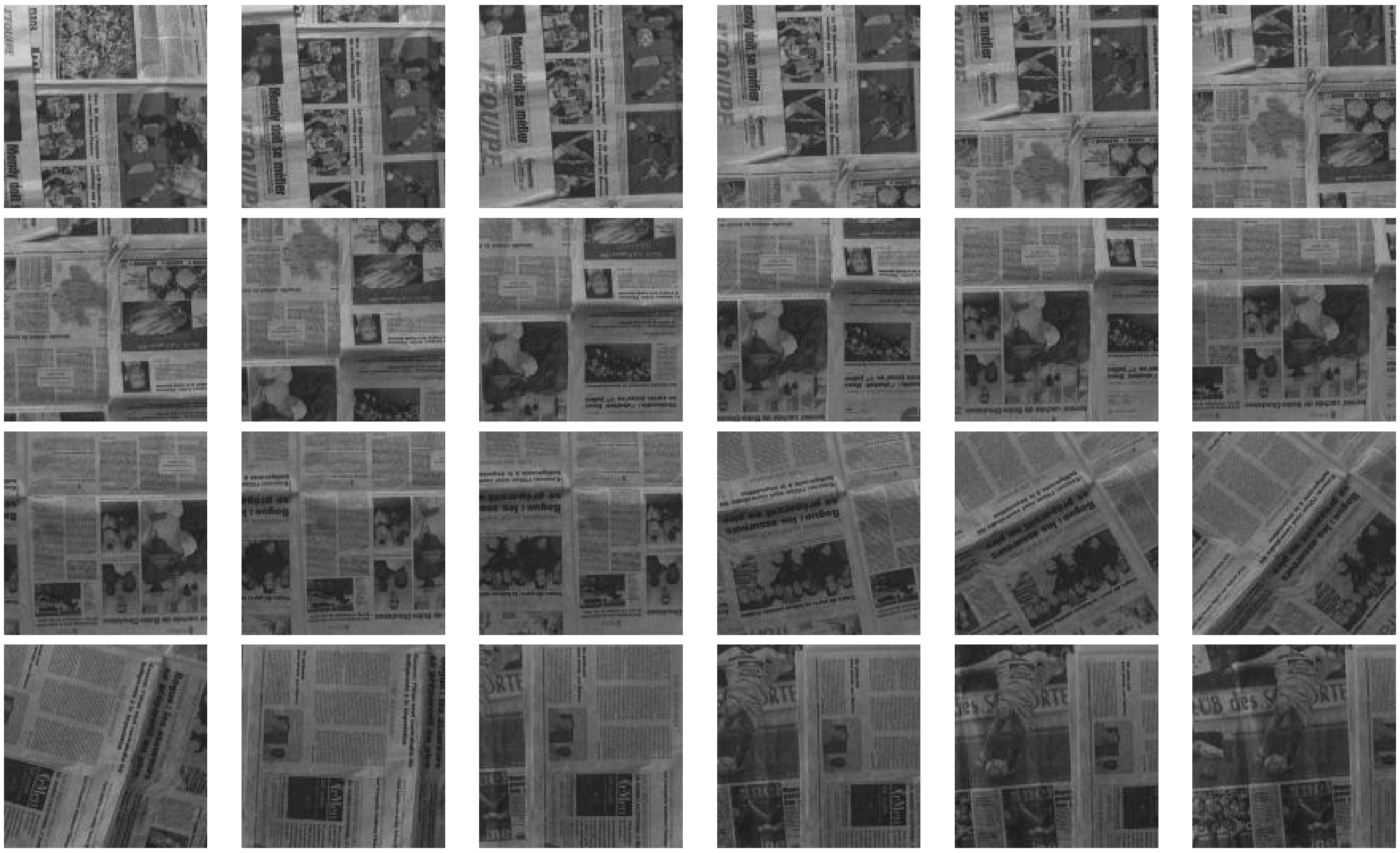


Figure 16: One image over 20 of the "newspaper" sequence (approximately one image every 6 seconds)

It is thus robust to occlusion. An important thing to underline is the necessity to estimate 8 parameters of motion, even if only 6 d.o.f. are constrained. This is due to the insufficiency of the 2D-affine model to express all the links between 3D motions and image motion. In the applications related to the second approach, the relation between the derivative of the chosen visual features and the camera acceleration were once not of full rank, and the other time of null rank. Nevertheless, it was shown that despite the noise in the acquisition of such features and despite the quite slow rate for a control in acceleration, the objective was globally reached. Moreover, we can underline the fact that for these two tasks, no image-based servoing were potentially usable to achieve the desired goals.

Concerning the first approach, improvement could be done, first, on the initial image processing for points matching. A complete automatic and accurate way could be found. Moreover, even if the algorithm can also work on non-planar scenes (50% outliers are allowed), we should study the case of less planar scenes. Concerning the second approach, on the practical level, future works will concern the reduction of computational costs to increase the robustness of the control loop. A theoretical proof of the stability should also be studied, especially when the interaction matrix is not of full rank. It would define the stability bounds. This analysis should also take into account the delay introduced by the linearization.

Appendix: Q_{p_i} matrices computation

The derivative of terms p_i can be computed from equation (1). This leads to:

$$\dot{c}_1 = -\frac{\dot{Z}_p T_x}{Z_p^2} + \frac{\dot{T}_x}{Z_p} + \dot{\Omega}_y \qquad \dot{c}_2 = -\frac{\dot{Z}_p T_y}{Z_p^2} + \frac{\dot{T}_y}{Z_p} - \dot{\Omega}_x \tag{38}$$

$$\left\{ \begin{array}{lcl} \dot{a}_1 & = & \frac{\dot{Z}_p}{Z_p^2}(T_z + \gamma_1 T_x) - \frac{1}{Z_p}(\dot{T}_z + \gamma_1 \dot{T}_x + \dot{\gamma}_1 T_x) \\ \dot{a}_2 & = & \frac{\gamma_2 \dot{Z}_p T_x}{Z_p^2} - \frac{1}{Z_p}(\gamma_2 \dot{T}_x + \dot{\gamma}_2) - \dot{\Omega}_z \\ \dot{a}_3 & = & \frac{\gamma_1 \dot{Z}_p T_y}{Z_p^2} - \frac{1}{Z_p}(\gamma_1 \dot{T}_y + \dot{\gamma}_1 T_y) + \dot{\Omega}_z \\ \dot{a}_4 & = & \frac{\dot{Z}_p}{Z_p^2}(T_z + \gamma_2 T_y) - \frac{1}{Z_p}(\dot{T}_z + \gamma_2 \dot{T}_y + \dot{\gamma}_2 T_y) \end{array} \right. \tag{39}$$

$$\dot{q}_1 = \frac{\gamma_1 \dot{Z}_p T_z}{Z_p^2} - \frac{\dot{\gamma}_1 T_z - \gamma_1 \dot{T}_z}{Z_p} - \dot{\Omega}_y \qquad \dot{q}_2 = \frac{\gamma_2 \dot{Z}_p T_z}{Z_p^2} - \frac{\dot{\gamma}_2 T_z - \gamma_1 \dot{T}_z}{Z_p} + \dot{\Omega}_x \tag{40}$$

Now, if we consider the equation $Z = Z_p + \gamma_1 X + \gamma_2 Y$ expressing the planar approximation of the considered object, we have for all points of this plane:

$$\dot{Z} = \dot{Z}_p + \gamma_1 \dot{X} + \gamma_2 \dot{Y} + \dot{\gamma}_1 X + \dot{\gamma}_2 Y$$

Using the value of $\dot{X}$, $\dot{Y}$ and $\dot{Z}$, expressing the kinematic relation between the camera and the object, given for example in [26], the previous equation leads to:

$$\begin{array}{l} T_z - \dot{Z}_p - \gamma_1(T_x + \Omega_y Z_p) - \gamma_2(T_y - \Omega_x Z_p) \\ -X(\Omega_y + \gamma_1^2 \Omega_y + \gamma_2 \Omega_z - \gamma_1 \gamma_2 \Omega_x + \dot{\gamma}_1) \\ +Y(\Omega_x + \gamma_2^2 \Omega_x + \gamma_1 \Omega_z - \gamma_1 \gamma_2 \Omega_y - \dot{\gamma}_2) = 0 \end{array} \tag{41}$$

The relation (41) is affine with respect to X and Y, and valid for all points of the plane. A necessary condition in order that the left term is null is therefore that each of the affine coefficients is also null, meaning:

$$\begin{array}{lcl} \dot{Z}_p & = & T_z - \gamma_1(T_x + \Omega_y Z_p) - \gamma_2(T_y - \Omega_x Z_p) \\ \dot{\gamma}_1 & = & -\Omega_y - \gamma_1^2 \Omega_y - \gamma_2 \Omega_z + \gamma_1 \gamma_2 \Omega_x \\ \dot{\gamma}_2 & = & \Omega_x + \gamma_2^2 \Omega_x + \gamma_1 \Omega_z - \gamma_1 \gamma_2 \Omega_y \end{array} \tag{42}$$

Then, by replacing $\dot{Z}_p$, $\dot{\gamma}_1$ and $\dot{\gamma}_2$ by their respective value in each $\dot{p}_i$ given in (38), (39), and (40), we obtain:

$$\left\{\begin{array}{rcl} \dot{c}_1 & = & \gamma_1 v_x^2 + \gamma_1 v_x \Omega_y + \gamma_2 v_x v_y - \gamma_2 v_x \Omega_x - v_x v_z + \frac{\dot{T}_x}{Z_p} + \dot{\Omega}_y \\ \dot{c}_2 & = & \gamma_1 v_x v_y + \gamma_1 v_y \Omega_y + \gamma_2 v_y^2 - \gamma_2 v_y \Omega_x - v_y v_z + \frac{\dot{T}_y}{Z_p} - \dot{\Omega}_x \end{array}\right.$$

$$\left\{\begin{array}{rcl} \dot{a}_1 & = & -\gamma_1^2 v_x^2 - \gamma_1\gamma_2 v_x v_y + \gamma_2 v_x \Omega_z + v_x \Omega_y - \gamma_1 v_z \Omega_y - \gamma_2 v_z v_y \\ & & +\gamma_2 v_z \Omega_x + v_z^2 - \frac{\gamma_1 \dot{T}_x}{Z_p} - \frac{\dot{T}_z}{Z_p} \\ \dot{a}_2 & = & \gamma_1\gamma_2 v_x^2 - \gamma_2^2 v_x v_y + \gamma_2 v_x v_z - \gamma_1 v_x \Omega_z - v_x \Omega_x - \frac{\gamma_2 \dot{T}_x}{Z_p} - \dot{\Omega}_z \\ \dot{a}_3 & = & -\gamma_1^2 v_x v_y - \gamma_1\gamma_2 v_y^2 + \gamma_1 v_y v_z + \gamma_2 v_y \Omega_z + v_y \Omega_y - \frac{\gamma_1 \dot{T}_y}{Z_p} + \dot{\Omega}_z \\ \dot{a}_4 & = & -\gamma_1\gamma_2 v_x v_y - \gamma_2^2 v_y^2 - \gamma_1 v_y \Omega_z - v_y \Omega_x - \gamma_1 v_x v_z - \gamma_1 v_z \Omega_y \\ & & +\gamma_2 v_z \Omega_x + v_z^2 - \frac{\gamma_2 \dot{T}_y}{Z_p} - \frac{\dot{T}_z}{Z_p} \end{array}\right.$$

$$\left\{\begin{array}{rcl} \dot{q}_1 & = & \gamma_1^2 v_x v_z + \gamma_1\gamma_2 v_y v_z - \gamma_1 v_z^2 - \gamma_2 v_z \Omega_z - v_z \Omega_y + \frac{\gamma_1 \dot{T}_z}{Z_p} + \dot{\Omega}_y \\ \dot{q}_2 & = & \gamma_1\gamma_2 v_x v_z + \gamma_2^2 v_y v_z - \gamma_2 v_z^2 + \gamma_1 v_z \Omega_z + v_z \Omega_x + \frac{\gamma_2 \dot{T}_z}{Z_p} - \dot{\Omega}_x \end{array}\right.$$

In each of these expressions, we can retrieve the L_{p_i} and Q_{p_i} matrices given in equations (11) and (14).

References

[1] P.K. Allen, A. Timcenko, B. Yoshimi, and 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, Apr. 1993.

[2] R. Cipolla and A. Blake. Image divergence and deformation from closed curves. *International Journal of Robotics Research*, 16(1):77–96, Jan. 1997.

[3] C. Colombo and B. Allotta. Image-based robot task planning and control using a compact visual representation. *IEEE Trans. on Systems, Man, and Cybernetics*, 29(1):92–99, Jan. 1999.

[4] D. Coombs and K. Roberts. Centering behavior using peripheral vision. In *Int. Conf. on Computer Vision and Pattern Recognition*, pages 440–445, New York City, NY, June 1993.

[5] E. Coste-Manière, P. Couvignon, and P.K. Khosla. Visual servoing in the task-function framework: a contour following task. *Journal of Intelligent Robotic Systems*, 12(1):1–22, Jan. 1995.

[6] A. Crétual. *Asservissement visuel à partir d'informations de mouvement dans l'image.* PhD thesis, Université de Rennes I, Nov. 1998.

[7] A. Crétual and F. Chaumette. Application of motion based visual servoing to target tracking. Submitted to *International Journal of Robotics Research*, 2001.

[8] A. Crétual and F. Chaumette. Dynamic stabilization of a pan and tilt camera for sub-marine image visualization. *Computer Vision and Image Understanding*, 79(1):47–65, July 2000.

[9] A. Dev, B.J.A. Krose, L. Dorst, and F.C.A. Groen. Observer curve and object detection from the optic flow. In *SPIE Conference on Intelligent Robots and Computer Vision*, Dec. 1994.

[10] T. Drummond and R. Cipolla. Real-time tracking of complex structures with on-line camera calibration. In *British Machine Vision Conference*, pages 574–583, Sept. 1999.

[11] T. Drummond and R. Cipolla. Visual tracking and control using Lie algebra. In *CVPR*, volume 2, pages 652–657, Fort Collins (CO), June 1999.

[12] 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.

[13] J.T. Feddema and C.S.G. Lee. Adaptative image feature prediction and control for visual tracking with a hand-eye coordinated camera. *IEEE Trans. on Systems, Man, and Cybernetics*, 20(5):1172–1183, Sept. 1990.

[14] E. Grosso, G. Metta, A. Oderra, and G. Sandini. Robust visual servoing in 3-d reaching tasks. *IEEE Trans. on Robotics and Automation*, 12(5):732–742, Oct. 1996.

[15] G.D. Hager. A modular system for robust hand-eye coordination using feedback from stereo vision. *IEEE Trans. on Robotics and Automation*, 13(4):582–595, Aug. 1997.

[16] C.G. Harris and M. Stephens. A combined corner and edge detector. In *Alvey Conference*, pages 189–192, 1988.

[17] K. Hashimoto, editor. *Visual servoing. Real-time control of robot manipulators based on visual sensory feedback.* World scientific series in robotics and automated systems. World scientific, 1993.

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

[19] F. Keçeci, M. Tonko, H.H. Nagel, and V. Gengenbach. Improving visually servoed disasembly operations by automatic camera placement. In *IEEE Int. Conf. on Robotics and Automation*, pages 2947–2952, Leuven, Belgium, May 1998.

[20] H. Michel and P. Rives. Singularities in the determination of the situation of a robot effector from the perspective view of 3 points. Rapport de Recherche 1850, INRIA, Feb. 1993.

[21] S. Negahdaripour and S. Lee. Motion recovery from image sequences using only first order optical flow information. *International Journal of Computer Vision*, 9(3):163–184, Dec. 1992.

[22] J.M. Odobez and P. Bouthemy. Robust multiresolution estimation of parametric motion models. *Journal of Visual Communication and Image Representation*, 6(4):348–365, Dec. 1995.

[23] P. Questa, E. Grossmann, and G. Sandini. Camera self orientation and docking maneuver using normal flow. In *SPIE AeroSense'95*, Orlando, Florida, Apr. 1995.

[24] C. Samson, M. Le Borgne, and B. Espiau. *Robot control : the task function approach.* Oxford University Press, 1991.

[25] J. Santos-Victor, G. Sandini, F. Curotto, and S. Garibaldi. Divergent stereo in autonomous navigation: from bees to robots. *International Journal of Computer Vision*, 14(2):159–177, Mar. 1995.

[26] M. Subarrao. Interpretation of image flow : a spatio-temporal approach. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 11(3):266–278, Mar. 1989.

[27] M. Subarrao and A. Waxman. Closed-form solutions to image equations for planar surface in motion. *Computer Vision, Graphics, and Image Processings*, 36(2):208–228, Nov. 1986.

[28] V. Sundareswaran, P. Bouthemy, and F. Chaumette. Exploiting image motion for active vision in a visual servoing framework. *International Journal of Robotics Research*, 15(6):629–645, Dec. 1996.