# DEALING WITH VISUAL FEATURES LOSS DURING A VISION-BASED TASK FOR A MOBILE ROBOT

SHORTENED TITLE: TREATING IMAGE LOSS DURING A VISION-BASED TASK

**David Folio**[12] **and Viviane Cadenat**[12]

[1] *LAAS-CNRS; Université de Toulouse; 7, avenue du Colonel Roche, F-31077 Toulouse, France.*
[2] *Université de Toulouse ; UPS.*

***In this paper, we address the problem of computing the image features when they become temporarily unavailable during a vision-based navigation task. The method consists in analytically integrating the relation between the visual features motion in the image and the camera motion. Then, we use this approach to design sensor-based control laws able to tolerate the complete loss of the visual data during a vision-based navigation task in an unknown environment. Simulation and experimentation results demonstrate the validity and the interest of our method.***

**Keywords**: Mobile robots, visual servoing, image features loss, obstacle avoidance, visibility management.

## 1. INTRODUCTION

Visual servoing techniques aim at controlling the robot motion using vision data provided by a camera to reach a desired goal defined in the image (Hutchinson et al. 1996; Chaumette and Hutchinson 2006). Thus, these approaches require that the image features remain always visible. Therefore, these techniques cannot be used anymore if these features are lost during the robotic task. A first common solution is to use methods allowing to preserve the visual features visibility. Most of them are dedicated to manipulator arms, and propose to treat this kind of problem by using redundancy (Marchand and Hager 1998; Mansard and Chaumette 2005), path-planning (Mezouar and Chaumette 2002), specific degrees of freedom (DOF) (Corke and Hutchinson 2001; Kyrki et al. 2004), zoom (Benhimane and Malis 2003) or even by making a tradeoff with the nominal vision-based task (Remazeilles et al. 2006). In a mobile robotic context we have to preserve not only the image data visibility, but also the robot safety. In that case, techniques allowing to avoid simultaneously collisions and visual data losses such as (Folio and Cadenat 2005a; Folio and Cadenat 2005b) appear to be limited. Indeed, this kind of methods is restricted to missions where an avoidance motion exists without leading to local minima (Folio 2007). Therefore, they do not allow to perform robotic tasks which cannot be realized if the visual features loss is not tolerated. A first solution is to let some of the features appear and disappear temporarily from the image as done in (Garcia-Aracil et al. 2005). However, this approach is limited to partial losses, and does not entirely solve the problem.

Address correspondence to David Folio and Viviane Cadenat, LAAS-CNRS; 7, avenue du Colonel Roche, F-31077 Toulouse, France. E-mails: dfolio@laas.fr, cadenat@laas.fr. Phone: +33 561 336 451.

## NOMENCLATURE

| | |
|---|---|
| $d_{\mathrm{coll}}$ | distance between the robot and obstacles |
| $d_{\mathrm{occ}}$ | distance between the visual data and occluding object (in pixel) |
| $D_-$, $D_0$ and $D_+$ | distances allowing to evaluate the occlusion risk $\mu_{\mathrm{occ}}$ (in pixel) |
| $d_-$, $d_0$ and $d_+$ | distances allowing to evaluate the collision risk $\mu_{\mathrm{coll}}$ |
| $f$ | camera focal length |
| $\mathbf{J}_{\mathrm{r}}$ | robot jacobian matrix |
| $\mathbf{L}$ | the interaction matrix |
| $\dot{\mathbf{q}}$ | robot control input |
| $\boldsymbol{s}$ | visual features |
| $T_{\mathrm{e}}$ | sampling time |
| $\mathbf{v}_{\mathrm{r}}^{\mathrm{c}}$ | the camera kinematic screw |
| $v$ | mobile-base linear velocity |
| $V$ | linear velocity |
| $(X, Y)$ | coordinates of a point in image frame |
| $z$ | the point depth in camera frame |
| **Greek Symbols** | |
| $\vartheta$ | pan-platform orientation |
| $\mu_{\mathrm{coll}}$ | risk of collision |
| $\mu_{\mathrm{occ}}$ | risk of occlusion |
| $\Omega$ | angular velocity |
| $\omega$ | mobile-base angular velocity |
| $\varpi$ | pan-platform angular velocity |
| **Subscripts** | |
| $\overrightarrow{x}_c$, $\overrightarrow{y}_c$, $\overrightarrow{z}_c$ | the camera frame axis |
| cg | center of gravity |
| coll | collision |
| occ | occlusion |
| vs | visual servoing |
| $k$ | evaluate at time $t_k$ |

In this paper, our first goal is to propose a method allowing to compute the visual features when they are lost or unavailable for a short amount of time during a vision-based task (eg. temporary camera or image processing failure, landmark occlusion,...). In this work, the *whole* image is considered to be **momentarily entirely unavailable** in order to treat total visual data losses. Our second motivation is to use this approach to design sensor-based control laws able to tolerate the total loss of the image data during a vision-based task to reach the desired goal defined in the image.

The article is organized as follows. In section 2., we present the robotic system and state the problem. In section 3., we describe our estimation method. In section 4., we address some control issues and show how to use the proposed estimation technique to design control laws able to treat the visual features loss. Finally, we present some experimental results to validate our work.

## 2. THE ROBOTIC SYSTEM AND PROBLEM STATEMENT

First, let us describe the robotic platform, before modelling our system. We consider the mobile robot SuperScout II[1] equipped with an ultrasonic sensor belt and a camera mounted on a pan-platform (see figure 1.a). This is a cylindric cart-like vehicle, dedicated to indoor navigation. Incremental encoders provide the pose and the velocity of the robotic system. A DFW-VL500 Sony color digital IEEE 1394 camera captures pictures in YUV 4:2:2 format with $640 \times 480$ resolution. An image processing module extracts interest points from the image. The robot is controlled by an onboard laptop computer running under Linux on which is installed a specific control architecture called $\mathrm{G^{en}{}_{o}M}$ (Generator of Module) (Fleury and Herrb 2001).

[1] The mobile robot SuperScout II is provided by the AIP-PRIMECA (http://www.aip-primeca.net).

Let us modelling our system to express the camera kinematic screw. To this aim, by considering figure 1.b, we define the following successive frames: $\mathcal{F}_M$ $(M, \overrightarrow{x}_M, \overrightarrow{y}_M, \overrightarrow{z}_M)$ linked to the robot, $\mathcal{F}_P$ $(P, \overrightarrow{x}_P, \overrightarrow{y}_P, \overrightarrow{z}_P)$ attached to the pan-platform, and $\mathcal{F}_c$ $(C, \overrightarrow{x}_c, \overrightarrow{y}_c, \overrightarrow{z}_c)$ linked to the camera. Let $\vartheta$ be the direction of the pan-platform wrt. $\overrightarrow{x}_M$, $P$ the pan-platform center of rotation and $D_x$ the distance between the robot reference point $M$ and $P$. The control input is defined by: $\dot{\mathbf{q}} = [v, \omega, \varpi]^T$, where $v$ and $\omega$ are the cart linear and angular velocities, and $\varpi$ is the pan-platform angular velocity wrt. $\mathcal{F}_M$. For this specific mechanical system, the camera kinematic screw $\mathbf{v}^c = [\boldsymbol{V}^T_{C/\mathcal{F}_0}, \boldsymbol{\Omega}^T_{\mathcal{F}_C/\mathcal{F}_0}]^T$ is related to the control input by the robot jacobian $\mathbf{J}$: $\mathbf{v}^c = \mathbf{J}\dot{\mathbf{q}}$. As the camera is constrained to move horizontally, it is sufficient to consider a reduced kinematic screw $\mathbf{v}^c_{\mathrm{r}}$, and a reduced jacobian matrix $\mathbf{J}_{\mathrm{r}}$ as follows:

$$\mathbf{v}^c_{\mathrm{r}} = \begin{pmatrix} V_{\overrightarrow{y}_c} \\ V_{\overrightarrow{z}_c} \\ \Omega_{\overrightarrow{x}_c} \end{pmatrix} = \begin{pmatrix} -\sin(\vartheta) & D_x\cos(\vartheta) + C_x & C_x \\ \cos(\vartheta) & D_x\sin(\vartheta) - C_y & -C_y \\ 0 & -1 & -1 \end{pmatrix} \begin{pmatrix} v \\ \omega \\ \varpi \end{pmatrix} = \mathbf{J}_{\mathrm{r}}(\vartheta)\,\dot{\mathbf{q}} \quad (1)$$

where $C_x$ and $C_y$ are the coordinates of $C$ along axes $\overrightarrow{x}_P$ and $\overrightarrow{y}_P$ (see figure 1.b).

We consider the problem of executing a vision-based navigation task in an unknown environment. During such a task, unexpected events may occur: camera or image processing failure, landmark losses, collisions with obstacles, and so on. In this paper, we focus on the problem of the visual features temporary loss during the mission and we propose a method allowing to compute them when they become unavailable. We then show how to embed it in our vision-based navigation task to reach the desired goal defined in the image frame.

## 3. VISUAL DATA ESTIMATION

In this section, we address the image data estimation problem. We first present some preliminaries and state the estimation problem before presenting our approach.

### 3.1. Preliminaries

We consider a static landmark with respect to which is defined the vision-based navigation task to be realized. We assume that it can be characterized by $n$ interest points which can be extracted by our image processing. Then, in this case, the considered visual data are represented by a $2n$-dimensional vector $\boldsymbol{s}$ made of the coordinates $(X_i, Y_i)$ of each projected point $P_i$ in the image plane (see figure 1.c), that is: $\boldsymbol{s} = [X_1, Y_1, \ldots, X_i, Y_i, \ldots, X_n, Y_n]^T$. For a fixed landmark, the variation of the visual signals $\dot{\boldsymbol{s}}$ is related to the reduced camera kinematic screw $\mathbf{v}^c_{\mathrm{r}}$ by means of the interaction matrix $\mathbf{L}_{(\boldsymbol{s})}$ as shown below (Espiau et al. 1992):

$$\dot{\boldsymbol{s}} = \mathbf{L}_{(\boldsymbol{s})}\,\mathbf{v}^c_{\mathrm{r}} = \mathbf{L}_{(\boldsymbol{s})}\,\mathbf{J}_{\mathrm{r}}\,\dot{\mathbf{q}} \quad (2)$$

In our specific case, the interaction matrix $\mathbf{L}_{(\boldsymbol{s})}$ is deduced from the optic flow equations. Indeed, for one point $P_i$ it is given by (Espiau et al. 1992):

$$\mathbf{L}_{(P_i)} = \begin{pmatrix} \mathbf{L}_{(X_i, z_i)} \\ \mathbf{L}_{(Y_i, z_i)} \end{pmatrix} = \begin{pmatrix} 0 & \frac{X_i}{z_i} & \frac{X_i Y_i}{f} \\ -\frac{f}{z_i} & \frac{Y_i}{z_i} & f + \frac{{Y_i}^2}{f} \end{pmatrix} \quad (3)$$

where $z_i$ is the depth of each projected point $P_i$ and $f$ is the camera focal length (see figure 1.c). Therefore, the whole interaction matrix $\mathbf{L}_{(s)}= [\ldots, \mathbf{L}_{(X_i,z_i)}{}^T, \mathbf{L}_{(Y_i,z_i)}{}^T, \ldots]^T$ depends on the depth $\boldsymbol{z} = [z_1, \ldots, z_n]^T$ of the considered visual data.

### 3.2. Estimation Problem Statement

Now, we focus on the problem of estimating (*all* or some) visual data $\boldsymbol{s}$ whenever they become unavailable during a vision-based task. Thus, the key-assumption which underlies our works is that the whole image is considered to be **temporarily completely unavailable**. Hence, the methods which only allow to treat partial losses of the visual features such as (Malis et al. 1999; Garcia-Aracil et al. 2005; Comport et al. 2004) are not suitable here. Following this reasoning, we have focused on techniques dedicated to image data reconstruction. Different approaches, such as signal processing techniques or tracking methods (Favaro and Soatto 2003; Jackson et al. 2004; Lepetit and Fua 2006), may be used to deal with this kind of problem. Here, we have chosen to use a simpler approach for several reasons. First of all, most of the above techniques rely on measures from the image which are considered to be totally unavailable in our case. Second, we suppose that we have few errors on the model and on the measures[2]. Third, as it is intended to be used to perform complex navigation tasks, the estimated visual signals must be provided sufficiently rapidly with respect to the control law sampling period. Another idea is to use a 3D model of the object together with projective geometry in order to deduce the lacking data. However, this choice would lead to depend on the considered landmark type and would require to localize the robot. This was unsuitable for us, as we do not want to make any assumption on the landmark 3D model. We have finally chosen to solve (2) using the previous available image measurements and the control inputs $\dot{\mathbf{q}}$. Thus, we assume in the sequel that the visual data can be measured at the beginning of the robotic task.

Therefore, our idea is to solve the dynamic system (2) to reconstruct the image data $\boldsymbol{s}$. However, as, in our case, (2) requires the depth $\boldsymbol{z}$, we have to compute it to be able to solve this system. Several approaches may be used to determine $\boldsymbol{z}$. First of all, it can be measured using specific sensors such as a telemeter or a stereoscopic system. Nevertheless, as none of them is mounted on our robot, we have to focus on approaches allowing to reconstruct it. For instance, a first possible solution is to use structure from motion (SFM) techniques (Jerian and Jain 1991; Chaumette et al. 1996; Soatto and Perona 1998; Oliensis 2002), signal processing methods (Matthies et al. 1989), or even pose relative estimation (Thrun et al. 2001). Unfortunately, these approaches require to use measures from the image, and, thus, they cannot be applied here when it becomes unavailable. That is why we propose another solution consisting in estimating depth $\boldsymbol{z}$ together with the visual data $\boldsymbol{s}$ (see remark 1). Our idea is to express the analytical relation between the variation of $\boldsymbol{z}$ and the camera motion. As we consider a target made of $n$ points, we only have to determine the depth variation of each of these points. It can be easily shown that, for one 3D point $p_i$ of coordinates $(x_i, y_i, z_i)^T$ expressed in $\mathcal{F}_c$ projected into a point $P_i(X_i, Y_i)$ in the image plane as shown in figure 1.c, the depth variation $\dot{z}_i$ is related to the camera motion according to: $\dot{z}_i = \mathbf{L}_{(\boldsymbol{z}_i)} \mathbf{v}_{\mathrm{r}}^c = \mathbf{L}_{(\boldsymbol{z}_i)} \mathbf{J}_{\mathrm{r}} \dot{\mathbf{q}}$. In our case, $\mathbf{L}_{(\boldsymbol{z}_i)} = (0,\, -1,\, \frac{z_i}{f} Y_i)$. Then,

[2] In the case where this assumption is not fulfilled, different techniques such as Kalman filtering for instance may be used to take into account explicitely the different noises present on the system.

the dynamic system to be solved for one point $P_i(X_i, Y_i)$ is given by:

$$\begin{aligned} \dot{X}_i &= \tfrac{X_i}{z_i} V_{\overrightarrow{z}_c} + \tfrac{X_i Y_i}{f} \Omega_{\overrightarrow{x}_c} && (4) \\ \dot{Y}_i &= -\tfrac{f}{z_i} V_{\overrightarrow{y}_c} + \tfrac{Y_i}{z_i} V_{\overrightarrow{z}_c} + \left(f + \tfrac{Y_i^2}{f}\right) \Omega_{\overrightarrow{x}_c} && (5) \\ \dot{z}_i &= -V_{\overrightarrow{z}_c} - \tfrac{z_i}{f} Y_i \Omega_{\overrightarrow{x}_c} && (6) \end{aligned}$$

where $V_{\overrightarrow{y}_c}$, $V_{\overrightarrow{z}_c}$ and $\Omega_{\overrightarrow{x}_c}$ are given in (1). Thus, our goal is to integrate the above differential equations during the control law sampling period $T_e$, that is for any $t \in [t_k, t_{k+1}]$. Then, introducing $\psi = [\boldsymbol{s}^T, \boldsymbol{z}^T]^T$ and $\mathbf{L}_{(\psi)} = [\mathbf{L}_{(s)}{}^T, \mathbf{L}_{(\boldsymbol{z})}{}^T]^T$, the differential system to be solved can be expressed as follows:

$$\begin{cases} \dot{\psi} \ = \mathbf{L}_{(\psi_k)}\, \mathbf{v}_{\mathrm{r}}^c = \mathbf{L}_{(\psi_k)}\, \mathbf{J}_{\mathrm{r}(\vartheta(t))}\, \dot{\mathbf{q}}(t_k) \\ \psi_k = \psi(t_k) = \left(\boldsymbol{s}_k^T, \boldsymbol{z}_k^T\right)^T = (X_{1k}, Y_{1k}, \ldots, X_{nk}, Y_{nk}, z_{1k}, \ldots, z_{nk})^T \end{cases} \qquad (7)$$

where $\psi_k$ and $\dot{\mathbf{q}}(t_k) = (v_k, \omega_k, \varpi_k)^T$ are the values of $\psi$ and $\dot{\mathbf{q}}$ evaluated at $t_k$ (see remark 1). As, on our robot, the control input is hold during $T_e$, the velocities vector $\dot{\mathbf{q}}$ can be considered as constant during this time interval.

***Remark 1:*** *While the image remains available, $\boldsymbol{s}_k$ is directly obtained from the image features extraction processing, and the initial depth $\boldsymbol{z}_k$ can be computed using one of the methods listed above: SFM, signal processing techniques, pose relative estimation approaches... When the image is lost, $\psi_k = \left(\boldsymbol{s}_k^T, \boldsymbol{z}_k^T\right)^T$ is determined from our estimator (see figure 2).*

***Remark 2:*** *Another idea to reconstruct the visual features would be to consider the exponential map approach and the direct measure of the camera velocity. This approach allows to determine the 3D coordinates of the point $p_i(x_i, y_i, z_i)$ using the following relation (Soatto and Perona 1998):*

$$\dot{p}_i = -V_{C/\mathcal{F}_0}(t) - \Omega_{\mathcal{F}_C/\mathcal{F}_0}(t) \wedge p_i(t) \quad \leftrightarrow \quad p_i(t_{k+1}) = R(t_k) p_i(t_k) + T(t_k)$$

*where $R \in SO(3)$ (special orthogonal group of transformations of $\mathbb{R}^3$) and $T \in \mathbb{R}^3$ define respectively the rotation and translation of the moving camera. Indeed, $R$ and $T$ are related to the camera rotation $\Omega_{\mathcal{F}_C/\mathcal{F}_0}$ and translation $V_{C/\mathcal{F}_0}$ motion thanks to an exponential map (Murray et al. 1994), that is:*

$$\begin{pmatrix} R & T \\ 0 & 1 \end{pmatrix} = \exp \begin{pmatrix} [\Omega]_\times & V_{C/\mathcal{F}_0} \\ 0 & 0 \end{pmatrix}$$

*where $[\Omega]_\times$ belongs to the set of $3 \times 3$ skew-matrices and is commonly used to describe the cross product of $\Omega_{\mathcal{F}_C/\mathcal{F}_0}$ with a vector in $\mathbb{R}^3$. However, this approach can be used only if the camera kinematic screw $\mathbf{v}^c = [V_{C/\mathcal{F}_0}^T, \Omega_{\mathcal{F}_C/\mathcal{F}_0}^T]^T$ can be assumed constant during the sampling period $T_e$, which is not our case.*

### 3.3. Resolution

A first idea is to solve differential system (7) using numerical schemes as done in (Folio and Cadenat 2007a; Folio and Cadenat 2007b). In these previous works, different numerical techniques such as Euler, Runge-Kutta, Adams-Bashforth-Moulton or Backward Differentiation Formulas have been used and compared. The main advantage of this kind of approaches is that it can be applied to any set of image features and to any kind of robots, provided that the estimation problem can be expressed using system (7). However, it is well known that numerical schemes always approximate the "true" solution and that the accuracy of the obtained result is closely related to the chosen method and to the integration step size. Here, as the considered visual data and robotic system are sufficiently simple, system (7) can be solved analytically. Although the obtained solution is restricted to our specific case, it appears to be much more accurate than the previous ones, as it is the **exact solution** of system (7) on the time control interval $T_e$. This is its main advantage.

Now, let us determine the analytical solution of the differential equation (7). In order to simplify the expressions, we will only treat the case of one point $P$ of coordinates $\{X(t), Y(t), z(t)\}$, which leads to solve the set of equations (4), (5) and (6). Subscript $i$ will not then be considered in the following calculus.

First, we compute the depth $z(t)$. To this aim, we derivate equation (6) with respect to time:

$$\ddot{z} = -V_{\vec{z}_c} - (\dot{Y}z(t) + Y(t)\dot{z})\frac{\Omega_{\vec{x}_c}}{f} - \frac{Y(t)z(t)}{f}\dot{\Omega}_{\vec{x}_c} \tag{8}$$

Then, multiplying (5) by $z(t)$ and (6) by $Y(t)$, we get:

$$\dot{Y}z(t) + Y(t)\dot{z} = -fV_{\vec{y}_c}(t) + fz(t)\Omega_{\vec{x}_c}(t)$$

Moreover, using (1), it can be shown that: $\dot{V}_{\vec{z}_c} = (-\upsilon_k \sin(\vartheta(t)) + D_x\omega_k \cos(\vartheta(t)))\,\varpi_k$ and $\dot{\Omega}_{\vec{x}_c} = 0$. Replacing these different terms in (8), we finally obtain the following second order differential equation to be solved for $z(t)$:

$$\ddot{z} + z(t)(\omega_k + \varpi_k)^2 = C_x\varpi_k(\omega_k + \varpi_k) - (\omega_k + 2\varpi_k)V_{\vec{y}_c}(t) \tag{9}$$

After some computations, we get the following results for $z(t)$ depending on the control inputs $\omega_k$ and $\varpi_k$ (a detailed proof is available in (Folio 2007)):

$$\begin{cases} \bullet \text{ if } \omega_k \neq -\varpi_k \text{ and } \omega_k \neq 0 : \\ z(t) = c_1 \sin\left(A_1\left(t - t_k\right)\right) + c_2 \cos\left(A_1\left(t - t_k\right)\right) \\ \qquad - D_x \cos\left(\vartheta(t)\right) + \frac{\upsilon_k}{\omega_k} \sin\left(\vartheta(t)\right) - C_x \\ \bullet \text{ if } \omega_k = -\varpi_k \neq 0 : \\ z(t) = \frac{\omega_k}{\varpi_k} D_x \left(\cos\left(\vartheta(t)\right) - \cos\left(\vartheta_k\right)\right) \\ \qquad - \frac{\upsilon_k}{\varpi_k}\left(\sin\left(\vartheta(t)\right) - \sin\left(\vartheta_k\right)\right) + z_k \\ \bullet \text{ if } \omega_k = 0 \text{ and } \varpi_k \neq 0 : \\ z(t) = c_3 \sin\left(\varpi_k\left(t - t_k\right)\right) + c_4 \cos\left(\varpi_k\left(t - t_k\right)\right) \\ \qquad - \upsilon_k\left(t - t_k\right) \cos\left(\vartheta(t)\right) + \frac{\upsilon_k}{2\varpi_k} \sin\left(\vartheta(t)\right) - C_x \\ \bullet \text{ if } \omega_k = \varpi_k = 0 : \\ z(t) = -\upsilon_k \cos\left(\vartheta_k\right)\left(t - t_k\right) + z_k \end{cases} \text{, with: } \begin{cases} A_1 = (\omega_k + \varpi_k) \\ c_1 = \frac{\upsilon_k}{\omega_k} \cos\left(\vartheta_k\right) + D_x \sin\left(\vartheta_k\right) \\ \qquad - \frac{Y_k z_k}{f} - C_y \\ c_2 = D_x \cos\left(\vartheta_k\right) + z_k \\ \qquad - \frac{\upsilon_k}{\omega_k} \sin\left(\vartheta_k\right) + C_x \\ c_3 = \frac{\upsilon_k}{2\varpi_k} \cos\left(\vartheta_k\right) \\ \qquad - \frac{Y_k z_k}{f} - C_y \\ c_4 = -\frac{\upsilon_k}{2\varpi_k} \sin\left(\vartheta_k\right) + z_k + C_x \end{cases} \tag{10}$$

Now, we consider the determination of $X(t)$. Equation (4) can be rewritten as: $\frac{\dot{X}}{X(t)} = -\frac{\dot{z}}{z(t)}$. $X(t)$ can then be computed by integrating this last relation for $t \in [t_k, t_{k+1}]$, and we get:

$$X(t) = \frac{z_k X_k}{z(t)} \tag{11}$$

Finally, $Y(t)$ is easily deduced from (6) as follows: $Y(t) = -f\frac{\dot{z} + V_{\vec{z}_c}(t)}{z(t)\Omega_{\vec{x}_c}(t)}$, where $V_{\vec{z}_c}(t)$ and $\Omega_{\vec{x}_c}(t)$ can be computed from the control inputs by using (1). Using the solution $z(t)$ given by (10), $Y(t)$ expresses as:

$$\begin{cases} \bullet \text{ if } \omega_k \neq -\varpi_k \text{ and } \omega_k \neq 0 : \\ Y(t) = -f \frac{C_y - c_1\cos(A_1(t-t_k)) - c_2\sin(A_1(t-t_k)) - D_x\sin(\vartheta(t)) - \frac{v_k}{\omega_k}\cos(\vartheta(t))}{z(t)} \\ \bullet \text{ if } \omega_k = -\varpi_k \neq 0 : \\ Y(t) = f \frac{v(\cos(\vartheta(t)) - \cos(\vartheta_k)) + \varpi_k z_k Y_k}{z(t)\varpi_k} \\ \bullet \text{ if } \omega_k = 0 \text{ and } \varpi_k \neq 0 : \\ Y(t) = -f \frac{c_1\cos(A_1(t-t_k)) - c_2\sin(A_1(t-t_k)) + v_k(t-t_k)\sin(\vartheta(t)) - \frac{v_k}{2\varpi_k}\cos(\vartheta(t)) + C_y}{z(t)} \\ \bullet \text{ if } \omega_k = \varpi_k = 0 : \\ Y(t) = -f \frac{v_k\sin(\vartheta_k)(t-t_k) + Y_k z_k}{z(t)} \end{cases} \qquad (12)$$

As $Y(t)$ depends on depth, we observe the same different cases as in $z(t)$, depending on the values of $\omega_k$ and $\varpi_k$. Moreover, the computation of $\{X(t), Y(t), z(t)\}$ requires the determination of the pan-platform direction: $\vartheta(t)$. This angle can be calculated by integrating $\dot{\vartheta} = \varpi_k$ between $t_k$ and $t$, that is: $\vartheta(t) = \varpi_k\,(t - t_k) + \vartheta_k$, where $\vartheta_k$ represents the pan-platform angular value at $t = t_k$, which is provided by the embedded encoder. Hence, as the robot jacobian depends on $\vartheta(t)$, the camera kinematic screw $\mathbf{v}_{\mathrm{r}}^{c}$ is not constant during the control law sampling time $T_{\mathrm{e}}$ (see remark 2).

Finally, the solution for the set of the $n$ points of the landmark is obtained by applying the above solution on each component of the previously defined vector $\psi$.

## 4. Applications

We propose in this section a panel of applications showing the interest and the relevance of our approach. Indeed, our estimation technique can be used in different related fields such as vision or image processing. Here, we have chosen to apply it in a vision-based context to compute the visual data whenever they are unavailable during the navigation task. Therefore, we will address some control issues and we will show how to use the proposed estimation method to design control laws able to treat the image data loss. We have considered two kinds of tasks: the first one is a positioning vision-based task during which a camera failure occurs; the second one is a more complex task consisting in realizing a visually guided navigation task amidst obstacles despite possible occlusions and collisions.

### 4.1. Execution of a Vision-Based Task Despite Camera Failure

Here, our objective is to perform a vision-based task despite a camera failure. We first describe the considered robotic task before presenting the control strategy and the obtained results.

**4.1.1. Vision-based task** Our goal is to position the embedded camera with respect to a landmark composed of $n$ points. To this aim, we have applied the visual servoing technique given in (Espiau et al. 1992) to mobile robots as in (Pissard-Gibollet and Rives 1995). The proposed approach relies on the task function formalism (Samson et al. 1991) and consists in expressing the visual servoing task by the following task function to be regulated to zero:

$$e_{\mathrm{vs}} = \mathbf{C}(\boldsymbol{s} - \boldsymbol{s}^{\star})$$

where $\boldsymbol{s}^\star$ represents the desired value of the visual signal, while $\mathbf{C}$ is a full-rank combination matrix allowing to take into account more visual features than available DOFs (Espiau et al. 1992). A classical controller $\dot{\mathbf{q}}_{(\boldsymbol{s})}$ making $e_{\text{vs}}$ vanish can be designed by imposing an exponential decrease, that is: $\dot{e}_{\text{vs}} = -\lambda_{\text{vs}} e_{\text{vs}} = \mathbf{C}\mathbf{L}_{(\boldsymbol{s})}\mathbf{J}_{\text{r}}\dot{\mathbf{q}}_{(\boldsymbol{s})}$, where $\lambda_{\text{vs}}$ is a positive scalar or a positive definite matrix. Fixing $\mathbf{C} = \mathbf{L}_{(\boldsymbol{s}^\star)}{}^{+}$ as in (Espiau et al. 1992), the visual servoing controller $\dot{\mathbf{q}}_{(\boldsymbol{s})}$ can be written as follows:

$$\dot{\mathbf{q}}_{(\boldsymbol{s})} = (\mathbf{C}\mathbf{L}_{(\boldsymbol{s})}\mathbf{J}_{\text{r}})^{-1}(-\lambda_{\text{vs}})\mathbf{L}_{(\boldsymbol{s}^\star)}{}^{+}(\boldsymbol{s} - \boldsymbol{s}^\star) \tag{13}$$

**4.1.2. Control strategy** As previously mentioned, the goal is to perform a positioning vision-based task with respect to a landmark, despite visual data loss due to a camera failure. The robot will be controlled in different ways, depending on the visual data availability. Two cases may occur: either the camera is able to provide the visual data or not (see figure 2). In the first case, controller (13) can be directly applied to the robot and the task is executed as usually done. In the second case, we use equations (11) and (12) to compute an estimation $(\widetilde{X}_i, \widetilde{Y}_i)$ of $(X_i, Y_i)$ of each point of the considered landmark. It is then possible to deduce the estimated visual data $\widetilde{s}$, and to use it to evaluate controller (13). Hence, during a camera failure, the vehicle is driven by a new controller: $\dot{\mathbf{q}}_{(\widetilde{\boldsymbol{s}})} = (\mathbf{C}\mathbf{L}_{(\widetilde{\boldsymbol{s}})}\mathbf{J}_{\text{r}})^{-1}(-\lambda_{\text{vs}})\mathbf{L}_{(\boldsymbol{s}^\star)}{}^{+}\,(\widetilde{\boldsymbol{s}} - \boldsymbol{s}^\star)$. Therefore, we propose to use the following global visual servoing controller:

$$\dot{\mathbf{q}}_{\text{vs}} = (1-\sigma)\dot{\mathbf{q}}_{(\boldsymbol{s})} + \sigma\dot{\mathbf{q}}_{(\widetilde{\boldsymbol{s}})} \tag{14}$$

where $\sigma = \{0, 1\}$ is a flag set to 1 when the image is unavailable.

***Remark 3:*** *There is no need to smooth controller* (14) *when the image features are lost and recovered. Indeed, when the occlusion occurs, the last provided information are used to feed our reconstruction algorithm and the values of $\boldsymbol{s}$ and $\widetilde{\boldsymbol{s}}$ are close. In a similar way, when the visual data are available anew, the same result holds, provided that the estimation errors remain small. If not, it will be necessary to smooth the controller by defining $\sigma$ as a continuous function of time $t$ for instance.*

**4.1.3. Simulation results** To validate our work, we have first realized numerous simulations using Matlab software. Moreover, to compare the different approaches efficiency, we will show the results obtained with the proposed analytical solutions and with the numerical schemes used in our previous works (Folio and Cadenat 2007a; Folio and Cadenat 2007b). Four classical algorithms have been considered: the well-known first order Euler scheme and three fourth order methods, namely the Runge-Kutta (RK4), Adams-Bashforth-Moulton (ABM) and Backward Differentiation Formulas (BDF) techniques. Therefore, we have simulated a mission whose objective is to position the camera relatively to a landmark made of $n = 9$ points. For each test, the initial conditions are identical, and the robot realizes the same navigation task (see figure 3.a). At the beginning of the task, the image data are available and the robot is then controlled using $\dot{\mathbf{q}}_{(\boldsymbol{s})}$. After 20 steps, we simulate a camera failure, then our estimation procedure is enabled and the robot is driven using controller $\dot{\mathbf{q}}_{(\widetilde{\boldsymbol{s}})}$ until the end of the mission.

Figures 3.b and 3.c show the values of the estimation errors norms $\|\boldsymbol{s} - \widetilde{\boldsymbol{s}}\|$ and $\|\boldsymbol{z} - \widetilde{\boldsymbol{z}}\|$ for each considered numerical schemes and the analytical solution. As previously mentioned, the latter is the most accurate approach as the obtained errors are close to zero. This result appears to be consistent, as the proposed method provides the **exact** solution of the considered dynamical system for any $t \in [t_k, t_{k+1}]$, whereas numerical schemes can only give an approximate result with cumulative errors. Moreover, the conditions of simulation are "ideal", that is there is no noise and every necessary data is perfectly known. This is the reason why the errors obtained when using the analytical solution are nearly zero.

**4.1.4. Experimental results** We have also experimented our approach on our SuperScout II. We have considered a vision-based navigation task which consists in positioning the embedded camera in front of a given landmark made of $n = 4$ points. Figure 4.a shows the trajectory performed by the robot. As previously, at the beginning of the task, the visual data are available and the robot is then controlled using $\dot{\mathbf{q}}_{(\boldsymbol{s})}$. After a few steps, the landmark is manually occluded. At this time, the visual signals are computed by our estimation procedure and the robot is driven using controller $\dot{\mathbf{q}}_{(\widetilde{\boldsymbol{s}})}$. It is then possible to keep on executing a task which would have aborted otherwise.

Figures 4.b and 4.c show the value of the estimation errors norms $\|\boldsymbol{s} - \widetilde{\boldsymbol{s}}\|$ and $\|\boldsymbol{z} - \widetilde{\boldsymbol{z}}\|$. Two specific experimental aspects must be pointed out to analyze these results. First, our estimation algorithm requires an initialization phase (which lasts about 25 steps) to produce the necessary data ($z$-depth, etc.) and to respect some experimental synchronization constraints. Then, the errors are fixed to zero during this step (see figures 4.b and 4.c). Second, when the reconstruction process is launched, the estimation errors are deduced by comparing the computed image features to the ones provided by the camera. When the occlusion occurs, no more measure is available to evaluate the error. In that case, we determine it on the base of the last measure available and on the predicted value. Therefore, during this phase which is represented by the grey zone on figures 4.b and 4.c, the estimation errors are meaningless. When the image data are available anew, the errors can be computed again and we can then appreciate the accuracy of our estimation algorithm. As we can see, when the occlusion is over, the errors are small, showing the efficiency of the proposed approach.

As one could have expected, simulation provides better results than experimentation. In particular, when testing our algorithm, we have to deal with several constraints related to our robotic platform. First, our SuperScout II does not run under a real time operating system; thus, the sampling period $T_e$ is not known accurately. Moreover, our modelling does not take into account the noises which appear on the image features extraction processing (about 1 pixel) and on the measurement of the robot velocities $\dot{\mathbf{q}}$. Note that they are processed from the embedded encoders, as our platform is not equipped with dedicated sensors. Nevertheless, despite these typical experimental constraints, and although the $z$-depth estimation is not very accurate, it is important to note that the image data remain properly estimated (as $\|\boldsymbol{s} - \widetilde{\boldsymbol{s}}\| < 1$ pixel). The robot can then reach the desired goal defined in the image.

### 4.2. Dealing with occlusions and collisions

Now, we aim at demonstrating the interest and the efficiency of our estimation technique to treat the specific problem of occlusions during a vision-based navigation task in a cluttered environment. When executing this kind of task, two problems must be addressed: the visual data loss and the risk of collision. A first solution is to design control strategies allowing to avoid simultaneously these two problems as proposed in (Folio and Cadenat 2005a; Folio and Cadenat 2005b). However, these methods are restricted to missions where such an avoidance motion is possible without leading to local minima (Folio 2007). Moreover, some robotic tasks cannot be performed if the visual features loss is not tolerated. Thus, trying to avoid systematically collisions and occlusions does not always appear as the most relevant strategy. In this context, the estimation technique developed above can be an interesting tool to enhance the range of realizable navigation tasks, provided that it can be embedded in the control strategy. Therefore, in this part, we show how to integrate the proposed estimation technique in the control law to treat the occlusions or camera failure. We first briefly describe how to detect the risks of occlusion and collision. In a second step, we detail our sensor-based control strategy before presenting some experimental results to validate our approach.

**4.2.1. Collision and occlusion detection** To perform the desired task, it is necessary to detect the *risks of collision* and *occlusion*. The danger of collision is evaluated using the data provided by the embedded ultrasonic sensors. From these data, it is possible to compute the distance $d_{\text{coll}}$ and the relative orientation $\alpha$ between the robot and the obstacle (see figure 5.a). Let $\xi_+$, $\xi_0$, $\xi_-$ be three envelopes which surround each obstacle at distances $d_+ > d_0 > d_-$. It is then possible to model the risk of collision by a parameter $\mu_{\text{coll}}(d_{\text{coll}})$ which smoothly increases from 0 when the robot is far from the obstacle ($d_{\text{coll}} > d_0$) to 1 when it is close to it ($d_{\text{coll}} < d_-$). The occlusion risk is evaluated in the image by detecting the occluding object left and right borders. From these data, we can deduce the shortest distance $d_{\text{occ}}$ between the image features and the occluding object $\mathcal{O}$ as shown on figure 5.b. Defining three envelopes $\Xi_+$, $\Xi_0$, $\Xi_-$ around the occluding object located at $D_+ > D_0 > D_-$ from it, we propose to model the risk of occlusion by a parameter $\mu_{\text{occ}}(d_{\text{occ}})$ which smoothly increases from 0 when $\mathcal{O}$ is far from the visual features ($d_{\text{occ}} > D_0$) to 1 when it is close to them ($d_{\text{occ}} < D_-$). Any expression satisfying the previous constraints can be used to define $\mu_{\text{coll}}(d_{\text{coll}})$ and $\mu_{\text{occ}}(d_{\text{occ}})$. A possible choice for these parameters can be found in (Folio and Cadenat 2005a).

**4.2.2. Global control law design** Our global control strategy relies on the risks of collision and occlusion, that is on parameters $\mu_{\text{coll}}(d_{\text{coll}})$ and $\mu_{\text{occ}}(d_{\text{occ}})$. It consists in two steps. First, we design two controllers allowing respectively to realize the sole vision-based task and to guarantee non-collision, while dealing with occlusions if needed. Second, we switch between these two controllers depending on the risk of collision. There exist several approaches for sequencing robotic tasks (Pissard-Gibollet and Rives 1995; Souères and Cadenat 2003; Mansard and Chaumette 2007)… Here, we have chosen a method which relies on convex combinations between the successive controllers (Cadenat et al. 1999). In that case, applications can be more easily carried out, but it is usually harder to guarantee the task feasibility (see remark 4). Thus, we propose the following global controller:

$$\dot{\mathbf{q}} = (1 - \mu_{\text{coll}}(d_{\text{coll}}))\, \dot{\mathbf{q}}_{\text{vs}} + \mu_{\text{coll}}(d_{\text{coll}})\dot{\mathbf{q}}_{\text{coll}} \tag{15}$$

where $\dot{\mathbf{q}}_{\text{vs}}$ is the previously defined visual servoing controller (14), while $\dot{\mathbf{q}}_{\text{coll}} = (v_{\text{coll}}, \omega_{\text{coll}}, \varpi_{\text{coll}})^T$ handles obstacle avoidance and visual signal estimation if necessary. This controller will be designed in the sequel.

***Remark 4:*** *Let us notice that $\mu_{\text{coll}}(d_{\text{coll}})$ and $\mu_{\text{occ}}(d_{\text{occ}})$ are defined to be maintained to 1 once they have reached this value (Folio and Cadenat 2005a). Moreover, the different envelops are chosen close enough to reduce the transition phase duration. Thus the control strategy is built to insure that the robot will be rapidly controlled by the most relevant controller. In this way, the risks of instability, target loss or collision during the switch are significantly reduced and the task feasibility can be considered to be guaranteed.*

**4.2.3. Obstacle avoidance** We propose to use a similar obstacle avoidance strategy to the one used in (Cadenat et al. 1999). The idea is to define around each obstacle a rotative potential field so that the repulsive force is orthogonal to the obstacle when the robot is close to it ($d_{\text{coll}} < d_{-}$), parallel to the obstacle when the vehicle is on $\xi_0$ envelop, and slightly directed towards the obstacle between $d_0$ and $d_+$ (see figure 5.a). The interest of such a potential is that it can make the robot move around the obstacle without requiring any attractive force, reducing local minima problems. This leads to the following potential function $\mathcal{X}_{(d_{\text{coll}})}$ (Cadenat et al. 1999):

$$\begin{cases} \mathcal{X}_{(d_{\text{coll}})} = \frac{1}{2}k_1\left(\frac{1}{d_{\text{coll}}} - \frac{1}{d_+}\right)^2 + \frac{1}{2}k_2\left(d_{\text{coll}} - d_+\right)^2 & \text{if } d_{\text{coll}} \leq d_+, \\ \mathcal{X}_{(d_{\text{coll}})} = 0 & \text{otherwise} \end{cases}$$

where $k_1$ and $k_2$ are positive gains to be chosen. The virtual repulsive force is defined to generate the desired rotative potential field around the obstacle by the couple $(F, \beta)$, where $F = -\frac{\partial \mathcal{X}_{(d_{\text{coll}})}}{\partial d_{\text{coll}}}$ is the modulus of the virtual repulsive force and $\beta = \alpha - \frac{\pi}{2d_0}d_{\text{coll}} + \frac{\pi}{2}$ its direction wrt. $\mathcal{F}_M$. The mobile base velocities $v_{\text{coll}}$ and $\omega_{\text{coll}}$ are then given by (Cadenat et al. 1999):

$$\dot{\mathbf{q}}_{\text{base}} = \begin{pmatrix} v_{\text{coll}} & \omega_{\text{coll}} \end{pmatrix}^T = \begin{pmatrix} k_v F \cos\beta & k_\omega \frac{F}{D_x}\sin\beta \end{pmatrix}^T \tag{16}$$

where $k_v$ and $k_\omega$ are positive gains to be chosen. Equation (16) drives only the mobile base in the obstacle neighborhood. However, if the pan-platform remains uncontrolled, it will be impossible to switch back to the execution of the vision-based task at the end of the avoidance phase. Therefore, the $\varpi_{\text{coll}}$ design problem must also be addressed. Two cases may occur in the obstacle vicinity: either the visual data are available or not. In the first case, the proposed approach is similar to (Cadenat et al. 1999) and the pan-platform is controlled to compensate the avoidance motion while centering the target in the image. As the camera is constrained to move within an horizontal plane, it is sufficient to regulate to zero the error $e_{\text{cg}} = Y_{\text{cg}} - Y_{\text{cg}}^\star$ where $Y_{\text{cg}}$ and $Y_{\text{cg}}^\star$ are the current and desired ordinates of the target gravity center. Rewriting equation (1) as $\mathbf{v}_{\text{r}}^c = \mathbf{J}_{\text{base}}\dot{\mathbf{q}}_{\text{base}} + \mathbf{J}_\varpi \varpi_{\text{coll}}$ and imposing an exponential decrease to regulate $e_{\text{cg}}$ to zero ($\dot{e}_{\text{cg}} = \mathbf{L}_{(Y_{\text{cg}})}\mathbf{v}_{\text{r}}^c = -\lambda_{\text{cg}}e_{\text{cg}}$, $\lambda_{\text{cg}} > 0$), we finally obtain (see (Cadenat et al. 1999) for more details):

$$\varpi_{\text{coll}} = \frac{-1}{\mathbf{L}_{(Y_{\text{cg}})}\mathbf{J}_\varpi}\left(\lambda_{\text{cg}}e_{\text{cg}} + \mathbf{L}_{(Y_{\text{cg}})}\mathbf{J}_{\text{base}}\dot{\mathbf{q}}_{\text{base}}\right) \tag{17}$$

where $\mathbf{L}_{(Y_{\text{cg}})}$ is the 2nd row of $\mathbf{L}_{(P_i)}$ evaluated for $Y_{\text{cg}}$ (see equation (3)). However, if the obstacle occludes the camera field of view, $\boldsymbol{s}$ is no more available and the pan-platform cannot be controlled anymore using (17). At this time, we compute the visual features using the analytical solutions (11) and (12), and we deduce an estimation $\widetilde{Y_{\text{cg}}}$ of $Y_{\text{cg}}$. The pan-platform controller during an occlusion phase will then be determined by replacing the real target gravity center ordinate $Y_{\text{cg}}$ by the computed one $\widetilde{Y_{\text{cg}}}$ in (17), that is: $\widetilde{\varpi}_{\text{coll}} = \frac{-1}{\mathbf{L}_{(\widetilde{Y_{\text{cg}}})}\mathbf{J}_\varpi}(\lambda_{\text{cg}}\widetilde{e}_{\text{cg}} + \mathbf{L}_{(\widetilde{Y_{\text{cg}}})}\mathbf{J}_{\text{base}}\dot{\mathbf{q}}_{\text{base}})$, where $\widetilde{e}_{\text{cg}} = \widetilde{Y_{\text{cg}}} - Y_{\text{cg}}^\star$, and $\mathbf{L}_{(\widetilde{Y_{\text{cg}}})}$ is deduced from (3). Now, it remains to apply the suitable controller to the pan-platform depending on the context. Recalling that parameter $\mu_{\text{occ}} \in [0; 1]$ allows to detect occlusions, we propose the following avoidance controller:

$$\dot{\mathbf{q}}_{\text{coll}} = \begin{pmatrix} v_{\text{coll}}, & \omega_{\text{coll}}, & (1 - \mu_{\text{occ}})\varpi_{\text{coll}} + \mu_{\text{occ}}\widetilde{\varpi}_{\text{coll}} \end{pmatrix}^T \tag{18}$$

Thanks to this controller, it is possible to avoid the obstacle while treating — if needed — the temporary visual features occlusion.

**4.2.4. Experimental results** We present hereafter some experimental results to validate the proposed approach. In this example, the landmark is made of $n = 4$ points. We still consider a positioning navigation task, but, now the environment is cluttered with one obstacle. Envelopes $\xi_-$, $\xi_0$ and $\xi_+$ are respectively located at $d_- = 40$cm, $d_0 = 56$cm and $d_+ = 70$cm from the obstacle. Figure 6 present the obtained results. As one can see, the mission is correctly performed despite the obstacles. Now, let us describe the sequence of the different events which occurred during the task. At the beginning of the task, there is no risk of collision, and the visual features are available. The robot is then controlled using $\dot{\mathbf{q}}_{(\boldsymbol{s})}$ and starts converging towards the landmark. Then, it enters the vicinity of the obstacle, inducing a danger of collision. $\mu_{\text{coll}}(d_{\text{coll}})$ increases to reach 1 (see figure 6.c) and the control law applied to the vehicle smoothly switches to controller $\dot{\mathbf{q}}_{\text{coll}}$ (18). The linear velocity decreases while the angular velocities $\omega_{\text{coll}}$ and $\varpi_{\text{coll}}$ grow (see figure 6.b) to make the mobile base avoid the obstacle, while the pan-platform centers the target in the image. When the obstacle is overcome, $\mu_{\text{coll}}$ vanishes, the control progressively switches back to $\dot{\mathbf{q}}_{(\boldsymbol{s})}$ and the robot is then driven to converge towards the target. However, at this time, the landmark is manually occluded and we apply controller $\dot{\mathbf{q}}_{(\widetilde{\boldsymbol{s}})}$ instead of $\dot{\mathbf{q}}_{(\boldsymbol{s})}$. The vehicle can then keep on executing the mission despite the image data loss. The vision-based task is then successfully realized (see figure 6.a).

Now, let us analyze the efficiency of the reconstruction. To this aim, we consider figures 6.d and 6.e which present the values of the norm of the errors $\|s - \widetilde{s}\|$ and $\|z - \widetilde{z}\|$. These figures show that these errors increase suddenly when we switch from one controller to the other. This phenomenon is closely related to the small delay and drift with which the embedded encoders provide the velocities to our estimation algorithm. Nonetheless, despite these problems, the errors remain small, showing the relevance of our approach.

## 5. CONCLUSIONS AND FUTURE WORKS

In this paper, we have considered the problem of the visual features loss during a vision-based navigation task. First of all, we have developed a method allowing to reconstruct these data when they become temporarily unavailable because of an occlusion or any other unexpected event. Then, we have shown how to embed this method in our sensor-based control laws to successfully perform the desired missions, despite the total loss of the image features. Our approach has been properly validated by experimental tests. Although the proposed reconstruction technique is subject to drift and limited to short duration visual signal losses, it offers a nice solution to recover from a problem which generally leads to a task failure. The obtained results clearly demonstrates the relevance and the interest of the designed approach to improve the mission execution. Thanks to our method, a wider range of visually guided navigation tasks is now realizable.

However, the developed analytical solutions are restricted to landmarks which can be characterized by points and dedicated to a particular robotic system. Therefore, a natural extension of these works would be to solve this problem for other kinds of visual features and more complex robots. Furthermore, it would be interesting to consider a moving landmark instead of a static one and to improve our global control strategy by sequencing dynamically the different controllers. Finally, we also aim at testing signal processing techniques such as Kalman filtering to take into account explicitely the different noises present on our real system.

## REFERENCES

Benhimane, S. and E. Malis. 2003. Vision-based control with respect to planar and non-planar objects using a zooming camera. *Proceedings of Int. Conf. on Advanced Robotics* 2: 991–996, Coimbra, Portugal.

Cadenat, V., P. Souères, R. Swain, and M. Devy. 1999. A controller to perform a visually guided tracking task in a cluttered environment. *Proceedings of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems* 2: 775–780, Korea.

Chaumette, F., S. Boukir, P. Bouthemy, and D. Juvin. 1996. Structure from controlled motion. *IEEE Trans. Pattern Anal. Mach. Intell.* 18(5): 492–504.

Chaumette, F. and S. Hutchinson. 2006. Visual servo control, part I: Basic approaches. *IEEE Robot. Autom. Mag.* 13(4): 82–90.

Comport, A. I., E. Marchand, and F. Chaumette. 2004. Robust model-based tracking for robot vision. *Proceedings of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems* 1: 692–697, Sendai, Japan.

Corke, P. and S. Hutchinson. 2001. A new partitioned approach to image-based visual servo control. *IEEE Trans. Robot. Autom.* 17(4): 507–515.

Espiau, B., F. Chaumette, and P. Rives. 1992. A new approach to visual servoing in robotics. *IEEE Trans. Robot. Autom.* 8(3): 313–326.

Favaro, P. and S. Soatto. 2003. Seeing beyond occlusions (and other marvels of a finite lens aperture). *Proceedings of IEEE Comp. Soc. Conf. on Computer Vision and Pattern Recognition* 2: II-579 – II-586, Saint Louis, USA.

Fleury, S. and M. Herrb. 2001. $G^{\mathrm{en}}{}_{\mathrm{o}}M$*: User Manual*. LAAS-CNRS.

Folio, D. 2007. *Stratégies de commande référencées multi-capteurs et gestion de la perte du signal visuel pour la navigation d'un robot mobile*. Ph. D. thesis, Université Paul Sabatier, Toulouse, France.

Folio, D. and V. Cadenat. 2005a. A controller to avoid both occlusions and obstacles during a vision-based navigation task in a cluttered environment. *Proceedings of European Control Conference*: 3898–3903 , Seville, Spain.

Folio, D. and V. Cadenat. 2005b. Using redundancy to avoid simultaneously occlusions and collisions while performing a vision-based task amidst obstacles. *Proceedings of European Conf. on Mobile Robots*, Ancona, Italy.

Folio, D. and V. Cadenat. 2007a. A new controller to perform safe vision-based navigation tasks amidst possibly occluding obstacles. *Proceedings of European Control Conference*, Kos, Greece.

Folio, D. and V. Cadenat. 2007b. Using simple numerical schemes to compute visual features whenever unavailable. *Proceedings of Int. Conf. on Informatics in Control Automation and Robotics*, Angers, France.

Garcia-Aracil, N., E. Malis, R. Aracil-Santonja, and C. Perez-Vidal. 2005. Continuous visual servoing despite the changes of visibility in image features. *IEEE Trans. Robot. Autom.* 21(6): 1214–1220.

Hutchinson, S., G. Hager, and P. Corke. 1996. A tutorial on visual servo control. *IEEE Trans. Robot. Autom.* 12(5): 651–670.

Jackson, J., A. Yezzi, and S. Soatto. 2004. Tracking deformable objects under severe occlusions. *Proceedings of IEEE Int. Conf. on Decision and Control* 3: 2990–2995, Atlanta, USA.

Jerian, C. and R. Jain. 1991. Structure from motion: A critical analysis of methods. *IEEE Trans. Syst., Man, Cybern.* 21(3): 572–588.

Kyrki, V., D. Kragic, and H. Christensen. 2004. New shortest-path approaches to visual servoing. *Proceedings of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems* 1: 349–355.

Lepetit, V. and P. Fua. 2006. Monocular model-based 3d tracking of rigid objects. *Found. Trends. Comput. Graph. Vis* 1(1): 1–89.

Malis, E., F. Chaumette, and S. Boudet. 1999. 2 1/2d visual servoing. *IEEE Trans. Robot. Autom.* 15(2): 238–250.

Mansard, N. and F. Chaumette. 2005. A new redundancy formalism for avoidance in visual servoing. *Proceedings of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems* 2: 1694–1700, Edmonton, Canada.

Mansard, N. and F. Chaumette. 2007. Task sequencing for sensor-based control. *IEEE Trans. Robot. Autom.* 23(1): 60–72.

Marchand, E. and G. Hager. 1998. Dynamic sensor planning in visual servoing. *Proceedings of IEEE Int. Conf. on Robotics and Automation* 3: 1988–1993, Leuven, Belgium.

Matthies, L., T. Kanade, and R. Szeliski. 1989. Kalman filter-based algorithms for estimating depth in image sequences. *Int. J. of Computer Vision* 3(3): 209–238.

Mezouar, Y. and F. Chaumette. 2002. Avoiding self-occlusions and preserving visibility by path planning in the image. *Robotics and Autonomous Systems* 41(2): 77–87.

Murray, R., Z. Li, and S. Sastry. 1994. *A mathematical introduction to robotic manipulation*. Boca Raton, FL: CRC Press.

Oliensis, J.. 2002. Exact two-image structure from motion. *IEEE Trans. Pattern Anal. Mach. Intell.* 24(12): 1618–1633.

Pissard-Gibollet, R. and P. Rives. 1995. Applying visual servoing techniques to control a mobile hand-eye system. *Proceedings of IEEE Int. Conf. on Robotics and Automation* 1: 166–171, Nagoya, Japan.

Remazeilles, A., N. Mansard, and F. Chaumette. 2006. Qualitative visual servoing: application to the visibility constraint. *Proceedings of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*: 4297–4303, Beijing, China.

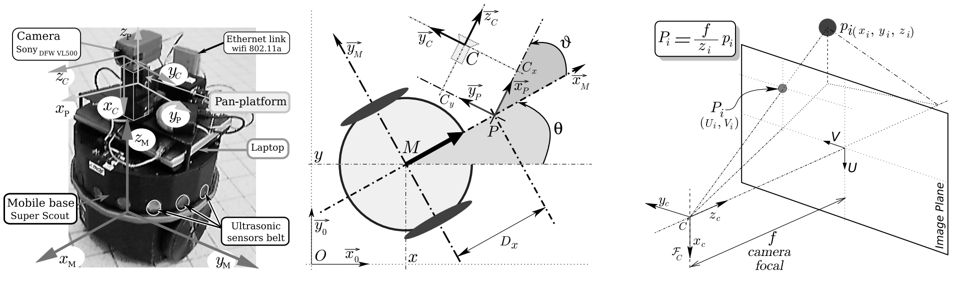
Samson, C., M. L. Borgne, and B. Espiau. 1991. *Robot Control : The task function approach* (Clarendon Press ed.). Oxford science publications.

Soatto, S. and P. Perona. 1998. Reducing ”structure from motion”: A general framework for dynamic vision part 2: Implementation and experimental assessment. *IEEE Trans. Pattern Anal. Mach. Intell.* 20(9): 943–960.

Souères, P. and V. Cadenat. 2003. Dynamical sequence of multi-sensor based tasks for mobile robots navigation. *Proceedings of Int. IFAC Symp. on Robot Control*: 423-428, Wroclaw, Poland.

Thrun, S., D. Fox, W. Burgard, and F. Dallaert. 2001. Robust mote-carlo localization for mobile robots. *Artifial Intelligence* 128(1–2): 99–141.

## List of Figures



1.a – Nomadic SuperScout II. 1.b – The robot model. 1.c – The pinhole camera model.

Figure 1: The robotic platform.

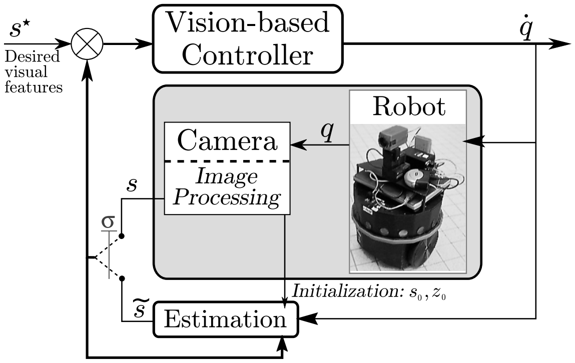
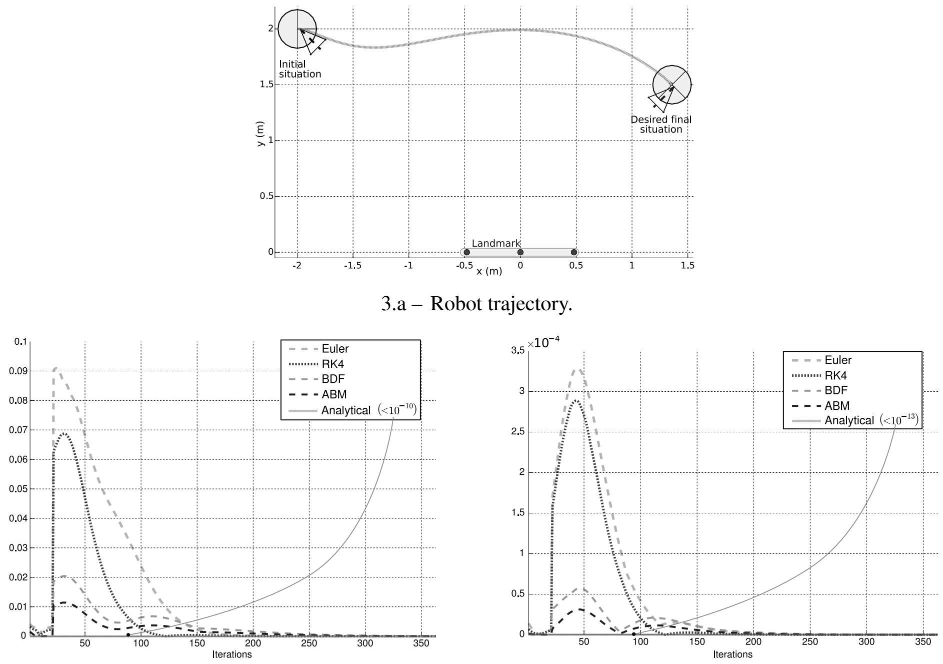


Figure 2: The control architecture.

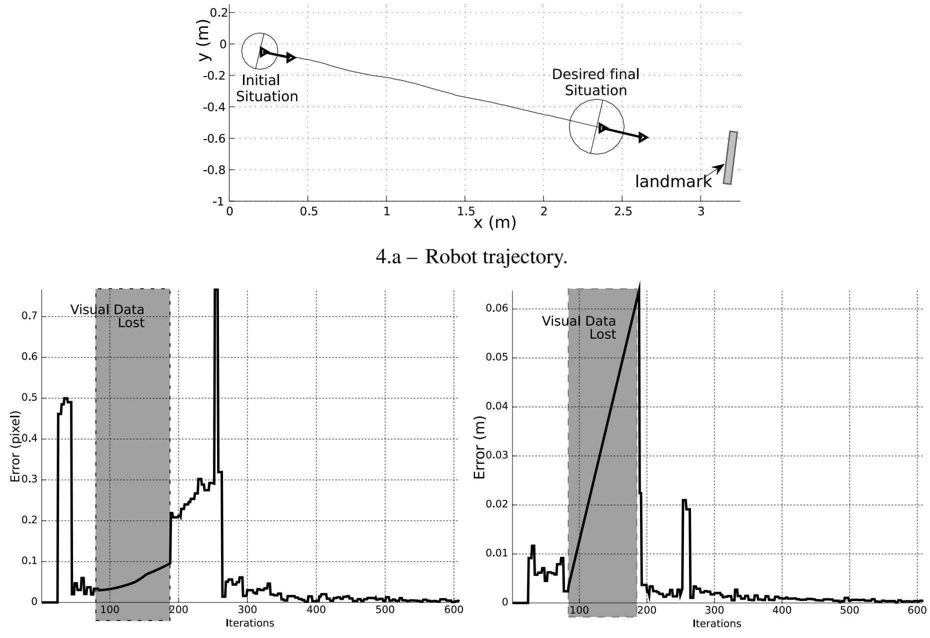


3.a – Robot trajectory.

3.b – Image data estimation errors: $\|\boldsymbol{s} - \widetilde{\boldsymbol{s}}\|$ (pixel).

3.c – z-depth estimation errors: $\|\boldsymbol{z} - \widetilde{\boldsymbol{z}}\|$ (m).

Figure 3: Visual servoing despite camera failure, simulation results.

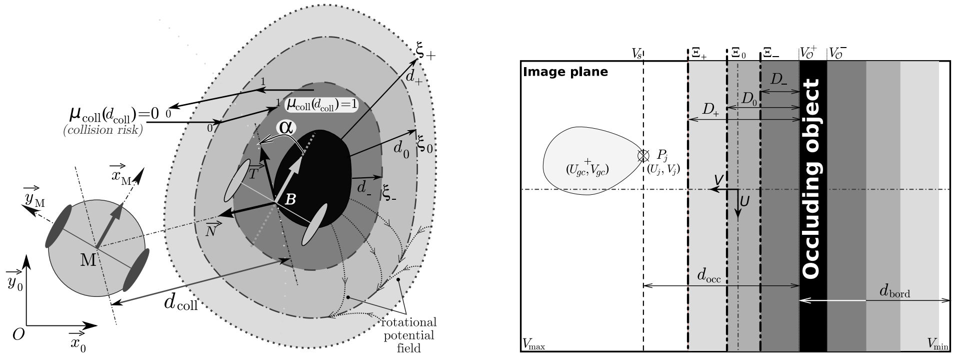


4.a – Robot trajectory.

4.b – Image data estimation errors: $\|\boldsymbol{s} - \widetilde{\boldsymbol{s}}\|$ (pixel).

4.c – z-depth estimation errors: $\|\boldsymbol{z} - \widetilde{\boldsymbol{z}}\|$ (m).

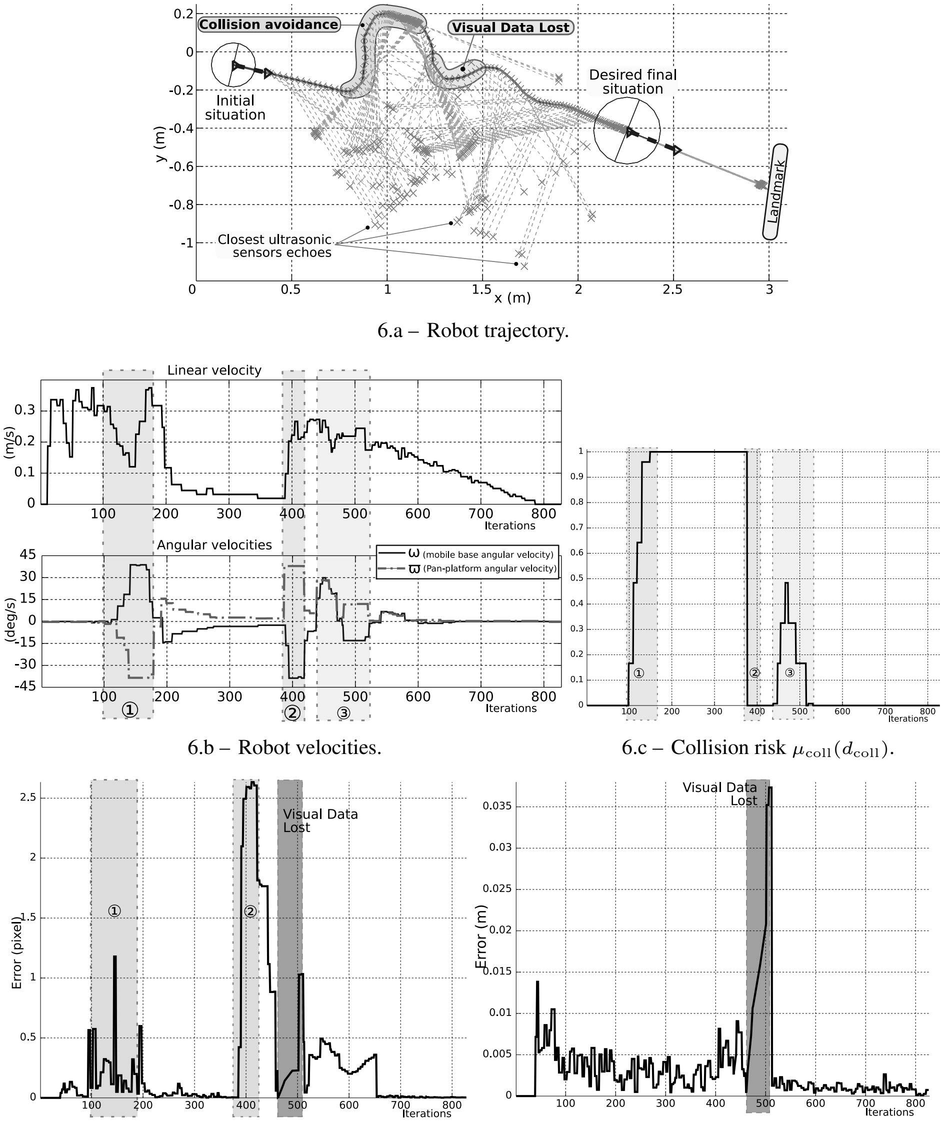
Figure 4: Visual servoing despite camera failure, experimental results.



5.a – Obstacle detection.

5.b – Occlusion detection.

Figure 5: Avoidance modeling.



6.a – Robot trajectory.

6.b – Robot velocities.

6.c – Collision risk $\mu_{\text{coll}}(d_{\text{coll}})$.

6.d – Image data estimation errors: $\|\boldsymbol{s} - \tilde{\boldsymbol{s}}\|$ (pixel).

6.e – z-depth estimation errors: $\|\boldsymbol{z} - \tilde{\boldsymbol{z}}\|$ (m).

Figure 6: Visual servoing with obstacle avoidance, experimental results.