Dealing with constraints in sensor-based robot control

Olivier Kermorgant, François Chaumette

Dealing with constraints in sensor-based robot control

Olivier Kermorgant, François Chaumette

***Abstract*—A framework is presented in this paper for the control of a multi-sensor robot under several constraints. In this approach, the features coming from several sensors are treated as a single feature vector. The core of our approach is a weighting matrix that balances the contribution of each feature, allowing to take constraints into account. The constraints are considered as additional features that are smoothly injected in the control law. Multi-sensor modeling is introduced for the design of the control law, drawing similarities with linear quadratic control. The main properties are exposed and we propose several strategies to cope with the main drawbacks. The framework is validated on a complex experiment, illustrating various aspects of the approach. The goal is the positioning of a 6 DOF robot arm with 3D visual servoing. The considered constraints are both eye-in-hand and eye-to-hand visibility, together with joint limits avoidance. The system is thus highly overdetermined, yet the task can be performed while ensuring several combination of constraints.**

***Index Terms*—Sensor-based control, sensor fusion, visual servoing, visibility constraint, joint limits avoidance**

I. Introduction

NAVIGATION or manipulation tasks are often subject to several constraints. They can be inherent to the controlled system (joint limits, limited velocity), related to the sensors (visibility constraint) or coming from the environment (obstacles). In this perspective, the goal is thus to perform the desired task while ensuring the constraints.

A popular approach in this field is path planning. The potential field method [25], [13] is a common technique to generate collision-free trajectories. This method has been applied to visual servoing in [35], where the trajectory is planned in the image space and allows ensuring the visibility and the joint limits constraints. Predictive control has also been used in visual servoing [1]. In this case, the whole trajectory is not planned but the objective function takes into account the prediction over a finite horizon. Path planning in sensor space has also been designed through LMI optimization [6], [10]. The main drawback of such schemes is that they require a model of the environment, and may not cope with unexpected obstacles.

On the other hand, reactive schemes such as sensor-based control have been used to cope with the constraints. They are often less complex to design than path planning schemes, and require less knowledge of the environment. The task function approach [39] is a popular technique to build sensor-based control laws. When dealing with several sensors, each sensor signal is given a reference signal and considered as an independent component of the global task function. Each sensor thus corresponds to a particular task. A classical scheme, often named gradient projection method (GPM), is to draw a hierarchy between the different tasks and to build a control scheme that prevents lower subtasks to disturb higher ones [17]. This is a classical way to combine sensor-based tasks and constraints such as joint limits avoidance in redundant systems [45], [30]. However, a common issue is when upper tasks constrain all the robot degrees of freedom (DOF), preventing lower subtasks from being performed. A solution can be to build a new operator that projects a subtask on to the norm of the main tasks [34], freeing some DOF that can then be used by secondary tasks. Task sequencing techniques [29] can also be used to make the task hierarchy dynamic.

With another formulation, sensor-based control laws can be designed without imposing a strict hierarchy between the tasks. Here the data coming from different sensors are treated as a unique, higher-dimensional signal. This is the approach chosen in [28] to fuse two cameras, and a force sensor and a camera, where the designed control law is equivalent to a weighted sum of the subtask control laws. In the general case, using several sensors raises the question of balancing their contributions in the control law during the servoing. Optimal methods such as Linear Quadratic (LQ) control [37], [36] can be applied in this approach, however the balance is often tuned by hand after several trials [44]. As we will see, our approach shares a similar formulation but avoids the manual tuning of the weights.

More recently, several schemes have been designed with a weighting at the level of the features: in [9] it allows addressing the problem of outliers in robust visual servoing, while in [5] it defines a task in terms of a desired region instead of a desired position. In [15], the visual features are deactivated in the case of visibility lost. Recently, the framework of varying-feature-set [31] has unified these approaches, with an emphasis on the continuity of the control law in the case of Jacobian rank change while signal components are added or removed from the control law. Yet, all these schemes were initially designed for only one sensor and to cope with specific issues in visual servoing. In this paper this framework is naturally extended to the multi-sensor case. Recent methods have been proposed to perform a sensor-based task under several unilateral constraints, with GPM framework [11], [30] or cascade of quadratic programs [19]. We will show that our non-hierarchical control law ensures several constraints while performing a multi-sensor task. In the presented paper, there is no concept of priority between the different tasks: only the global error is taken into account. This allows defining a real

This work was presented in part at ICRA'11 and IROS'11.
The authors are with Inria Rennes-Bretagne Atlantique, Rennes, France. Olivier Kermorgant is now affiliated to ICube, University of Strasbourg, France. kermorgant@unistra.fr, francois.chaumette@irisa.fr

multi-sensor task that is performed in all sensor spaces at the same time, as presented in [24] in the case of multi-camera visual servoing.

The main contribution of this paper is to propose a canonical weighting at the level of the features with an automatic computation of the weights. It allows avoiding any difficult and cumbersome manual tuning. Instead of balancing between the tasks, a multi-sensor task is defined, then the features themselves are balanced with a weighting function that takes into account the several sensors and constraints. As we will see, balancing at the level of the features allows focusing on the most critical constraints, which is not the case if all the constraints are considered as a single task and share the same weight. This approach does not require any hierarchy between the tasks and show nice properties in the sensors space and in the robot behavior.

The proposed approach is a generic framework that embeds our previous works about multi-camera visual servoing [24], robot positioning while ensuring the visibility constraint [23] and avoiding the joint limits [22]. In this paper, all these issues are addressed within an homogeneous framework. As we will show, this allows regrouping very easily all the tasks and constraints into a single experiment. The robot can thus perform eye-in-hand/eye-to-hand cooperation, together with joint limits avoidance while ensuring the visibility constraint in both images. As far as we know, this is the first time such a complete and complex configuration is considered.

The paper is organized as follows. The general modeling of a multi-sensor robot is presented in Section II. We also show how the proposed weighting of the signal error can take unilateral constraints into account. Then, the control law, its stability analysis and its main properties are described in Section III. Several additional strategies are presented in Section IV for specific issues that may occur in practice. Finally, experimental results are presented in Section V.

II. Multi-sensor modeling

This section presents the general modeling of a multi-sensor robot. First, we define the global kinematic model, then we introduce the weighted signal error that will be used in the control law. We propose a generic weighting function that allows both balancing the sensor features and taking into account unilateral constraints.

A. Kinematic model

We consider a robotic system equipped with k sensors providing data about the robot pose in its environment. The robot joint positions are denoted $\mathbf{q}$ and we define $n = \dim(\mathbf{q})$. Each sensor S_i delivers a signal $\mathbf{s}_i$ of dimension m_i with $\sum_{i=1}^{k} m_i = m$ and we assume $m \geq n$. A signal component is called a sensor feature. In the case of a motionless environment, the signal time derivative is directly related to the sensor velocity $\mathbf{v}_i$ expressed in the sensor frame by:

$$\dot{\mathbf{s}}_i = \mathbf{L}_i \mathbf{v}_i \tag{1}$$

where $\mathbf{L}_i$ is named the interaction matrix of $\mathbf{s}_i$ [39], [4] and is of dimension $(m_i \times 6)$. Its analytical form can be derived for many features coming from exteroceptive sensors. It depends

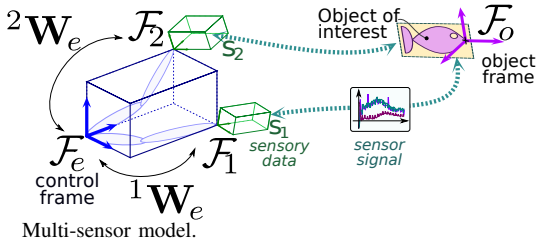


Fig. 1. Multi-sensor model.

mainly on the type of considered sensory data $\mathbf{s}$ and on the sensor intrinsic parameters. $\mathbf{L}_i$ may also depend on other data: for instance the interaction matrix of an image point observed by a camera depends on the depth of that point, which is not actually measured in the image [4].

Now, we consider a reference frame $\mathcal{F}_e$ in which the robot velocity can be controlled. This frame can be for instance the end-effector frame for a robot arm as shown in Fig. 1. The screw transformation matrix allows expressing the sensor velocity $\mathbf{v}_i$ wrt. the robot velocity $\mathbf{v}_e$:

$$\mathbf{v}_i = {}^i\mathbf{W}_e \mathbf{v}_e \tag{2}$$

${}^i\mathbf{W}_e$ is given by [38]:

$${}^i\mathbf{W}_e = \left[\begin{array}{c|c} {}^i\mathbf{R}_e & \left[{}^i\mathbf{t}_e\right]_\times {}^i\mathbf{R}_e \\ \hline \mathbf{0}_{3\times 3} & {}^i\mathbf{R}_e \end{array}\right] \tag{3}$$

where ${}^i\mathbf{R}_e \in SO(3)$ and ${}^i\mathbf{t}_e \in \mathbb{R}^3$ are respectively the rotation matrix and the translation vector between $\mathcal{F}_e$ and $\mathcal{F}_{\mathbf{s}i}$. $\left[{}^i\mathbf{t}_e\right]_\times$ is the (3×3) skew-symmetric matrix related to ${}^i\mathbf{t}_e$. Denoting ${}^e\mathbf{J}_q \in \mathbb{R}^{m\times n}$ the robot Jacobian, we have:

$$\dot{\mathbf{s}}_i = \mathbf{L}_i {}^i\mathbf{W}_e {}^e\mathbf{J}_q \dot{\mathbf{q}} \tag{4}$$

Denoting $\mathbf{s} = (\mathbf{s}_1, \ldots, \mathbf{s}_k)$ the m-dimensional signal of the multi-sensor set, (4) allows linking the signal time variation with the joint velocity:

$$\dot{\mathbf{s}} = \mathbf{J}_s \dot{\mathbf{q}} \tag{5}$$

with:

$$\mathbf{J}_s = \mathbf{L}\mathbf{W}_e {}^e\mathbf{J}_q = \begin{bmatrix} \mathbf{L}_1 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & \mathbf{L}_k \end{bmatrix} \begin{bmatrix} {}^1\mathbf{W}_e \\ \vdots \\ {}^k\mathbf{W}_e \end{bmatrix} {}^e\mathbf{J}_q \tag{6}$$

where $\mathbf{L} \in \mathbb{R}^{m\times 6k}$ contains the interaction matrices of the sensors and $\mathbf{W}_e \in \mathbb{R}^{6k\times 6}$ contains the transformation matrices. In the sequel we assume $\mathbf{J}_s$ is of full rank n. We will mention in Section III-B1 that this assumption could be relaxed, but this article focuses on the full rank case. We now define the weighted error that will be used in the control law.

B. Weighted error

The goal of sensor-based control is to design a control law that makes the robot reach a desired value $\mathbf{s}^*$ of the sensor features. This desired value may be obtained by teaching-by-showing, or through a model at the desired pose: for example in [32], visual servoing is performed with the desired value being the projection of the object model at the desired camera pose.

1) Weighted error: We define the weighted multi-sensor signal error as:

$$\mathbf{e}_\mathbf{H} = \mathbf{H}\mathbf{e} \tag{7}$$

where $\mathbf{e}$ is the sensor error defined as $\mathbf{e} = \mathbf{s} - \mathbf{s}^*$ and $\mathbf{H}$ is a diagonal positive semi-definite weighting matrix that depends on the current value of $\mathbf{s}$. As in all varying-feature-set schemes [31], each component h_i of $\mathbf{H}$ may vary in order to ensure specific constraints, manage priorities or add or remove a sensor or a feature from the control law. In the case of k sensors, $\mathbf{H}$ yields:

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \mathbf{H}_k \end{bmatrix} \tag{8}$$

where $\mathbf{H}_i$ is the weighting matrix for sensor S_i.

2) Weighting canonical form: Weighting can be performed for several purposes. First, the most simple goal is to balance the disparate sensor contributions during the scheme. As in Linear Quadratic (LQ) control, this amounts to optimizing the system behavior by defining a specific weight for each sensor feature. In this paper, we propose to also use the weight of a sensor feature to take into account unilateral constraints on that feature. We thus define a generic weighting by:

$$\forall i \in [1, m] : \; h_i = h_i^t + h_i^c \tag{9}$$

where h_i^t is tuned for the general balance of the feature and where h_i^c allows taking potential constraints into account. Classical control laws such as visual servo schemes usually use the simplest weighting that corresponds to $\mathbf{H} = \mathbb{I}_m$, that is:

$$\forall i, h_i^t = 1 \;\text{ and }\; h_i^c = 0 \tag{10}$$

Each weight $(h_i^t)_i$ may also be tuned independently, as in LQ control. In practice, several trials are often necessary to determine the best weighting [44]. In varying-feature-set schemes [15], [9], the weights h_i^t vary between 0 and 1 depending on the confidence in each sensor feature. In this paper we do not focus on the tuning of this weight term, and we set $h_i^t = 1$ if the feature s_i is always used for the actual navigation task, and $h_i^t = 0$ if the feature s_i only corresponds to a constraint to be ensured. We now explicit the generic formulation for the term h_i^c handling the constraints.

A constraint is usually expressed by an inequality on the value of a sensor feature. This is typically the case for joint limits or image visibility, and is also valid when range sensors are measuring the distance to the obstacles. Singularity avoidance can also be considered by setting a lower bound for $\det(\mathbf{J}_s^\top \mathbf{J}_s)$. In all cases, this corresponds to having to keep the feature value s_i in an interval $[s_i^-, s_i^+]$. In that case, a safe interval $[s_i^{s-}, s_i^{s+}]$ can be defined by:

$$\begin{cases} s_i^{s-} = s_i^- + \rho_i (s_i^+ - s_i^-) \\ s_i^{s+} = s_i^+ - \rho_i (s_i^+ - s_i^-) \end{cases} \tag{11}$$

where $\rho_i \in [0, 0.5]$ is a tuning parameter. The weighting term h_i^c handling the constraint is then given by:

$$h_i^c = \begin{cases} \frac{s_i - s_i^{s+}}{s_i^+ - s_i} & \text{if } s_i > s_i^{s+} \\ \frac{s_i^{s-} - s_i}{s_i - s_i^-} & \text{if } s_i < s_i^{s-} \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

h_i^c is represented in Fig. 2. Similarly to a repulsive field [25], the weight is null in the safe region and continuously increases to ∞ as the feature approaches the limit. A constraint is said to be active when its weight h_i^c is non-null. The next section presents the control law and its main properties. In particular, the weights in case of one or several active constraints are studied in Section III-C5.

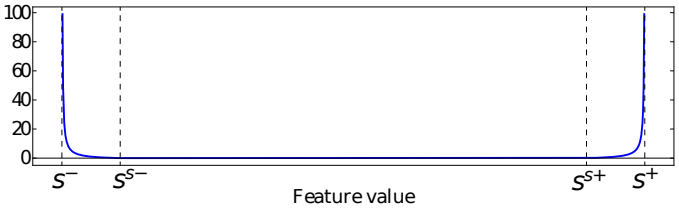


Fig. 2. Generic weighting h^c for the basic constraints

III. Control law

We now present the generic control law associated with the weighted error defined in (7) and its link with the linear quadratic (LQ) approach. The main properties for sensor fusion are then presented. In particular, when only one constraint is considered we show in Section III-C that a sufficient weight ensures the corresponding feature error is decreasing, and that a minimal weight can be determined. We then expose additional strategies that can be used for specific issues.

A. Weighted control scheme

In the task function approach [39], the task error $\mathbf{e}_{\text{task}} \in \mathbb{R}^n$ is defined by:

$$\mathbf{e}_{\text{task}} = \mathbf{C}(\mathbf{s} - \mathbf{s}^*) = \mathbf{C}\mathbf{e} \tag{13}$$

where $\mathbf{C} \in \mathbb{R}^{n \times m}$ is named the combination matrix and allows to take into account the redundancy between the sensor features. A classical controller is then:

$$\dot{\mathbf{q}} = -\lambda \mathbf{e}_{\text{task}} = -\lambda \mathbf{C}\mathbf{e} \tag{14}$$

A popular choice that tries to ensure an exponential decrease of $\mathbf{e}_{\text{task}}$ is $\mathbf{C} = \widehat{\mathbf{J}}_s^{\,+}$, that is an estimation of the Moore-Penrose pseudo-inverse of $\mathbf{J}_s$. In our case, $\mathbf{J}_s^+ = (\mathbf{J}_s^\top \mathbf{J}_s)^{-1} \mathbf{J}_s^\top$ since $\mathbf{J}_s$ is full rank. This strategy can be seen as a particular case of LQ control [36]. In this framework, a cost function F has to be minimized and is defined with:

$$F = (\mathbf{s} - \mathbf{s}^*)^\top \mathbf{Q} (\mathbf{s} - \mathbf{s}^*) + \dot{\mathbf{q}}^\top \mathbf{R} \dot{\mathbf{q}} \tag{15}$$

where $\mathbf{Q}$ and $\mathbf{R}$ are weighting matrices that are usually tuned in order to obtain an optimal behavior of the robot. The selection of the elements of $\mathbf{Q}$ and $\mathbf{R}$ may be computed from a pole placement tuning or considerations on the variance of observed data [44]. In practice, several trials are often necessary to obtain the desired behavior. The corresponding control input yields [26]:

$$\dot{\mathbf{q}} = -\lambda (\widehat{\mathbf{J}}_s^\top \mathbf{Q} \widehat{\mathbf{J}}_s + \mathbf{R})^{-1} \widehat{\mathbf{J}}_s^\top \mathbf{Q} (\mathbf{s} - \mathbf{s}^*) \tag{16}$$

This control law is the same as (14) for the particular weighting $\mathbf{Q} = \mathbb{I}_m$ and $\mathbf{R} = 0$.

When considering the weighted error $\mathbf{e}_\mathbf{H}$ instead of $\mathbf{e}$, the associated Jacobian is $\mathbf{J}_\mathbf{H} = \mathbf{H}\mathbf{J}_s$. In this case, control law (14) yields:

$$\dot{\mathbf{q}} = -\lambda (\mathbf{H}\widehat{\mathbf{J}}_s)^+ \mathbf{e}_\mathbf{H} = -\lambda (\mathbf{H}\widehat{\mathbf{J}}_s)^+ \mathbf{H}\mathbf{e} \tag{17}$$

The combination matrix of $\mathbf{e}$ is thus given by:

$$\mathbf{C} = (\mathbf{H}\widehat{\mathbf{J}}_s)^+\mathbf{H} \tag{18}$$

When compared with LQ control, this combination matrix corresponds to the particular weighting $\mathbf{Q} = \mathbf{H}^2$ and $\mathbf{R} = 0$. Indeed, in this case the LQ scheme (16) yields:

$$\dot{\mathbf{q}} = -\lambda(\widehat{\mathbf{J}}_s^\top \mathbf{H}^2 \widehat{\mathbf{J}}_s)^{-1}\widehat{\mathbf{J}}_s^\top \mathbf{H}^2(\mathbf{s}-\mathbf{s}^*) \tag{19}$$

$$= -\lambda\left((\mathbf{H}\widehat{\mathbf{J}}_s)^\top(\mathbf{H}\widehat{\mathbf{J}}_s)\right)^{-1}(\mathbf{H}\widehat{\mathbf{J}}_s)^\top\mathbf{H}(\mathbf{s}-\mathbf{s}^*) \tag{20}$$

$$= -\lambda(\mathbf{H}\widehat{\mathbf{J}}_s)^+\mathbf{H}(\mathbf{s}-\mathbf{s}^*) = -\lambda\mathbf{C}(\mathbf{s}-\mathbf{s}^*) \tag{21}$$

The main difference between our scheme and classical LQ control is about the use of the weighting matrix $\mathbf{Q}$. In LQ control it is usually tuned in order to obtain an optimal behavior of the robot. In the proposed scheme, we focus on the balance between the different features and sensors and potential constraints. Actually, the two strategies may be used in a complementary way: if both $\mathbf{H}$ and $\mathbf{Q}$ are defined from their respective frameworks, a global scheme can be designed by using the weighting matrix $\mathbf{H}^\top\mathbf{Q}\mathbf{H}$. Also, a control cost matrix $\mathbf{R} \neq 0$ could be used if needed.

As for the estimation of $\mathbf{J}_s$, the most popular choices are summarized in [24], showing the induced behaviors. From (6), computing $\mathbf{J}_s$ amounts to choosing how to estimate the interaction matrices $\mathbf{L}_i$ and the transformation matrices ${}^i\mathbf{W}_e$.

a) Interaction matrices: Several possibilities exist for the interaction matrices [4]. Two classical choices are to use the current interaction matrix, or its value at the desired pose $\mathbf{L}^*$. In this case the interaction matrix is constant. Another popular strategy is the mean interaction matrix $1/2(\mathbf{L}+\mathbf{L}^*)$, which was recently shown as an approximation of second-order minimization [41].

b) Transformation matrices: If the sensors are rigidly attached to the effector, then all transformation matrices are constant and can usually be estimated in an offline calibration step. In the other case, for instance in eye-to-hand configuration, $\mathbf{W}_e$ is not constant and the desired value $\mathbf{W}_e^*$ depends on the final 3D pose of the sensors wrt. the effector. This pose is generally unknown in sensor-based control. The most plausible choice is thus to estimate the current transformation matrices from the robot geometrical model and calibration.

In the sequel we assume that an estimation of the current matrices $\mathbf{L}$ and $\mathbf{W}_e$ is available, allowing to estimate $\mathbf{J}_s$ in real-time. We now study the properties of the proposed scheme.

B. Control scheme properties

This section explores the basic properties of the control law. First we expose the condition for control law continuity and study the case of null weights. We then show local asymptotic stability.

1) Continuity and influence of null weights: The continuity of varying-feature-set control laws has been studied in [31]. In the general case, continuity is ensured under three conditions: $\mathbf{H}$ and $\mathbf{J}_s$ are continuous and the pseudo-inverse operator is continuous for $\mathbf{H}\mathbf{J}_s$. The latter is ensured under the assumption that $\mathbf{H}\mathbf{J}_s$ is full-rank, which implies in particular that there are always at least n non-null weights. The case of rank change is solved in [31] with a generalized pseudo-inverse, however in this paper we use the classical pseudo-inverse and assume $\mathbf{H}\mathbf{J}_s$ is full-rank. Usual sensor features have a continuous Jacobian $\mathbf{J}_s$. The formulation of the weighting matrix in Section II-B2 is also continuous, hence the control law is continuous.

Control law (17) is designed to ensure that $\mathbf{H}\dot{\mathbf{e}} = -\lambda\mathbf{H}\mathbf{e}$, which is different from classical design $\dot{\mathbf{e}} = -\lambda\mathbf{e}$. This difference clearly appears for configurations with null weights. Assuming $\mathbf{s} = (\mathbf{s}_1, \mathbf{s}_0)$ where features $\mathbf{s}_0$ have null weights ($\mathbf{H}_0 = 0$), the control law (17) can be written:

$$\dot{\mathbf{q}} = -\lambda\begin{bmatrix}\mathbf{H}_1\widehat{\mathbf{J}}_1\\ \mathbf{H}_0\widehat{\mathbf{J}}_0\end{bmatrix}^+\begin{bmatrix}\mathbf{H}_1\mathbf{e}_1\\ \mathbf{H}_0\mathbf{e}_0\end{bmatrix} = -\lambda\begin{bmatrix}\mathbf{H}_1\widehat{\mathbf{J}}_1\\ \mathbf{0}\end{bmatrix}^+\begin{bmatrix}\mathbf{H}_1\mathbf{e}_1\\ \mathbf{0}\end{bmatrix}$$

$$= -\lambda\begin{bmatrix}(\mathbf{H}_1\widehat{\mathbf{J}}_1)^+ & \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{H}_1\mathbf{e}_1\\ \mathbf{0}\end{bmatrix} = -\lambda(\mathbf{H}_1\widehat{\mathbf{J}}_1)^+\mathbf{H}_1\mathbf{e}_1 \tag{22}$$

The scheme is thus equivalent to the control law for active features only. In particular, it is different from not taking into account the weighting matrix in the pseudo-inverse. More precisely, in [16] the combination matrix is defined as $\mathbf{C} = \widehat{\mathbf{J}}_s^+\mathbf{H}$, inducing the following:

$$\dot{\mathbf{q}} = -\lambda\widehat{\mathbf{J}}_s^+\mathbf{H}\mathbf{e} \tag{23}$$

$$= -\lambda\begin{bmatrix}\widehat{\mathbf{J}}_1\\ \widehat{\mathbf{J}}_0\end{bmatrix}^+\begin{bmatrix}\mathbf{H}_1\mathbf{e}_1\\ \mathbf{0}\end{bmatrix} \tag{24}$$

The zeroed error components are thus still taken into account and the system behaves exactly as if the desired values for $\mathbf{e}_0$ had been reached, which induces an undesired conservative behavior to ensure the useless constraints $\mathbf{e}_0 = 0$. This is not the case with our approach.

2) Local asymptotic stability: Varying-feature-set schemes usually neglect the time variation of $\mathbf{H}$ by assuming the weighting matrix is varying slowly, or that it is null at the convergence as in region-reaching visual servoing [5]. Actually, when $\mathbf{H}$ is integrated into the combination matrix and assumed to be varying wrt. $\mathbf{s}$, the stability analysis is the same as with a varying $\mathbf{J}_s$ [4]. As for classical IBVS schemes, a direct consequence is that global asymptotic stability cannot be proven as soon as redundant features are involved ($m > n$). From (6) and (18), the task error variation yields:

$$\dot{\mathbf{e}}_{\text{task}} = \mathbf{C}\dot{\mathbf{e}} + \dot{\mathbf{C}}\mathbf{e} = (\mathbf{C}\mathbf{J}_s + \mathbf{O})\dot{\mathbf{q}}$$

$$= -\lambda(\mathbf{C}\mathbf{J}_s + \mathbf{O})\mathbf{e}_{\text{task}} \tag{25}$$

where $\mathbf{O} \in \mathbb{R}^{n\times n} = 0$ when $\mathbf{e}_{\text{task}} = 0$ [4]. With the combination matrix from (18), this scheme is known to be locally asymptotically stable (LAS) in a neighborhood of $\mathbf{e} = 0$ if [18]:

$$\mathbf{C}\mathbf{J}_s = (\mathbf{H}\widehat{\mathbf{J}}_s)^+\mathbf{H}\mathbf{J}_s > 0 \tag{26}$$

The system is thus LAS when $\mathbf{H}\mathbf{J}_s$ and $\mathbf{H}\widehat{\mathbf{J}}_s$ are full rank and when the Jacobian $\mathbf{J}_s$ is sufficiently well estimated, which is the case in general. In this case, potential local minima correspond to configurations where $\mathbf{H}^2(\mathbf{s}-\mathbf{s}^*) \in \text{Ker}\ \widehat{\mathbf{J}}_s^\top$. We will see in Section IV-A how to deal with this issue. Let us also note that determining theoretically the convergence domain seems to be out of reach. However, as we will see in Section V, it reveals to be surprisingly large in practice.

C. Particular case of one constraint

In this section we focus on the case where only one active constraint is involved. In that case, we show that a sufficiently high weight induces the decreasing of the corresponding feature error. In particular, we determine the minimal weight ensuring the corresponding constraint is respected. Dealing with several active constraints simultaneously is finally discussed at the end of this section.

1) Sufficient weight: We assume the reference value s_i^* of the feature s_i is in the confidence interval. A sufficient condition for the associated constraint to be ensured is that the error $e_i = s_i - s_i^*$ decreases. We now show that this can be ensured at each iteration if the associated weight is high enough. A classical Lyapunov function associated with the weighted error is $V(\mathbf{e_H}) = \frac{1}{2}\mathbf{e_H}^\top \mathbf{e_H}$. Assuming we are in the domain of local stability, the time derivative of V yields:

$$\dot{V} = \frac{\partial V}{\partial \mathbf{e_H}} \dot{\mathbf{e}}_\mathbf{H} = \sum_{i=1}^{m} h_i^2 e_i \dot{e}_i < 0 \tag{27}$$

The error e_i decreases *iif* $e_i\dot{e}_i < 0$, which is equivalent to:

$$h_i^2 > -\frac{1}{e_i\dot{e}_i} \sum_{j\neq i} h_j^2 e_j \dot{e}_j \tag{28}$$

Hence, in any configuration there exists a sufficiently high weight h_i that ensures the corresponding feature error norm is decreasing. Note that if $\dot{e}_i = 0$ or $e_i = 0$, the corresponding constraint is de facto ensured.

This property on sufficient weights has been recently highlighted in [21], where the parallel is drawn with the GPM approach and constrained optimization.

Isolating the particular feature s_i , the control law (17) can be written as the minimum-norm solution to:

$$\min_{\dot{\mathbf{q}}} \|\mathbf{H}_{\neq i}\mathbf{J}_{\neq i}\dot{\mathbf{q}} - \mathbf{H}_{\neq i}\dot{\mathbf{e}}^*_{\neq i}\|^2 + h_i \|\mathbf{J}_i\dot{\mathbf{q}} - \dot{\mathbf{e}}_i^*\|^2 \tag{29}$$

where $\dot{\mathbf{e}}^*$ is such that $\mathbf{H}\dot{\mathbf{e}}^* = -\lambda\mathbf{H}\mathbf{e}$ and $X_{\neq i}$ denotes a value related to all features except s_i. It has been shown in [42] that the solution to (29) when h_i tends to infinity is exactly the solution to the constrained minimization:

$$\begin{cases} \min_{\dot{\mathbf{q}}} \|\mathbf{H}_{\neq i}\mathbf{J}_{\neq i}\dot{\mathbf{q}} - \mathbf{H}_{\neq i}\dot{\mathbf{e}}^*_{\neq i}\|^2 \\ \text{s.t. } \mathbf{J}_i\dot{\mathbf{q}} = \dot{\mathbf{e}}_i^* \end{cases} \tag{30}$$

Note that (30) corresponds to the GPM approach with feature s_i used as the priority task and the other features as the secondary task. At a given iteration and if the Jacobian $\mathbf{J}_i$ is sufficiently well estimated, the condition $\dot{e}_i e_i < 0$ is ensured with the system (30) since in this case, $\dot{e}_i e_i \approx \dot{e}_i^* e_i = -\lambda e_i^2 < 0$. Hence, coming back to (29), there exists a value $h_i^{\min}$ such that:

$$\forall h_i > h_i^{\min}, \dot{e}_i e_i < 0 \tag{31}$$

The decrease of the error, hence the corresponding constraint, can be thus ensured with a finite weight at any given iteration. We now explicit the computation of this minimal weight.

2) Minimal weight: We denote s_i the feature corresponding to the considered constraint. The goal here is to disturb the task $\mathbf{e}_{\neq i}$ as little as possible by determining the weight h_i that is as small as possible, yet sufficiently high to ensure the corresponding constraint.

The time variation of the constrained feature is given by:

$$\dot{s}_i = \mathbf{J}_i\dot{\mathbf{q}} = -\lambda \mathbf{J}_i(\mathbf{H}\mathbf{J}_s)^+\mathbf{H}(\mathbf{s}-\mathbf{s}^*) \tag{32}$$

$$= -\lambda \mathbf{J}_i(\mathbf{J}_s^\top\mathbf{H}^\top\mathbf{H}\mathbf{J}_s)^{-1}(\mathbf{J}_{\neq i}^\top\mathbf{H}_{\neq i}^2\mathbf{e}_{\neq i} + h_i^2\mathbf{J}_i^\top e_i) \tag{33}$$

We now show that ensuring $\dot{s}_i = 0$ leads to a linear condition on h_i^2.

We have:

$$(\mathbf{J}_s^\top\mathbf{H}^\top\mathbf{H}\mathbf{J}_s)^{-1} = \frac{\text{adj}(\mathbf{J}_s^\top\mathbf{H}^2\mathbf{J}_s)}{\det(\mathbf{J}_s^\top\mathbf{H}^2\mathbf{J}_s)} = \frac{1}{D(\mathbf{h})}\mathbf{A}(\mathbf{h}) \tag{34}$$

where $D(\mathbf{h})$ is a strictly positive polynomial of (h_i^2), being the determinant of a symmetric invertible matrix, and $\mathbf{A}(\mathbf{h})$ is the adjugate matrix of $\mathbf{J}_s^\top\mathbf{H}^2\mathbf{J}_s$, that is the matrix of the cofactors. $\mathbf{A}$ is a $(n\times n)$ full rank symmetric matrix. The constrained feature time variation thus yields, up to the scale factor $\frac{\lambda}{D}$:

$$\dot{s}_i \propto -\mathbf{J}_i\mathbf{A}(\mathbf{J}_{\neq i}^\top\mathbf{H}_{\neq i}^2\mathbf{e}_{\neq i} + h_i^2\mathbf{J}_i^\top e_i) \tag{35}$$

We now show that $\mathbf{J}_i\mathbf{A}$ does not depend on h_i. To do so, it is sufficient to show that $\mathbf{J}_1\mathbf{A}$ does not depend on h_1, as any permutation of the rows of $\mathbf{s}$, $\mathbf{J}$ and $\mathbf{H}$ would lead to the same control law. Let $\mathbf{Q}$ and $\mathbf{R}$ be the QR decomposition of $\mathbf{J}^\top$. We have:

$$\mathbf{J}\mathbf{A} = \mathbf{R}^\top\mathbf{Q}^\top\text{adj}(\mathbf{Q}\mathbf{R}\mathbf{H}^2\mathbf{R}^\top\mathbf{Q}^\top) \tag{36}$$

$$= \mathbf{R}^\top\mathbf{Q}^\top\text{adj}(\mathbf{Q}^\top)\text{adj}(\mathbf{R}\mathbf{H}^2\mathbf{R}^\top)\text{adj}(\mathbf{Q}) \tag{37}$$

$$= \mathbf{R}^\top\text{adj}(\mathbf{R}\mathbf{H}^2\mathbf{R}^\top)\text{adj}(\mathbf{Q}) \tag{38}$$

as $\mathbf{Q}^\top\text{adj}(\mathbf{Q}^\top) = \det(\mathbf{Q}^\top)\mathbb{I}_n = \mathbb{I}_n$. $\mathbf{R}$ being upper triangular, h_1 only appears in the first element of $\mathbf{R}\mathbf{H}^2\mathbf{R}^\top$. From the adjugate matrix properties, the first row of $\text{adj}(\mathbf{R}\mathbf{H}^2\mathbf{R}^\top)$ does not depend on h_1. Thus, as $\mathbf{R}^\top$ is lower triangular, the first row of $\mathbf{J}\mathbf{A}$ does not depend on h_1, which concludes the demonstration that can be extended to all indexes.

We denote the two h_i-independent scalars:

$$\begin{cases} c_i &= -\mathbf{J}_i\mathbf{A}\mathbf{J}_{\neq i}^\top\mathbf{H}_{\neq i}^2\mathbf{e}_{\neq i} \\ a_i &= \mathbf{J}_i\mathbf{A}\mathbf{J}_i^\top e_i \end{cases} \tag{39}$$

From (35), $\dot{s}_i$ is thus linear wrt. h_i^2 and can be written:

$$\dot{s}_i \propto c_i - h_i^2 a_i \tag{40}$$

This leads to two configurations C1 and C2 that are represented in Fig. 4:

- Approaching the constraint (C1): if c_i and a_i have the same sign, the robot is going towards the constraint. In that case there exists a positive h_i^2 such that $\dot{s}_i$ is null.
- Avoiding the constraint (C2): if c_i and a_i do not have the same sign, the robot is moving away from the constraint: self-avoidance occurs and the avoidance scheme can be ignored.

From this observation, we define the values s^{a-} and s^{a+} where the feature has to stop:

$$\begin{cases} s_i^{a-} &= s_i^- + \rho^a(s_i^+ - s_i^-) \\ s_i^{a+} &= s_i^+ - \rho^a(s_i^+ - s_i^-) \end{cases} \tag{41}$$

where $\rho^a < \rho$ is a tuning parameter. The minimal value can thus be computed analytically from (40):

$$h_i^{\min} = \begin{cases} \sqrt{\frac{c_i}{a_i}} & \text{if } \frac{c_i}{a_i} > 0 \quad \text{(C1)} \\ 0 & \text{else} \quad \text{(C2)} \end{cases} \tag{42}$$

where $h_i^{\min} = 0$ corresponds to the configurations where self-avoidance occurs.

Such minimal weight ensures that the constraint is ensured at least when $s_i = s_i^a$ since in this case we have $\dot{s}_i = 0$.

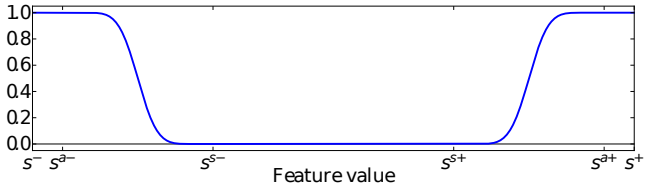


Fig. 3. Activation function for lower and upper bounds.

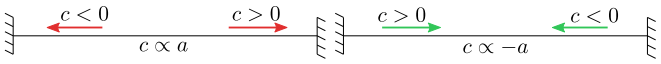
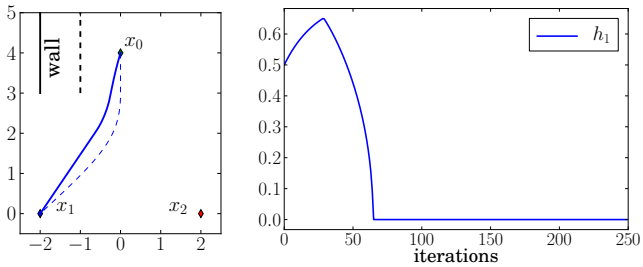


Fig. 4. Configurations C1 (left) and C2 (right). The feature approaches the nearest limit in C1, while it goes away in C2.



(a) Trajectory to x_1 (b) Weight to x_1

Fig. 5. Minimal weight in simulation. Robot trajectory (a) and corresponding weight (b). The dotted line shows the trajectory with the generic weight (12). The minimal weight value is 0.5 at the beginning, then increases as the robot approaches the wall. The minimal weight is null once the robot has passed the wall.

However, as we do not need to ensure the constraint before $s_i = s_i^a$, the minimal weight is smoothly taken into account with an injection function.

3) Injection function: We address the injection of $h^{\min}$ with the following form of the weights:

$$\forall i, h_i^c = \mu_i(s_i) h_i^{\min} \tag{43}$$

where $\mu_i(s_i) \in [0,1]$ is a continuous function.

To ensure continuity of $\mathbf{HJ}_s$ and $\mathbf{He}$, weights must be null at feature activation and deactivation, and increasing as the constrained feature values vary from the safe limit to the physical limit. In our case, the injection function is null when $s_i = s_i^s$ and equal to 1 when $s_i = s_i^a$. Such a function can be defined with a sigmoid:

$$\mu_i = \begin{cases} \frac{1}{2}\left(1+\tanh\left(\frac{1}{s_i^{\mathrm{a}+}-s_i} - \frac{1}{s_i-s_i^{\mathrm{s}+}}\right)\right) & \text{if } s_i^{\mathrm{s}+} < s_i < s_i^{\mathrm{a}+} \\ \frac{1}{2}\left(1-\tanh\left(\frac{1}{s_i^{\mathrm{a}-}-s_i} - \frac{1}{s_i-s_i^{\mathrm{s}-}}\right)\right) & \text{if } s_i^{\mathrm{a}-} < s_i < s_i^{\mathrm{s}-} \\ 1 & \text{else} \end{cases} \tag{44}$$

μ_i is $\mathcal{C}^\infty$ and smoothly increases the weight as the feature reaches the limit, with $\mu_i(s_i^{\mathrm{a}-}) = \mu_i(s_i^{\mathrm{a}+}) = 1$ and $\mu_i(s_i^{\mathrm{s}-}) = \mu_i(s_i^{\mathrm{s}+}) = 0$. The proposed injection function is represented on Fig. 3. This allows activating the feature as progressively as possible, hence with the smallest disturbance on the main task.

4) Example in simulation: The proposed minimal weight is illustrated in simulation. The simulation setup is voluntarily simple and consists in a 2D Cartesian robot that has to reach a point. The task Jacobian is thus $\mathbb{I}_2$. The constraint is to keep a minimum distance to the wall that is present. The simulation is represented in Fig. 5. The robot starts in $x_0(0,4)$. The measured distance to the wall is $d = 2$ while the desired distance has been set to $d^* = 4$. The activation values are defined as $d^a = 1$ and $d^s = 3$. Denoting $\mathbf{e}_x$ the task error and e_d the error related to the constraint, the variables from (39) yield:

$$\mathbf{J}_x = \mathbb{I}_2 \quad \mathbf{J}_d = [1\ 0] \quad \mathbf{A} = \mathbb{I}_2 \quad e_d = -2 \tag{45}$$

From (39), we thus have $a = -2$ and $c = [-1\ 0]\mathbf{e}_x$.

- If the desired position is $x_1(-2,0)$ we have $c = -2$. From (42), the minimum weight is thus $h^{\min} = 1$. As $d = 2$, the actual weight will be $h = \mu(d)h^{\min} = .5 \times h^{\min} = 0.5$. As long as $d > d^a$ the robot will thus approach the wall if the task requires so. The weight is represented in Fig. 5b. We can see the initial value is indeed 0.5, then increases as the robot comes nearer to the wall. This means the weight is not high enough to have $\dot{d} = 0$ at this position, which is the desired behavior as we want the robot to stop approaching the wall only at $d^a = 1$.
- If the desired position is $x_2(2,0)$ we have $c = 2$: self-avoidance occurs, as it can be guessed in Fig. 5a. This also occurs at the end of the task to x_1 once the wall is passed, inducing a null weight and a straight line trajectory.

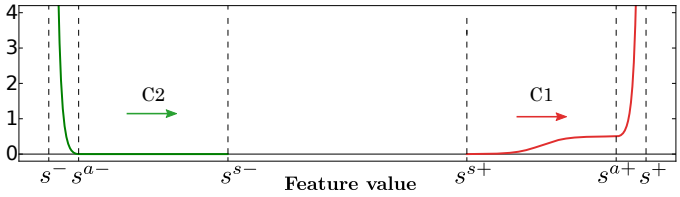
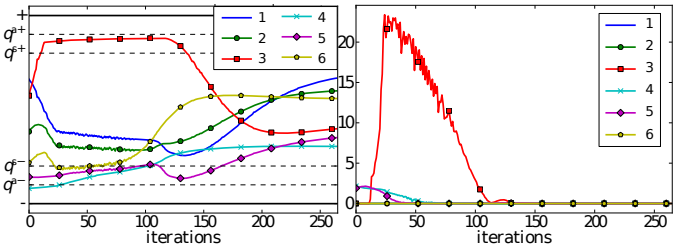


Fig. 6. Weights for configuration C1 (upper bound) and C2 (lower bound). In C1 (red), the feature is going towards its limit and a non-null weight has to be used (here $h^{\min} = 0.5$). In C2 (green), the other features induce the avoidance, hence the weight can be null in the activation area. If the feature still approaches the limits, the generic weighting h^c is used in both cases.

5) Ensuring several constraints: In the case of several constraints having to be ensured simultaneously, coupling terms appear since a system of equations (40) is highly non-linear. A solution still exists to stop all the endangered constraints (for instance $\dot{\mathbf{q}} = 0$ is always a solution) but it would be difficult to compute analytically the corresponding set of optimal weights. The minimal weighting (42) can thus be used together with the generic weighting (12). With this strategy, the weighting is minimal in $[s^{\mathrm{a}-}, s^{\mathrm{a}+}]$ but is still robust to multiple avoidance. Such a weighting is represented in Fig. 6. We have assumed that C1 holds for the upper bound with an optimal weight of $h^{\min} = 0.5$ and that C2 holds for the lower bound, hence the optimal weight is null. If the feature goes out of $[s^{\mathrm{a}-}, s^{\mathrm{a}+}]$ the generic weighting is used for both bounds. In this case, an endangered constraint will have its weight increased until it reaches a sufficient value, which explains why the generic weighting (12) is not bounded. As the sufficient and minimal weights for one constraint depend on the other constraints (see (28) and (42)), this can lead to a general increasing of the weights corresponding to all the endangered constraints until avoidance. In the general case the induced behavior is

Data: $\alpha = 1$, $\alpha^+ > 1$, $\alpha^- < 1$
while *convergence has not been reached* **do**
  compute optimal weights $\mathbf{H} = \text{Diag}(\mathbf{H}_c, \mathbf{H}_t)$;
  compute $\mathbf{v}_e$ using (17);
  if $\|\mathbf{v}_e\| < v_\epsilon$ *and* $\|\mathbf{H}(\mathbf{s} - \mathbf{s}^*)\| > e_\epsilon$ **then**
    $\alpha \leftarrow \alpha^+ \alpha$;
  else
    $\alpha \leftarrow 1 + \alpha^-(\alpha - 1)$;
  end
  apply control law using weights $\text{Diag}(\alpha\mathbf{H}_c, \mathbf{H}_t)$;
end

Algorithm 1: Escape from local minima



(a) Joint positions $\mathbf{q}$ (b) Joint weights $\mathbf{H_q}$

Fig. 7. Joint position and weights while escaping from a local minimum. Oscillations appear in h_3 (red), inducing small oscillations in the robot motion.

satisfactory even if it remains possible to define a task under constraints that would be impossible to perform. In such a case, weights cannot be proven to be finite anymore since the system is no more stable and (27) does not hold. Finally, the sole generic weighting may also be used, leading to a less optimal behavior as seen with the dotted trajectory in Fig. 5a. We now highlight practical issues for the presented system.

IV. Potential issues

Three undesired behaviors may be encountered in the presented system. First, as in all sensor-based approaches, local minima may appear as soon as the system is overdetermined, that is $m > n$. Reaching a desired position where the constraints are active is a second issue. Finally, having potentially high weights may cause oscillations in some cases. In this section, we propose several strategies for each of these issues.

A. Escaping from local minima

The main drawback of the proposed scheme, as for all redundant reactive sensor-based schemes, is the potential existence of local minima. Indeed, as soon as $m > n$ only local stability can be proven. As no planning is considered with a higher-level controller, the approach that has been investigated is to detect that a local minimum has been reached, and try to escape from it. A local minimum is easily detected as it is necessarily a configuration where the end-effector velocity is almost null, while some of the weighted error components $\mathbf{H}(\mathbf{s} - \mathbf{s}^*)$ are not null. The detection condition can thus be defined by two parameters v_ϵ and e_ϵ such as a local minimum corresponds to a configuration where:

$$\|\mathbf{v}_e\| < v_\epsilon \quad \text{and} \quad \|\mathbf{H}(\mathbf{s} - \mathbf{s}^*)\| > e_\epsilon \tag{46}$$

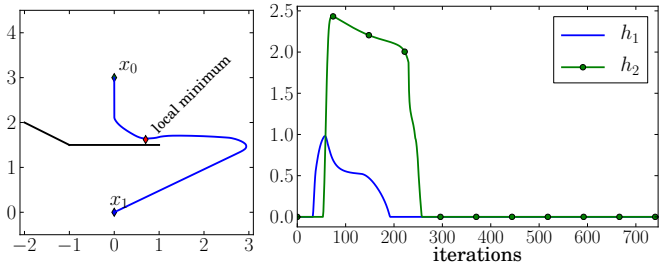
Once a local minimum has been detected, we allow the system to perform non-optimal motion in terms of the sensor-based task, by increasing the weights corresponding to the active constraints. This can be seen as a random walk [2] where we use the structure to compute the escaping motion. We denote $\mathbf{e}_c$ the set of features that regroups the active constraints, and $\mathbf{e}_t$ the other features. The corresponding strategy to modify the weighting matrix is described in Algorithm 1: the weights $\mathbf{H}_c$ are artificially increased by a multiplicative factor α, until reaching a configuration where:

$$\begin{cases} \begin{bmatrix} \mathbf{H}_c & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_t \end{bmatrix}^2 (\mathbf{s} - \mathbf{s}^*) \in \text{Ker } \widehat{\mathbf{J}}_s^+ \\ \begin{bmatrix} \alpha\mathbf{H}_c & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_t \end{bmatrix}^2 (\mathbf{s} - \mathbf{s}^*) \notin \text{Ker } \widehat{\mathbf{J}}_s^+ \end{cases} \tag{47}$$

In this case, the obtained motion is null if $\alpha = 1$ while it is not with the obtained $\alpha > 1$. This may not be true for any given α but in this case Algorithm 1 will carry on increasing α and eventually lead to a configuration that is out of the null space. Meanwhile, if α makes the weights reach very high values, the system is slowed down by the adaptive gain detailed in Section IV-C. α is always equal to 1 as long as no local minimum has been reached. During normal convergence, α is slowly set back to 1. The proposed algorithm makes the active constraints more repulsive, which can be seen as a temporary hierarchy between the active constraints and the other features. Still, such a hierarchy seems natural as constraints have of course to be ensured. On the opposite, going out of a local minimum often prevents from performing optimally the positioning task, as the escaping motion is usually opposed to the motion that is induced by the task. That is why we temporarily increase the weights of the constraints. The tuning of α^+ and α^- may be difficult. In practice, α has to reach a sufficiently high value in order to ensure that the robot will not go back to the same local minimum. A condition to allow the escape from a local minimum is that $\alpha^+\alpha^- > 1$. This corresponds to α increasing faster than it decreases back to 1. The value we used are $\alpha^+ = 1.05$ and $\alpha^- = 0.99$. This strategy is inspired by simulated annealing [3], where the parameter α acts as the annealing temperature. We now show two simulation examples of the proposed algorithm.

a) Joint limits: The induced behavior is represented in Fig. 7, for a simulation of joint limits avoidance in visual servoing. A local minimum occurs around iteration 10. We can see in Fig. 7a that the joint positions are barely evolving from iteration 20 to 70, and that joint limit avoidance is active for joint 3. The joint weights are artificially increased as seen in Fig. 7b. As the most critical joint is q_3, the corresponding weight is far more important than the others. This allows escaping from the local minimum and induced oscillations are very small in practice. Finally, it is interesting to notice that local minima rarely occur wrt. the number of features compared to the available DOFs. In particular, in [23] we have performed exhaustive simulations fusing 2D and 3D visual servoing. No local minima have been found in this configuration.

b) 2D Cartesian robot: We use the robot setup presented in Section III-C4. Walls are set up such that a local minimum



(a) Trajectory (b) Weights

Fig. 8. 2D Cartesian robot escaping from a local minimum (a). Without the proposed strategy the robot is stuck in the indicated position. The corresponding weights (one per wall) (b) are quickly increasing at the beginning, before slowly decreasing.

exists, as shown in Fig. 8a. Without the proposed algorithm the robot ends up in the indicated position. The proposed strategy allows the robot escaping the local minimum, and uses the structure of the task to find an exit. In Fig. 8b we can see that the corresponding weights are quickly increased, before slowly decreasing. This illustrates the balance between α^+ and α^-. Of course, as it is only a reactive scheme some local minima still exist, particularly when the situation is symmetrical. Indeed, in this case increasing the weight would only lead to going backwards. Complex traps such as U-shapes may not be escaped either: in such cases a planning strategy should be used.

B. Reaching an unsafe position

If the desired position is outside the safe area, that is $s^* \in]s^-, s^{s-}] \cup]s^{s+}, s^+]$, the main task cannot be perfectly performed as it does not correspond to the global minimum of the complete weighted task. Indeed, denoting $\mathbf{s} = (\mathbf{s}_t, \mathbf{s}_c)$ where $\mathbf{s}_t$ corresponds to the main task and $\mathbf{s}_c$ to the constraints, the desired position is defined by:

$$\mathbf{q}^* = \arg\min_{\mathbf{q}} \left(\mathbf{e}_t^\top \mathbf{H}_t^2 \mathbf{e}_t\right) \neq \arg\min_{\mathbf{q}} \left(\mathbf{e}^\top \mathbf{H}^2 \mathbf{e}\right) \tag{48}$$

where $\mathbf{e}^\top \mathbf{H}^2 \mathbf{e} = \mathbf{e}_t^\top \mathbf{H}_t^2 \mathbf{e}_t + \mathbf{e}_c^\top \mathbf{H}_c^2 \mathbf{e}_c$. A sufficient condition to overcome the inequality (48) is to ensure that $\mathbf{H}_c = 0$ in a neighborhood of the desired position.

To do so, we introduce a progress parameter $\xi(\|\mathbf{e}_t\|)$ smoothly making the constraint weights null when the main task gets close to completion.

$$\xi(\|\mathbf{e}_t\|) = \begin{cases} 0 & \text{if } \|\mathbf{e}_t\| \leq e_0 \\ 1 & \text{if } \|\mathbf{e}_t\| \geq e_1 \\ \frac{1}{2}\left(1 + \tanh\left(\frac{1}{e_1 - \|\mathbf{e}_t\|} - \frac{1}{\|\mathbf{e}_t\| - e_0}\right)\right) & \text{else} \end{cases} \tag{49}$$

where e_0 and e_1 are defined so that the constraints are totally ignored when the main task is close to completion, that is $\|\mathbf{e}_t\| < e_0$. The corresponding weighting matrix yields $\mathbf{H} = \text{Diag}(\mathbf{H}_t, \xi(\|\mathbf{e}_t\|)\mathbf{H}_c)$ and is equal to $\mathbf{H}^* = \text{Diag}(\mathbf{H}_t, 0)$ at the vicinity of the desired position. The desired position can thus be reached. Finally, depending on the situation, one may or may not use this progress parameter: indeed in some configurations it is preferable to converge to a compromise between the desired position and the constraints, typically if the desired position lies outside of the boundaries of the constraints.

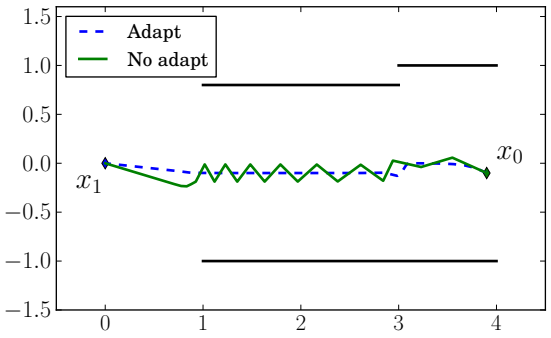


Fig. 9. Oscillations in a corridor. Without the adaptive gain, the robot oscillates between the two walls (green line). The adaptive gain allows drawing a smooth trajectory (dotted blue line).

C. Avoiding oscillations

The generic weights (12) increase when approaching the constraints. When several constraints are being reached, this may lead to oscillations or even to violating the constraints due to discretization. In our case, an efficient way to cope with this issue is an adaptive gain depending on $\|\mathbf{H}\|$ that slows the system in the vicinity of the constraints. The LAS analysis in Section III-B2 is of course still valid with a varying gain, since it can be considered as part of the varying combination matrix. The control gain λ involved in (17) is given by:

$$\lambda(\|\mathbf{H}\|) = (\lambda_0 - \lambda_\infty) e^{-\frac{\lambda_0'}{\lambda_0 - \lambda_\infty}\|\mathbf{H}\|} + \lambda_\infty \tag{50}$$

where:

- $\lambda_0 = \lambda(0)$ is the gain in 0, that is for very small weights
- $\lambda_\infty = \lim_{\|\mathbf{H}\| \to \infty} \lambda(\|\mathbf{H}\|)$ is the gain to infinity, that is for very high weights
- λ_0' is the slope of λ at $\|\mathbf{H}\| = 0$.

In practice we have used the values $\lambda_0 = 1$, $\lambda_\infty = 0.1$ and $\lambda_0' = 0.5$. The proposed strategy is illustrated in simulation in Fig. 9, with the 2D Cartesian robot setup. This time the walls draw a corridor. If the gain is too high, oscillations appear (green line). This is not the case if the adaptive gain is used (dotted blue line).

Finally, in the case of opposed constraints, hence several increasing weights, such an adaptive gain would eventually make the robot stop if no solution exist. This seems an acceptable behavior in such a bad situation.

We now present the experimental results illustrating various aspects of the proposed scheme.

V. Experimental results

In order to illustrate the proposed approach, experiments are carried on a 6-DOFs Gantry robot. The control laws are implemented using ViSP software [33]. We first detail the experimental setup and its calibration. The sensors and constraints are then introduced one after the other in the control law.

The eye-in-hand camera observes a fixed object, the CAD model of which is known. Its edges are tracked to allow for

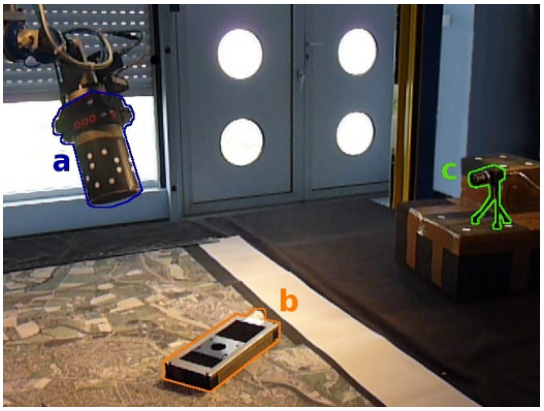


Fig. 10. Experimental setup: (a) Eye-in-hand camera with a 3D landmark, (b) Observed object, (c) Eye-to-hand camera

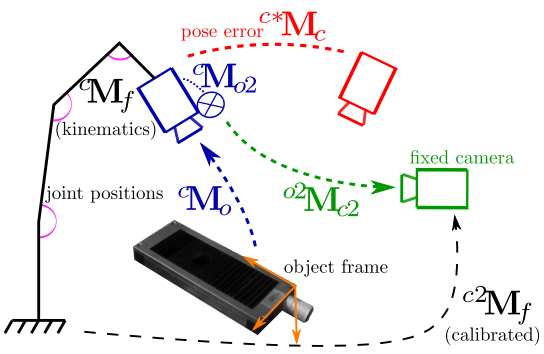


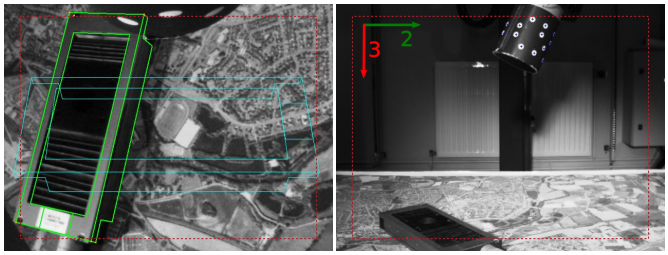
Fig. 11. Integration of various subsystems: hybrid eye-in-hand features for the visibility constraint, eye-to-hand cooperation and joint positions to avoid joint limits.

the pose estimation at camera rate (30Hz) [8]. The eye-in-hand camera carries a landmark that allows its 3D tracking in the eye-to-hand view [32]. The carried landmark is composed by 30 dots. Both cameras are calibrated. The pose between the eye-in-hand camera and the landmark $^{c}\mathbf{M}_{o2}$ is roughly calibrated. The eye-to-hand camera pose wrt. the robot reference frame $^{f}\mathbf{M}_{c2}$ is also roughly calibrated (see Fig. 11). Fig. 12 represents the two initial images. The robot translation joints 2 and 3 are represented in Fig. 12b. Joint 2 thus corresponds to an horizontal motion, while joint 3 corresponds to a vertical motion in the eye-to-hand view. The initial and desired poses make it necessary for the robot to move away from the observed object in order to keep it entirely in the FoV. As we will see, this backward motion makes the end-effector approach not only the upper limit of the eye-to-hand image, but also some joint limits.

As the desired position is out of the safe joint interval, the joint weights are progressively set to 0 (49) according to the strategy exposed in Section IV. The adaptive gain (50) is also computed from the activation matrix norm. We now present the system behavior while the constraints are added one after the other.

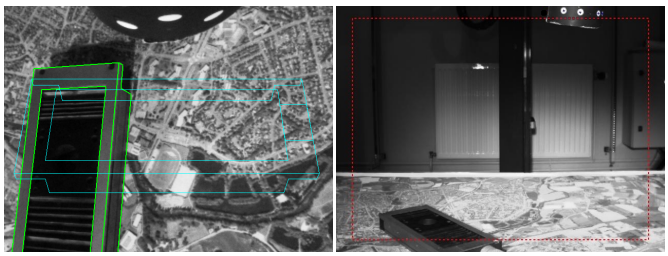
A. Pure position-based visual servo (case 0)

As previously said, the pose between the eye-in-hand camera and the object is estimated at each iteration of the control



(a) Eye-in-hand initial image (b) Eye-to-hand initial image

Fig. 12. Initial images. The object is large in the eye-in-hand image (a). The 3D landmark approaches the top of the eye-to-hand image (b).



(a) Case 0: eye-in-hand image. (b) Case 1: eye-to-hand image

Fig. 13. Without the visibility constraint, the observed object leaves the FoV in case 0 (a). The moving landmark leaves the FoV in case 1 (b).

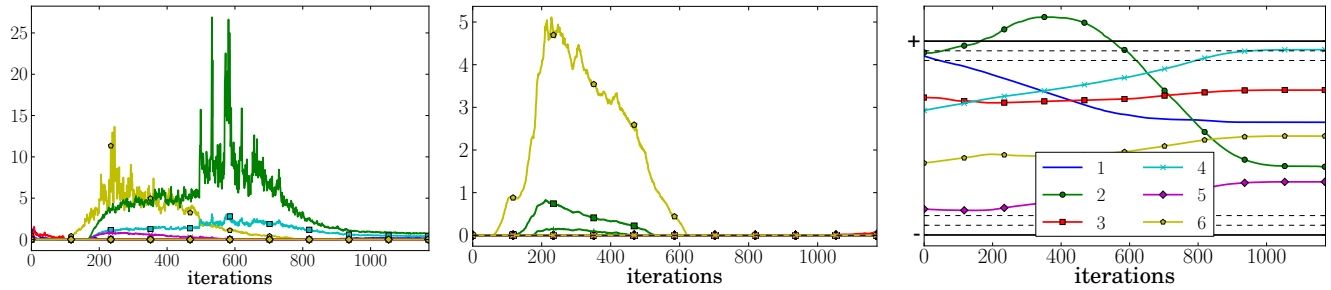
scheme. It is thus possible to perform PBVS [43].

The corresponding 3D features are $\mathbf{s}_{3d} = ({}^{c*}\mathbf{t}_c, {}^{c*}\theta\mathbf{u}_c)$. They describe the transformation between the current and the desired camera pose. The associated desired features is a null vector, and the interaction matrix $\mathbf{L}_{3d}$ is known to be bloc-diagonal, inducing decoupled translational and rotational motions [4]. In perfect conditions, the corresponding camera trajectory is a 3D straight line. The associated weighting is classically constant, that corresponds to $\mathbf{H}_{3d} = \mathbb{I}_6$. Furthermore, this ensures the matrix $\mathbf{HJ}_s$ is full rank, which is a condition for the control law continuity.

The main drawback of PBVS is the lack of control in the image: control is done only in the 3D space and does not ensure the observed object stays in the FoV. In our case, this lack of control clearly appears in Fig. 13a. After few iterations, the object leaves the FoV and the task cannot be performed anymore. We thus add the visibility constraint into the scheme.

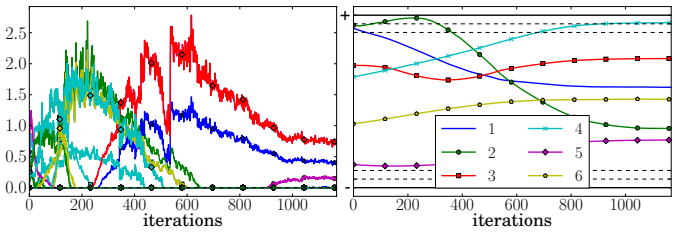
B. Adding the visibility constraint (case 1)

The visibility constraint in visual servoing has been previously addressed through switching control law [14], or visual planning [12], [40], [7], [20]. Here, we define a set of 3D points $({}^{o}\mathbf{x}_1, \ldots, {}^{o}\mathbf{x}_p)$ that are attached to the observed object, typically the nodes of the CAD model. As the camera pose $^{c}\mathbf{M}_o$ is estimated in real time, the 2D coordinates of the projection of the 3D points can easily be computed together with their depth. The visibility constraint is taken into account by adding the feature vector $\mathbf{s}_{2d}$ as the Cartesian coordinates of these 2D points. The well-known analytical expression of the interaction matrix of an image point depends both on its image coordinates (x, y) and on its depth Z [4]. The interaction matrix of $\mathbf{s}_{2d}$ can thus be computed in real time. Similarly, the



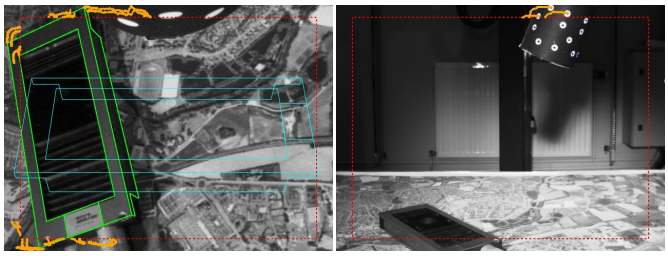
(a) Eye-in-hand weights $\mathbf{H}_{2d}$ (b) Eye-to-hand weights $\mathbf{H}_{\text{ext}}$ (c) Joint positions $\mathbf{q}$

Fig. 15. Case 2. Visibility constraints in eye-in-hand (a) and eye-to-hand (b) are competing at iteration 240. At this time, joint 2 passes its upper limit (c).



(a) Visibility weights $\mathbf{H}_{2d}$ (b) Joint positions $\mathbf{q}$

Fig. 14. Case 1. The weighting is quite small for the visibility constraint (a). The joint positions are inside their limits but joint 2 (green) approaches the upper bound (b).



(a) Eye-in-hand view (iter. 240) (b) Eye-to-hand view (iter. 240)

Fig. 16. Case 2. This time the camera goes to the right of the eye-to-hand image while ensuring the eye-in-hand visibility constraint.

corresponding desired features $\mathbf{s}^*_{2d} = (x^*, y^*)$ are computed from the desired camera pose ${}^{c*}\mathbf{M}_o$. Let (x^-, x^+, y^-, y^+) be the image borders: a safe region can be defined as in (11). In this experiment we use $\rho = 5\%$. Finally, the feature vector is defined by:

$$\mathbf{s} = \begin{bmatrix} \mathbf{s}_{3d} \\ \mathbf{s}_{2d} \end{bmatrix} \quad \begin{matrix} \cdots \text{ PBVS (dim. 6)} \\ \cdots \text{ Visibility (dim. } 2\times 12) \end{matrix} \tag{51}$$

where the dimension of the feature vectors are detailed: 6 components for the PBVS, and 2×12 for the visibility constraint (12 nodes in the object CAD model). The corresponding weighting matrix is $\mathbf{H} = \text{Diag}(\mathbb{I}_6, \mathbf{H}_{2d})$ where $\mathbf{H}_{2d}$ is derived from (9) using $h^t = 0$ and h^c given by (12). We can notice that the global minimum corresponds to the desired pose: indeed, if $\mathbf{s}_{3d} = \mathbf{s}^*_{3d}$ then ${}^c\mathbf{M}_o = {}^{c*}\mathbf{M}_o$, and $\mathbf{s}_{2d} = \mathbf{s}^*_{2d}$. Hence, the progress parameter (49) is not used for this constraint, as the robot will converge to the desired pose even if some constraints are active. The resulting images are shown in Fig. 16. The active nodes are plotted in orange for the visibility constraint. This time the object stays in the FoV during the whole scheme. The visibility weights $\mathbf{H}_{2d}$ are represented in Fig. 14a. Their value remains small ($h < 3$) and yet allows ensuring the constraint. Several features are active around iteration 240. The maximum value is obtained around iteration 600 for only one 2D feature. This corresponds to one of the nodes approaching the left border during the rotation around the optical axis (see the video accompanying this paper). Joint positions (Fig. 14b) stay inside their limits, yet no avoidance is specified in this experiment. Finally, Fig. 13b shows the 3D landmark goes out of the eye-to-hand view around iteration 240, that is when the camera moves away from the object to keep it in the FoV.

C. Adding the eye-to-hand visibility constraint (case 2)

We now take into account the visibility constraint in the eye-to-hand view. The modeling is the same as previously exposed. The considered points are the 30 points from the 3D landmark. We denote $\mathbf{s}_{\text{ext}}$ the corresponding 2D features. The global feature vector is thus $\mathbf{s} = (\mathbf{s}_{3d}, \mathbf{s}_{2d}, \mathbf{s}_{\text{ext}})$ and the weighting matrix is $\mathbf{H} = \text{Diag}(\mathbf{H}_{3d}, \mathbf{H}_{2d}, \mathbf{H}_{\text{ext}})$ where $\mathbf{H}_{\text{ext}}$ is defined exactly as $\mathbf{H}_{2d}$.

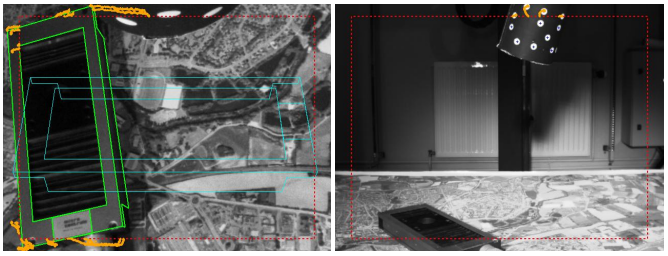
The resulting images are shown in Fig. 16. This time, the 3D landmark stays in the eye-to-hand FoV. The eye-in-hand visibility constraint can still be ensured as the camera moves to the right instead of moving up. As seen in Fig. 15c, this makes joint 2 (green) pass its upper limit (which is not the real limit so that it has been possible to realize this experiment).

The corresponding weights are represented in Fig. 15. Adding a new constraint makes the visibility weights $\mathbf{H}_{2d}$ increase when compared to the previous section. Indeed, eye-in-hand and eye-to-hand visibility constraints are competing around iteration 240 which makes the eye-in-hand weights pass 10, while one of the eye-to-hand weights reaches 5. As previously, the maximum weight is reached around iteration 600 for the visibility constraint.

D. Adding the joint limits avoidance (case 3)

We now take into account the joint positions in the task. The global feature vector yields:

$$\mathbf{s} = \begin{bmatrix} \mathbf{s}_{3d} \\ \mathbf{s}_{2d} \\ \mathbf{s}_{\text{ext}} \\ \mathbf{q} \end{bmatrix} \quad \begin{matrix} \cdots \text{ PBVS (dim. 6)} \\ \cdots \text{ Eye-in-hand visibility (dim. } 2\times 12) \\ \cdots \text{ Eye-to-hand visibility (dim. } 2\times 30) \\ \cdots \text{ Joint positions (dim. 6)} \end{matrix} \tag{52}$$



(a) Eye-in-hand view (iter. 240) (b) Eye-to-hand view (iter. 240)

Fig. 17. Case 3. The camera cannot move to the right anymore when the observed object is large in the eye-in-hand image (a). This time the 3D landmark comes towards the eye-to-hand camera while rotating around the optical axis (b).

The corresponding weighting matrix is thus $\mathbf{H} = \text{Diag}(\mathbf{H}_{3d}, \mathbf{H}_{2d}, \mathbf{H}_{\text{ext}}, \mathbf{H}_{\mathbf{q}})$ where $\mathbf{H}_{\mathbf{q}}$ regroups the joint weights. We use the strategy exposed in Section III-C2: $\mathbf{H}_{\mathbf{q}}$ corresponds to the optimal weighting (43). The activation and safe areas are defined with $\rho = 10\%$ and $\rho^{\text{a}} = 5\%$. For this constraint, the progress parameter (49) is used as the desired position is likely to lie in the joint unsafe area.

Fig. 17 shows the eye-in-hand and eye-to-hand images that correspond to iterations 240, when the main difficulty occurs: some 2D points are very near to the image border in both views. We can see in Fig. 19c that at the same time one of the joint limits is being avoided very closely. This corresponds to a configuration where the camera has to move away from the object in order to keep it in the FoV, but has its motion limited by both the eye-to-hand visibility constraint and the joint limit. The values of the weights are represented in Fig. 18 and clearly reflect this phenomenon. Indeed, all three curves indicate that the constraints are endangered at the same time. Several weights reach their maximum around iteration 240. The corresponding values are higher than in the previous cases: the eye-in-hand visibility constraint has some weights reaching 20, while the eye-to-hand and the joint weights reach 5. As previously announced these values are still acceptable and do not endanger the conditionning of $\mathbf{H}$ and the whole system is stable. Reducing artificially the joint limit would typically lead to a configuration where the task could actually not be performed without violating the constraints and the robot would have stopped at this position.

As previously, a second peak value is reached for one visibility weight around iteration 600. This time the corresponding value is less than 10, compared to 25 in case 2. This constraint is thus less endangered by the trajectory from case 3 than the one from case 2. As for the general behavior of the robot, Fig. 19a reveals that some oscillations appear in the velocity setpoint. This is due to the high number of constraints that are near to violation at the same time. We can notice in Fig. 18c that the adaptive gain is reduced by 10 at this time. This is clearly visible in Fig. 19a that the system slows down around iteration 240. The measurement of the joint positions in Fig. 19c shows that the general motion remains smooth all along the task and especially even when joint 2 is near its limit. Finally, Fig. 19b represents the PBVS error. Even if the corresponding weighting matrix is the identity, the other weights prevents the PBVS from decreasing exponentially. The convergence is still satisfactory and oscillations are quite small.

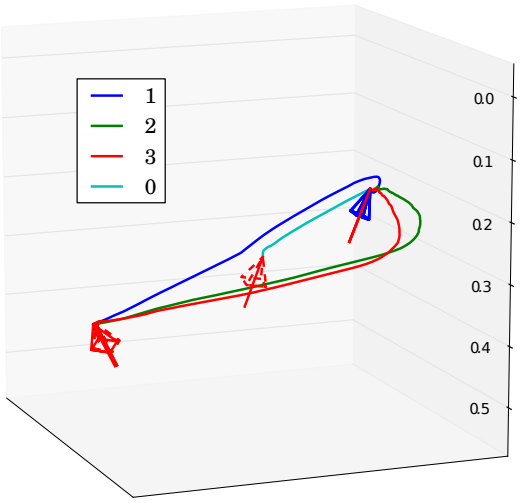


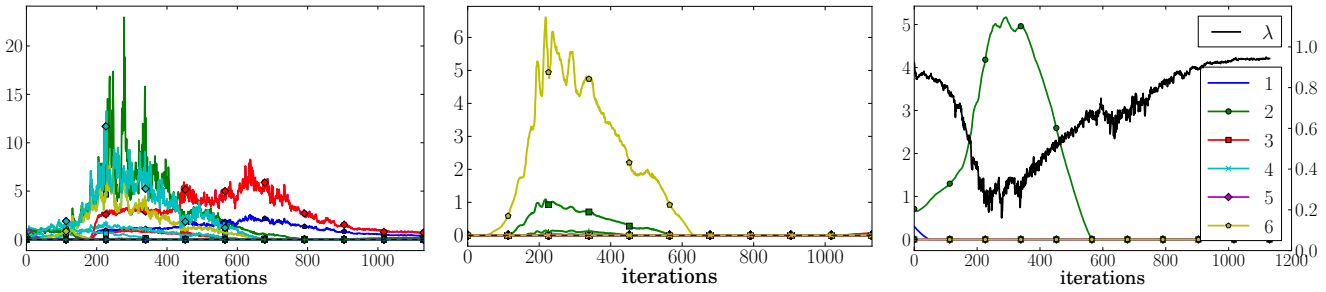
Fig. 20. 3D trajectories for the presented cases, observed from the eye-to-hand camera. Pure PBVS (cyan) begins by a straight line, before getting inconsistent when the tracker loses the object. Case 1 (blue) corresponds to the trajectory that reaches the highest point, as the eye-to-hand visibility is not taken into account. Case 2 (green) makes the camera go to the right instead of going up. Finally, case 3 (red) forces the camera to draw another trajectory in order to ensure all the constraints.

E. Comparison between the several cases

Fig. 20 compares the trajectories corresponding to the presented experiments. This shows very clearly that the end-effector has many available trajectories to perform the positioning task. However, not all of them respect the constraints: actually only the last configuration (case 3, red) does. When performing only PBVS (case 0, cyan), the trajectory is a straight line till the failure due to the tracker losing the object. When not using the eye-to-hand image, the camera tends to go up (case 1, blue). On the opposite, the runs that use the eye-to-hand camera share a lower trajectory. In particular, the camera going right instead of going up is clearly visible for case 2 (green). We highlight that no local minimum is reached for any combination of features. The implicit concurrency allows adding new constraints without having to model the potential coupling.

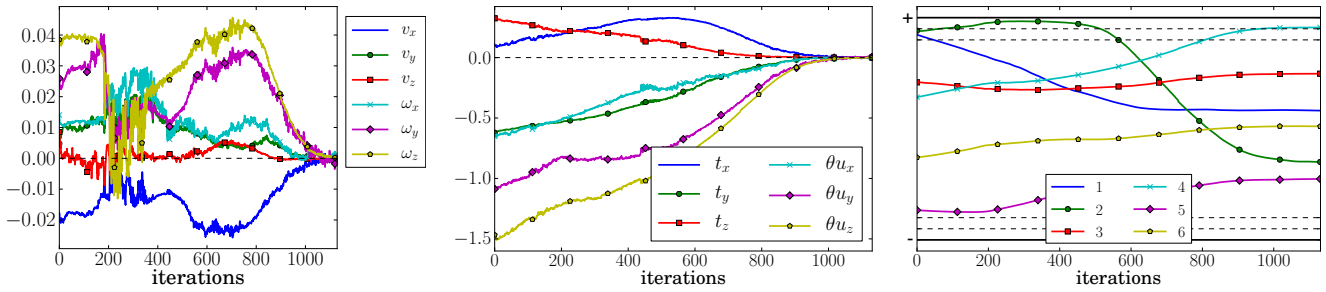
F. Other experiments

The presented experiment illustrates the general properties of our approach. In complementary works, we have also highlighted specific aspects through other types of experiments. In particular, exhaustive simulations have been carried for the visibility constraint (case 1). We have simulated 9900 servoings, that is a large set of combinations for 100 initial and final poses, showing that 98% converge while ensuring the visibility constraint with the maximum weight being less than 20. The other cases converge with higher weights, or with a smaller control gain. In [23], our approach has also been compared with other control laws that address the visibility



(a) Eye-in-hand weights $\mathbf{H}_{2d}$ (b) Eye-to-hand weights $\mathbf{H}_{\text{ext}}$ (c) Joint weights and adaptive gain

Fig. 18. Weights of the different subsystems: eye-in-hand 2D points (left), eye-to-hand 2D points (middle) and joint positions (right). All constraints occur around iteration 300, making the weights have significative values. The adaptive gain (black curve) shows the system slows down at this moment.



(a) Camera velocity setpoint (b) PBVS error (c) Joint positions

Fig. 19. General behavior of the robot. Camera velocity (left) shows some oscillations when passing the vicinity of all constraints. PBVS error (middle) indicates that the visual servoing is performed during the task. Joint positions (right) highlight the limit being avoided.

constraint. The joint limits avoidance has been validated in [22] on a 6-DOF robot arm Adept Viper850 for several positioning tasks in visual servoing. Finally, the proposed framework has also been used in ultrasound images in [27], to maintain the visibility of an anatomic element of interest during tele-echography.

VI. Conclusion

This paper has proposed a generic approach for multi-sensor and multi-constraints fusion in sensor-based control. The literature classically addresses this issue by a hierarchical approach or by performing a weighted mean of the velocities that are computed for each task. We proposed to perform the data fusion at the level of the features, by introducing a dynamic weighting matrix. While some tuning aspects are similar to LQ control, activation and deactivation of the features is part of the varying-feature-set approach, that has been recently formalized for one sensor [31]. The general idea is that a robotic system can handle a high-dimensional task and several constraints without having to explicit the hierarchy or performing a manual tuning of the feature weights. The main properties of the proposed control law have been exposed, concurring to the classical conditions on the system rank with local asymptotic stability and potential local minima in the case of redundancy. The scheme is generic even for sensors that are not rigidly attached to the end-effector frame. The main drawbacks of the proposed scheme are related to its nature being only reactive. The additional strategies that are proposed for the particular cases of local minima and unsafe desired position both consist in modifying the activation matrix independently from its initial design in terms of subsystems integration. This can be viewed as the beginning of a higher-level controller that takes into account the global configuration and balances the weighting matrix so that the induced trajectory avoids or escapes local minima. The versatility of the approach has been illustrated by considering a multi-sensor, multi-constraint task. Several experiments have shown that the proposed approach can handle various combinations of sensors and constraints for a positioning task. Future work will consist in extending this framework to other types of sensors such as laser range or haptic devices. Other strategies, such as relaxing some constraints, could also increase the convergence domain of the proposed scheme. This could be suited for instance for the visibility constraint, where some parts of the object could be allowed to leave the field of view.

Acknowledgment

The authors would like to acknowledge Pierre-Brice Wieber for his valuable feedback on this work.

References

[1] G. Allibert, E. Courtial, and F. Chaumette, "Predictive control for constrained image-based visual servoing," *IEEE Trans. on Robotics*, vol. 26, no. 5, pp. 933–939, 2010.

[2] J. Barraquand and J. Latombe, "Robot motion planning: A distributed representation approach," *Int. Journal of Robotics Research*, vol. 10, no. 6, pp. 628–649, 1991.

[3] S. Brooks and B. Morgan, "Optimization using simulated annealing," *The Statistician*, vol. 44, no. 2, pp. 241–257, 1995.

[4] F. Chaumette and S. Hutchinson, "Visual servo control. I. Basic approaches," *IEEE Robot. Autom. Mag.*, vol. 13, no. 4, pp. 82–90, 2006.

[5] C. Cheah, D. Wang, and Y. Sun, "Region-reaching control of robots," *IEEE Trans. on Robotics*, vol. 23, no. 6, pp. 1260–1264, 2007.

[6] G. Chesi, "Visual servoing path planning via homogeneous forms and LMI optimizations," *IEEE Trans. on Robotics*, vol. 25, no. 2, pp. 281–291, 2009.

[7] G. Chesi and Y. Hung, "Global path-planning for constrained and optimal visual servoing," *IEEE Trans. on Robotics*, vol. 23, no. 5, pp. 1050–1060, 2007.

[8] A. Comport, E. Marchand, and F. Chaumette, "Efficient model-based tracking for robot vision," *Advanced Robotics*, vol. 19, no. 10, pp. 1097–1113, October 2005.

[9] ——, "Statistically robust 2-D visual servoing," *IEEE Trans. on Robotics*, vol. 22, no. 2, pp. 415–420, 2006.

[10] P. Danes and D. Bellot, "Towards an LMI approach to multicriteria visual servoing in robotics," *European journal of control*, vol. 12, no. 1, p. 86, 2006.

[11] W. Decré, R. Smits, H. Bruyninckx, and J. De Schutter, "Extending iTaSC to support inequality constraints and non-instantaneous task specification," in *IEEE Int. Conf. on Robotics and Automation*, 2009, pp. 964–971.

[12] L. Deng, F. Janabi-Sharifi, and W. Wilson, "Hybrid motion control and planning strategies for visual servoing," *IEEE Trans. on Industrial Electronics*, vol. 52, no. 4, pp. 1024–1040, 2005.

[13] F. Fahimi, C. Nataraj, and H. Ashrafiuon, "Real-time obstacle avoidance for multiple mobile robots," *Robotica*, vol. 27, no. 2, pp. 189–198, Mar. 2009.

[14] N. Gans and S. Hutchinson, "Stable visual servoing through hybrid switched-system control," *IEEE Trans. on Robotics*, vol. 23, no. 3, pp. 530–540, 2007.

[15] N. García-Aracil, E. Malis, R. Aracil-Santonja, and C. Pérez-Vidal, "Continuous visual servoing despite the changes of visibility in image features," *IEEE Trans. on Robotics*, vol. 21, no. 6, pp. 1214–1220, 2005.

[16] A. Hafez and C. Jawahar, "Visual servoing by optimization of a 2D/3D hybrid objective function," in *IEEE Int. Conf. on Robotics and Automation*, Roma, Italy, 2007, pp. 1691–1696.

[17] K. Hosoda, K. Igarashi, and M. Asada, "Adaptive hybrid control for visual and force servoing in an unknown environment," *IEEE Robot. Autom. Mag.*, vol. 5, no. 4, pp. 39–43, 1998.

[18] A. Isidori, *Nonlinear control systems*. Springer Verlag, 1995.

[19] O. Kanoun, F. Lamiraux, and P.-B. Wieber, "Kinematic Control of Redundant Manipulators: Generalizing the Task-Priority Framework to Inequality Task." *IEEE Trans. on Robotics*, vol. 27, no. 4, pp. 785–792, 2011.

[20] M. Kazemi, M. Mehrandezh, and K. Gupta, "Kinodynamic planning for visual servoing," in *IEEE Int. Conf. on Robotics and Automation*, 2011, pp. 2478–2484.

[21] F. Keith, P. Wieber, N. Mansard, and A. Kheddar, "Analysis of the Discontinuities in Prioritized Tasks-Space Control Under Discreet Task Scheduling Operations," September 2011.

[22] O. Kermorgant and F. Chaumette, "Avoiding joint limits with a low-level fusion scheme," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, San Francisco, USA, September 2011, pp. 768–773.

[23] ——, "Combining IBVS and PBVS to ensure the visibility constraint," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, San Francisco, USA, September 2011, pp. 2849–2854.

[24] ——, "Multi-sensor data fusion in sensor-based control: application to multi-camera visual servoing," in *IEEE Int. Conf. on Robotics and Automation*, Shanghai, China, May 2011, pp. 4518–4523.

[25] O. Khatib, "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots," *The International Journal of Robotics Research*, vol. 5, no. 1, pp. 90–98, 1986.

[26] F. Lewis and V. Syrmos, *Optimal control*. Wiley-Interscience, 1995.

[27] T. Li, O. Kermorgant, and A. Krupa, "Maintaining visibility constraints during tele-echography with ultrasound visual servoing," in *IEEE Int. Conf. on Robotics and Automation*, Saint Paul, USA, May 2012.

[28] E. Malis, G. Morel, and F. Chaumette, "Robot Control Using Disparate Multiple Sensors," *Int. Journal of Robotics Research*, vol. 20, no. 5, pp. 364–377, May 2001.

[29] N. Mansard and F. Chaumette, "Task sequencing for sensor-based control," *IEEE Trans. on Robotics*, vol. 23, no. 1, pp. 60–72, Feb. 2007.

[30] N. Mansard, O. Khatib, and A. Kheddar, "A unified approach to integrate unilateral constraints in the stack of tasks," *IEEE Trans. on Robotics*, vol. 25, no. 3, pp. 670–685, 2009.

[31] N. Mansard, A. Remazeilles, and F. Chaumette, "Continuity of varying-feature-set control laws," *IEEE Trans. Autom. Control*, vol. 54, no. 11, pp. 2493–2505, November 2009.

[32] E. Marchand, F. Chaumette, F. Spindler, and M. Perrier, "Controlling an uninstrumented manipulator by visual servoing," *The International Journal of Robotics Research*, vol. 21, no. 7, p. 635, 2002.

[33] E. Marchand, F. Spindler, and F. Chaumette, "ViSP for visual servoing: a generic software platform with a wide class of robot control skills," *IEEE Robot. Autom. Mag.*, vol. 12, no. 4, December 2005.

[34] M. Marey and F. Chaumette, "A new large projection operator for the redundancy framework," in *IEEE Int. Conf. on Robotics and Automation*, Anchorage, Alaska, May 2010.

[35] Y. Mezouar and F. Chaumette, "Design and tracking of desirable trajectories in the image space by integrating mechanical and visibility constraints," in *IEEE Int. Conf. on Robotics and Automation*, vol. 1, 2001, pp. 731–736.

[36] B. Nelson and P. Khosla, "Strategies for increasing the tracking region of an eye-in-hand system by singularity and joint limit avoidance," *The Int. J. of Robotics Research*, vol. 14, no. 3, p. 255, 1995.

[37] N. Papanikolopoulos, P. Khosla, and T. Kanade, "Visual tracking of a moving target by a camera mounted on a robot: A combination of control and vision," *IEEE Trans. Robot. Autom.*, vol. 9, no. 1, pp. 14–35, 1993.

[38] R. Paul, *Robot manipulators: mathematics, programming, and control: the computer control of robot manipulators*. The MIT Press, 1981.

[39] C. Samson, M. Le Borgne, and B. Espiau, *Robot Control : The task function approach*. Clarendon Press, 1991.

[40] F. Schramm and G. Morel, "Ensuring visibility in calibration-free path planning for image-based visual servoing," *IEEE Trans. on Robotics*, vol. 22, no. 4, pp. 848–854, 2006.

[41] O. Tahri and Y. Mezouar, "On visual servoing based on efficient second order minimization," *Robotics and Autonomous Systems*, 2009.

[42] C. Van Loan, "On the method of weighting for equality-constrained least-squares problems," *SIAM Journal on Numerical Analysis*, pp. 851–864, 1985.

[43] W. Wilson, W. Hulls, and G. Bell, "Relative end-effector control using cartesian position based visual servoing," *IEEE Trans. on Robotics and Automation*, vol. 12, no. 5, pp. 684–696, 2002.

[44] B. Wittenmark, R. Evans, and Y. Soh, "Constrained pole-placement using transformation and LQ-design* 1," *Automatica*, vol. 23, no. 6, pp. 767–769, 1987.

[45] T. Yoshikawa, "Basic optimization methods of redundant manipulators," *Laboratory Robotics and Automation*, vol. 8, no. 1, pp. 49–60, 1996.

Olivier Kermorgant graduated from École Centrale Paris, France in 2004. For two years he has been a Research Engineer in the Measurement and Control department at Arcelor Research, Metz, France. From 2008 to 2011 he was with the Lagadic group at Inria Rennes where he received the Ph.D. degree in signal processing from University of Rennes, France in 2011. He then joined the Ocean Systems Laboratory at Heriot-Watt University, Edinburgh, Scotland as a Research Assistant.
Since 2012 he has been Assistant Professor at University of Strasbourg, France. His research interests include sensor-based robot control, disturbance rejection and optimization.

François Chaumette graduated from École Nationale Supérieure de Mécanique, Nantes, France, in 1987. He received the Ph.D. degree in computer science from the University of Rennes, France, in 1990. Since 1990, he has been with Inria in Rennes where he is now senior research scientist and head of the Lagadic group (`http://www.irisa.fr/lagadic`). His research interests include robotics and computer vision, especially visual servoing and active perception.

Dr. Chaumette is IEEE Fellow. He received the AFCET/CNRS Prize for the best French thesis in automatic control in 1991. He also received the 2002 King-Sun Fu Memorial Best IEEE Transactions on Robotics and Automation Paper Award. He has been Associate Editor of the IEEE Transactions on Robotics from 2001 to 2005 and is now in the Editorial Board of the Int. Journal of Robotics Research.