Avoiding joint limits with a low-level fusion scheme

Olivier Kermorgant and François Chaumette

***Abstract*— Joint limits avoidance is a crucial issue in sensor-based control. In this paper we propose an avoidance strategy based on a low-level data fusion. The joint positions of a robot arm are considered as features that are continuously added to the control scheme when they approach the joint limits, and removed when the position is safe. We expose an optimal tuning of the avoidance scheme, ensuring the main task is disturbed as little as possible. We propose additional strategies to solve the particular cases of unsafe desired position and local minima. The control scheme is applied to the avoidance of joint limits while performing visual servoing. Both simulation and experimental results illustrate the validity of our approach.**

***Index Terms*— Joint limits avoidance, sensor fusion, multi-sensor, visual servoing,**

I. Introduction

Robot control from sensor inputs leads to particular issues, including joint limits avoidance, since a task that is defined in the sensor space may lead to motions that reach the robot joint limits. Different aspects of redundancy are classically used to solve this problem. The gradient projection method (GPM) has been widely used [15], [16], [17] and consists in modeling the avoidance with a cost function, the gradient of which is projected on to the null space of the sensor-based task. In that case the avoidance scheme cannot disturb the main task, hence the robot joint limits may be reached in some cases. Variations of GPM have been proposed in [1] for the scaling of a velocity saturation scheme, and in [10], where a nonlinear projection improves the injection of the secondary task. This approach has been recently extended in [13], where full-rank tasks are considered and where the scaling of the avoidance scheme is optimal. However, problems may still occur when avoiding several limits at the same time. The other popular schemes consist in somehow determining the best compromise between performing the sensor-based task, and avoiding the joint limits. This can be expressed with a weighted least norm formulation [3] or within a LQ scheme [14]. The classical drawback of joint limits avoidance methods is that it is difficult to ensure limits avoidance and main task performing at the same time. Also, particular configurations can endanger the strategies, for instance if the main task is full rank or if the robot desired position corresponds to an unsafe joint position. Our work uses the low-level sensor fusion method [9] to merge the sensor data with the joint positions into a single task. This allows us to propose an optimal tuning of the avoidance scheme, that disturbs the main task as little as possible and takes into account full-rank main tasks. Additional strategies are proposed to solve issues that may occur in particular configurations: first, the case of unsafe desired position is considered. We then address the problem of local minima that may appear even the main task is globally stable.
Section II introduces the general modeling, then the analytical computation of the optimal tuning is exposed in Section III. Simulations are presented in Section IV to illustrate specific aspects of the proposed scheme. Finally, Section V exposes experimental results.

Olivier Kermorgant and François Chaumette are with INRIA Rennes-Bretagne Atlantique, Campus de Beaulieu, Rennes, France.
E-mail: Firstname.Name@inria.fr

II. General modeling

A. Sensor-based control

We consider a robotic system equipped with k sensors providing data about the robot pose in its environment. 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 $m \geq 6$. The low-level fusion scheme [9] defines the m-dimensional signal of the multi-sensor set as $\mathbf{s}_s = (\mathbf{s}_1, \ldots, \mathbf{s}_k)$. Considering a reference frame $\mathcal{F}_e$ in which the robot is controlled, the time derivative of $\mathbf{s}_s$ is directly related to the robot velocity screw $\mathbf{v}_e$:

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

with:

$$\mathbf{L}_s = \mathbf{L}\mathbf{W}_e = \begin{bmatrix} \mathbf{L}_1 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & \mathbf{L}_k \end{bmatrix} \begin{bmatrix} {}^1\mathbf{W}_e \\ \vdots \\ {}^k\mathbf{W}_e \end{bmatrix} \tag{2}$$

where $\mathbf{L} \in \mathbb{R}^{m \times 6k}$ contains the interaction matrices of the sensors [4] and $\mathbf{W}_e \in \mathbb{R}^{6k \times 6}$ contains the screw transformation matrices, making $\mathbf{L}_s \in \mathbb{R}^{m \times 6}$ the global interaction matrix of the task [9]. Denoting ${}^e\mathbf{J}_q \in \mathbb{R}^{m \times n}$ the robot jacobian, $\dot{\mathbf{s}}_s$ can be expressed with the articular velocity $\dot{\mathbf{q}}$:

$$\dot{\mathbf{s}}_s = \mathbf{L}_s {}^e\mathbf{J}_q \dot{\mathbf{q}} = \mathbf{J}_s \dot{\mathbf{q}} \tag{3}$$

and a classical controller is then:

$$\dot{\mathbf{q}} = -\lambda \mathbf{J}_s^+ \mathbf{e}_s \tag{4}$$

where $\mathbf{e}_s = \mathbf{s}_s - \mathbf{s}_s^*$ is the task error, depending on the desired sensor features $\mathbf{s}_s^*$.

B. Joint limits avoidance

When dealing with joint limits avoidance, we want to perform the defined sensor-based task unless the joints are close to their limits, denoted q^- and q^+. To do so, a safe interval $[q^{s-}, q^{s+}]$ is classically defined for each joint:

$$\begin{cases} q_i^{s-} & = q_i^- + \rho(q_i^+ - q_i^-) \\ q_i^{s+} & = q_i^+ - \rho(q_i^+ - q_i^-) \end{cases} \tag{5}$$

where $\rho \in [0, 0.5]$ is a tuning parameter. In the redundancy framework, a projection operator is classically used so that the avoidance task is realized at best under the constraint that the main task is not disturbed [15], [16], [17]. However, joints may be reached if the avoidance task is in the null space of the projection operator.

C. Joint positions as additional features

Assuming that a subset $\mathbf{q}_u = \mathbf{D_q q}$ of p robot joints are in an unsafe position, the low-level fusion scheme (3) can be increased by adding a term corresponding to the unsafe joint position error $\mathbf{e}_q = \mathbf{q}_u - \mathbf{q}_u^*$, with:

$$q_i^* = \begin{cases} q_i^{\mathrm{s}-} & \text{if } q_i \leq q_i^{\mathrm{s}-} \\ q_i^{\mathrm{s}+} & \text{if } q_i \geq q_i^{\mathrm{s}+} \end{cases} \tag{6}$$

The framework of varying-feature-set [11] allows continuous adding or removing task components by defining a weighted error $\mathbf{e_H} = \mathbf{He}$ where $\mathbf{H}$ is a diagonal activation matrix. The error derivative yields $\dot{\mathbf{e}}_\mathbf{H} = \mathbf{H}\dot{\mathbf{e}} + \dot{\mathbf{H}}\mathbf{e}$, where the second term is usually neglected, that is:

$$\dot{\mathbf{e}}_\mathbf{H} = \mathbf{H}\begin{bmatrix} \dot{\mathbf{s}}_s \\ \dot{\mathbf{q}}_u \end{bmatrix} = \mathbf{H}\begin{bmatrix} \mathbf{J}_s \\ \mathbf{D_q} \end{bmatrix}\dot{\mathbf{q}} = \mathbf{HJ}\dot{\mathbf{q}} \tag{7}$$

where $\mathbf{J} \in \mathbb{R}^{(m+p)\times n}$ is the global jacobian. The corresponding control law is computed the same way than in Section II-A and becomes:

$$\dot{\mathbf{q}} = -\lambda(\mathbf{HJ})^+\mathbf{He} \tag{8}$$

Several schemes have been designed with such a weighting matrix within a set of sensor-based features: in [6] the activation matrix is used to address the problem of outliers in robust visual servoing, while in [5] it defines a task in term of a desired region instead of a desired position. In [8], the visual features are deactivated in the case of visibility lost. In the case of joint limits avoidance, we use a particular weighting matrix:

$$\mathbf{H} = \mathrm{Diag}(1,\ldots,1,h_1,\ldots,h_p) = \begin{bmatrix} \mathbb{I}_m & 0 \\ 0 & \tilde{\mathbf{H}} \end{bmatrix} \tag{9}$$

where the weights corresponding to the sensor features are always equal to 1, while the weights corresponding to the joints are varying in order to allow smooth activation and deactivation of the unsafe joints. In practice, taking only the unsafe joint features into account is equivalent to adding all joint components with a null weight for the safe ones.

As explained in [11], the varying-feature-set formalism ensures the continuity of the control law when $\mathbf{HJ}$ and $\mathbf{He}$ are continuous, with the pseudo-inverse also continuous. A sufficient condition for the latter is the matrix $\mathbf{HJ}$ being of constant rank. In the following, this condition is ensured by assuming $\mathbf{J}_s$ is full rank. Continuity of $\mathbf{HJ}$ and $\mathbf{He}$ is ensured by the computation of the joint weights (h_i), which is exposed in Section III. In the following, we denote $\mathbf{e}_s$ as the main task, that is when $p = 0$ and $\mathbf{H} = \mathbb{I}_m$.

III. Weights computation

The scheme that is presented in Section II-C is not a sum between joint velocities computed for the main task, and joint velocities computed for the avoidance task. The fusion approach draws a compromise between the main task performing and joint limits avoidance, that depends on the activation matrix $\tilde{\mathbf{H}}$. The tuning of the avoidance scheme is a crucial issue in joint limits avoidance: if activated too slowly joints may be reached, while oscillations may appear if the avoidance scheme has a large magnitude. In the fusion formalism, as there is no projection operator preventing the avoidance task from disturbing the main task, the activation matrix has to be designed with minimal weights in order to disturb the main task as little as possible. We address this problem with the following form of the activation weights:

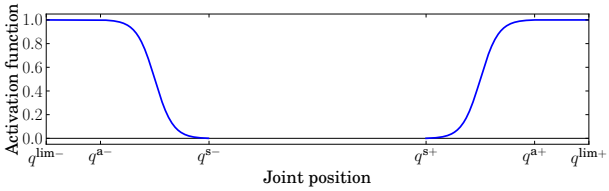


Fig. 1. Activation function for lower and upper bounds.

$$\forall i, h_i = \mu_i(q_i)h_i^{\mathrm{opt}} \tag{10}$$

where $\mu_i(q_i) \in [0,1]$ is an activation function to ensure the continuity of the weights and $h_i^{\mathrm{opt}} \geq 0$ is a scale factor ensuring that the limit is avoided. We now define these two parameters.

A. Activation function

To ensure continuity of $\mathbf{HJ}$ and $\mathbf{He}$, weights must be null at feature activation and deactivation, and increasing as the joint values vary from the safe limit to the physical limit. As in (5) we define the positions $q^{\mathrm{a}-}$ and $q^{\mathrm{a}+}$ from where the activation is full:

$$\begin{cases} q_i^{\mathrm{a}-} & = q_i^- + \rho_\mathbf{q}^\mathrm{a}(q_i^+ - q_i^-) \\ q_i^{\mathrm{a}+} & = q_i^+ - \rho_\mathbf{q}^\mathrm{a}(q_i^+ - q_i^-) \end{cases} \tag{11}$$

where $\rho_\mathbf{q}^\mathrm{a} < \rho$ is a tuning parameter. The activation function can then be defined with a sigmoid:

$$\mu_i = \begin{cases} \frac{1}{2}\left(1+\tanh(\frac{1}{q_i^{\mathrm{a}+}-q_i} - \frac{1}{q_i-q_i^{\mathrm{s}+}})\right) & \text{if } q_i^{\mathrm{s}+} < q_i < q_i^{\mathrm{a}+} \\ \frac{1}{2}\left(1-\tanh(\frac{1}{q_i^{\mathrm{a}-}-q_i} - \frac{1}{q_i-q_i^{\mathrm{s}-}})\right) & \text{if } q_i^{\mathrm{a}-} < q_i < q_i^{\mathrm{s}-} \\ 1 & \text{else} \end{cases} \tag{12}$$

μ_i is $\mathcal{C}^\infty$ and smoothly increases the weight as the joint reaches the limit, with $\mu_i(q_i^{\mathrm{a}-}) = \mu_i(q_i^{\mathrm{a}+}) = 1$ and $\mu_i(q_i^{\mathrm{s}-}) = \mu_i(q_i^{\mathrm{s}+}) = 0$. The proposed activation function is represented on Fig. 1.

B. Scale factor

Contrary to the classical approach, low-level fusion does not add a secondary velocity value to the main task control law. Hence, scaling the avoidance scheme does not consist in compensating for the main task in the velocity space, but rather in finding a set of minimal weights that ensures joint limits are not reached. We will show that with only one joint limit to be avoided at a time, an optimal weight can be analytically computed to be minimal and sufficient to avoid the limit. On the other hand, determining the scale factor for avoiding several limits at the same time does not have an analytical solution, but the optimal weight for one joint is still a good scale factor for the avoidance scheme.

We recall that we assume that $\mathbf{J}_s$ is full rank, which ensures $\mathbf{HJ}$ is also full rank. With the activation matrix from (9), the proposed control law yields:

$$\dot{\mathbf{q}} = -\lambda(\mathbf{HJ})^+\mathbf{H}(\mathbf{s}-\mathbf{s}^*) = -\lambda\begin{bmatrix} \mathbf{J}_s \\ \tilde{\mathbf{H}} \end{bmatrix}^+\mathbf{H}(\mathbf{s}-\mathbf{s}^*) \tag{13}$$

From the pseudo-inverse form $\mathbf{M}^+ = (\mathbf{M}^\top\mathbf{M})^{-1}\mathbf{M}^\top$ that can be applied since $\mathbf{HJ}$ is full rank, the inverse part yields:

$$(\mathbf{J}^\top\mathbf{H}^\top\mathbf{H}\mathbf{J})^{-1} = (\mathbf{J}_s^\top\mathbf{J}_s + \tilde{\mathbf{H}}^2)^{-1} = \frac{\text{adj}(\mathbf{J}_s^\top\mathbf{J}_s + \mathbf{G})}{\det(\mathbf{J}_s^\top\mathbf{J}_s + \mathbf{G})} = \frac{1}{D(\mathbf{g})}\mathbf{N}(\mathbf{g}) \quad (14)$$

where $g_i = h_i^2$ and $\mathbf{G} = \tilde{\mathbf{H}}^2$. $D(\mathbf{g})$ is a strictly positive polynomial of (g_i), being the determinant of a symmetric invertible matrix, and $\mathbf{N}(\mathbf{g})$ is the adjugate matrix of $\mathbf{J}_s^\top\mathbf{J}_s + \mathbf{G}$, that is the matrix of the cofactors. $\mathbf{N}$ is a $(n\times n)$ full rank symmetric matrix.From the adjugate properties, N_{jk} depends neither on g_j, nor on g_k. Also, $N_{ii} > 0$ from the Sylvester's criterion on positive-definite matrices. The control law can then be written:

$$\dot{\mathbf{q}} = -\frac{\lambda}{D}\mathbf{N}\left(\mathbf{J}_s^\top(\mathbf{s}_s - \mathbf{s}_s^*) + \mathbf{G}(\mathbf{q} - \mathbf{q}^*)\right) \quad (15)$$

and each joint velocity yields, up to the scale factor $\frac{\lambda}{D}$:

$$\forall i, \dot{q}_i \propto -\mathbf{N}_i\mathbf{c} - \mathbf{N}_i\mathbf{G}(\mathbf{q} - \mathbf{q}^*) \quad (16)$$

where $\mathbf{c} = \mathbf{J}_s^\top(\mathbf{s}_s - \mathbf{s}_s^*)$ is a vector not depending on $\mathbf{g}$ and $\mathbf{N}_i$ is the i$^{\text{th}}$ row of $\mathbf{N}$, thus not depending on g_i. We denote $\mathbf{N}_s = \mathbf{N}(\mathbf{g} = 0) = \text{adj}(\mathbf{J}_s^\top\mathbf{J}_s)$. We have $-\mathbf{N}_i\mathbf{c} = \dot{q}_{si} + \dot{q}_{ci}$, where $\dot{q}_{si} = -\mathbf{N}_{si}\mathbf{c}$ depends only on the main task and $\dot{q}_{ci}$ depends on $(g_j)_{j\neq i}$. For instance, if $n{=}m{=}p{=}2$, setting:

$$\mathbf{J}_s = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \quad \text{and} \quad \mathbf{c} = \mathbf{J}_s^\top \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} \quad (17)$$

we have:

$$\mathbf{N} = \begin{bmatrix} b^2 + d^2 + g_2 & -ab - cd \\ -ab - cd & a^2 + c^2 + g_1 \end{bmatrix} \quad (18)$$

from which we deduce:

$$\dot{\mathbf{q}}_s = -\begin{bmatrix} (b^2 + d^2)(ae_1 + be_2) - (ab + cd)(ce_1 + de_2) \\ (a^2 + c^2)(ce_1 + de_2) - (ab + cd)(ae_1 + be_2) \end{bmatrix}$$
$$\dot{\mathbf{q}}_c = -\begin{bmatrix} (ae_1 + be_2)g_2 \\ (ce_1 + de_2)g_1 \end{bmatrix} \quad (19)$$

We can check that $N_{ii} > 0$ and that $\dot{q}_{c1}$ (resp. $\dot{q}_{c2}$) depends only on g_2 (resp. g_1). Coming back to the general case, two cases can be distinguished: ,

- Only joint i is in its unsafe area: in that case, $\dot{q}_i$ depends linearly on g_i:

$$\dot{q}_i \propto -N_i\mathbf{c} - N_{ii}g_i(q_i - q_i^*) = \dot{q}_i^0 - N_{ii}g_i(q_i - q_i^*) \quad (20)$$

This leads to two configurations C1 and C2:

– the main task is reaching the limit (C1): if $\dot{q}_{si}$ and $(q_i - q_i^*)$ have the same sign, the robot is going towards a joint limit. As $N_{ii} > 0$ there exists a positive g_i such that $\dot{q}_i$ is null.
– the main task is avoiding the limit (C2): if $\dot{q}_{si}$ and $(q_i - q_i^*)$ do not have the same sign, the robot is moving away from the limit: self-avoidance occurs. The avoidance scheme can be ignored.

From this observation, assuming we want the joint to stop when $q_i = q_i^{\text{a}}$, the optimal value can be computed analytically from (20):

$$h_i^{\text{opt}} = \begin{cases} h_i^{\min} = \sqrt{\frac{\dot{q}_{si}}{N_{ii}(q_i^{\text{a}} - q_i^{\text{s}})}} & \text{if } \frac{\dot{q}_{si}}{(q_i^{\text{a}} - q_i^{\text{s}})} > 0 \quad \text{(C1)} \\ 0 & \text{else} \quad \text{(C2)} \end{cases} \quad (21)$$

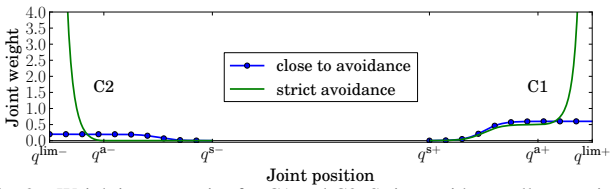


Fig. 2. Weighting strategies for C1 and C2. Strict avoidance allows weights to increase instead of keeping equal to the optimal one. Configuration C1 is represented for the upper bound ($h^{\text{opt}} = 0.5$), while C2 is represented for the lower bound. ϵ is set to 0.2.

where $h_i^{\text{opt}} = 0$ corresponds to the configurations where the main task itself is avoiding the limit.

- $p > 1$ joints are in their unsafe area: (15) is a system of p polynomials of degree p:

$$\forall i \in S = \{i_1, \ldots, i_p\}, \dot{q}_i \propto -\mathbf{N}_i\mathbf{c} - \sum_{j\in S} N_{ij}g_j(q_j - q_j^*) \quad (22)$$

Even if each polynomial is actually linear for each g_i, such a system cannot be solved analytically. At the joint position $q_i = q_i^{\text{a}}$, the proposed weight (21) leads to the following joint velocity, once again up to the scale factor $\frac{\lambda}{D}$:

$$\forall i \in S, \dot{q}_i \propto \dot{q}_i^c - \sum_{j\neq i} N_{ij}g_j(q_j - q_j^*) \quad (23)$$

The expression being of unknown sign, it is not possible to ensure the joint limit is avoided in the case of several unsafe joints. However, a set of weights ensuring the limits are not reached always exists, hence two strategies are proposed.

a) Close to avoidance: Increasing the optimal scale factor makes it possible to design a general scaling strategy:

$$h_i^{\text{opt}} = \begin{cases} (1 + \epsilon)h_i^{\min} & \text{if } h_i^{\min} > 0 \quad \text{(C1)} \\ \epsilon & \text{else} \quad \text{(C2)} \end{cases} \quad (24)$$

where a high $\epsilon > 0$ makes it more likely to avoid several joint limits in critical positions, but it is never ensured.

b) Strict avoidance: In the gradient projection methods, if the secondary task is in the null space of the projection operator then it is possible that a joint limit is reached. On the opposite, the proposed scheme can ensure no limit is ever reached, by having sufficiently high weights when coming close to the limit ($q_i > q_i^{\text{a}}$). A strict avoidance scheme can be designed from the previous one:

$$h_i^{\text{opt}} = \begin{cases} (1 + h_i^\infty(q_i))h_i^{\min} & \text{if } h_i^{\min} > 0 \quad \text{(C1)} \\ h_i^\infty(q_i) & \text{else} \quad \text{(C2)} \end{cases} \quad (25)$$

where $h_i^\infty(q_i)$ is an adaptative gain such that $h_i^\infty(q_i^{\text{a}}) = 0$ and $\lim_{q_i \to q_i^{\text{lim}}} h_i^\infty(q_i) = \infty$, for example:

$$h_i^\infty(q_i) = \begin{cases} \frac{q_i^{\text{a}-} - q_i}{q_i - q_i^-} & \text{if } q_i < q_i^{\text{a}-} \\ \frac{q_i - q^{\text{a}+}}{q^+ - q_i} & \text{if } q_i > q_i^{\text{a}+} \\ 0 & \text{else} \end{cases} \quad (26)$$

The drawback of the strict avoidance scheme is that the main task may be neglected if it is the only way to avoid a limit. Yet, the behavior is better than stopping the joint since the fusion scheme always takes into account the coupling with the main task. When avoiding several limits at the same time, configuration C2 for joint i corresponds to the configurations where the main task is avoiding the joint limit, but some

Algorithm 1 Escape from local minima

Require: $\alpha = 1$, $\alpha^+ > 1$, $\alpha^- < 1$
  while performing servoing **do**
    measure end-effector velocity $\mathbf{v}_e$
    compute optimal weighting matrix $\tilde{\mathbf{H}}$
    if $\|\mathbf{v}_e\| < v_0$ and $\tilde{\mathbf{H}} \neq 0$ **then**
      $\alpha \leftarrow \alpha^+ \alpha$
    else
      $\alpha \leftarrow 1 + \alpha^-(\alpha - 1)$
    end if
    apply control law using weighting matrix $\alpha\tilde{\mathbf{H}}$
  end while

joint velocity may be created by the avoidance of the other joint limits (see (23)). A compensation of the main task is thus meaningless and h^{opt} only depends on the adaptative gain h^{∞}. Fig. 2 represents the proposed weighting strategies including the activation function and the scale factor.

C. Additional strategies

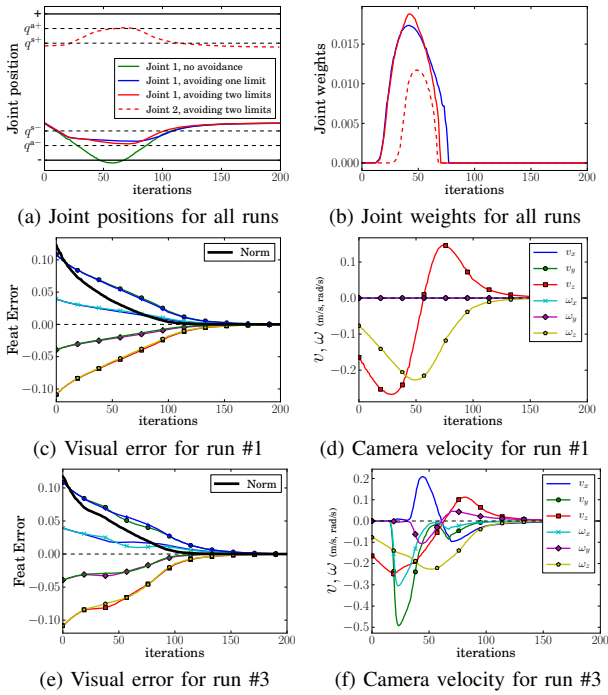
This section exposes additional strategies that address the cases of unsafe desired position and local minima.

1) Unsafe desired position: If the desired position is unsafe, a full-rank main task cannot be perfectly performed as the non-null joint error will prevent the main task error from reaching zero. In that case, he activation scheme can be progressively ignored wrt. the main task convergence. To do so, we introduce a progress parameter $\xi(\|\mathbf{e_s}\|)$ smoothly making the activation weights null when the main task gets close to completion. In that case, $h_i = \xi(\|\mathbf{e_s}\|)\mu_i(q_i)h_i^{\text{opt}}$ where:

$$\xi(\|\mathbf{e_s}\|) = \begin{cases} 0 & \text{if } \|\mathbf{e_s}\| \leq e_0 \\ 1 & \text{if } \|\mathbf{e_s}\| \geq e_1 \\ \frac{1}{2}\left(1+\tanh\left(\frac{1}{e_1-\|\mathbf{e_s}\|} - \frac{1}{\|\mathbf{e_s}\|-e_0}\right)\right) & \text{else} \end{cases} \tag{27}$$

where e_0 and e_1 are defined so that the avoidance scheme is totally ignored when the main task is close to completion, that is $\|\mathbf{e_s}\| < e_0$.

2) Avoiding local minima: Task functions may be subject to local minima when the number of features is greater than the available DOFs, which is the case here since we consider a full-rank main task and add unsafe joints in the feature set. A local minimum is easily detected as it is necessarily a configuration where the end-effector velocity is almost null, while at least one joint weight is non-null. Indeed, the global minimum always corresponds to null weights, either because it is in the safe area or thanks to the progress parameter. A popular optimization method to escape from local minima is simulated annealing [2], [7] where the minimized criterion is allowed to increase during the scheme under a probabilistic model. In our case, we take advantage of the particular form of the weighting matrix $\mathbf{H}$ (see (9)), where the weights for the sensor features are always equal to 1 while the joint weights are varying. The proposed strategy is described in Algorithm 1: as the end-effector velocity is approaching 0 while $\tilde{\mathbf{H}} \neq 0$, the computed joint weights are artificially

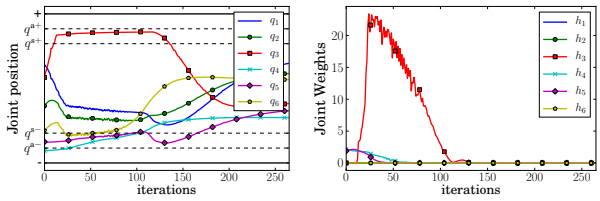


(a) Joint positions for all runs (b) Joint weights for all runs

(c) Visual error for run #1 (d) Camera velocity for run #1

(e) Visual error for run #3 (f) Camera velocity for run #3

Fig. 3. Joint positions (a) and corresponding weights (c) in a task with no avoidance (green), one limit avoidance (blue) and two limits avoidance (red). Joint physical limits are in plain lines, q^{a} and q^{s} are in dotted lines. Visual error (left) and camera velocity (right) are represented for run #1 (middle) and run #3 (bottom).



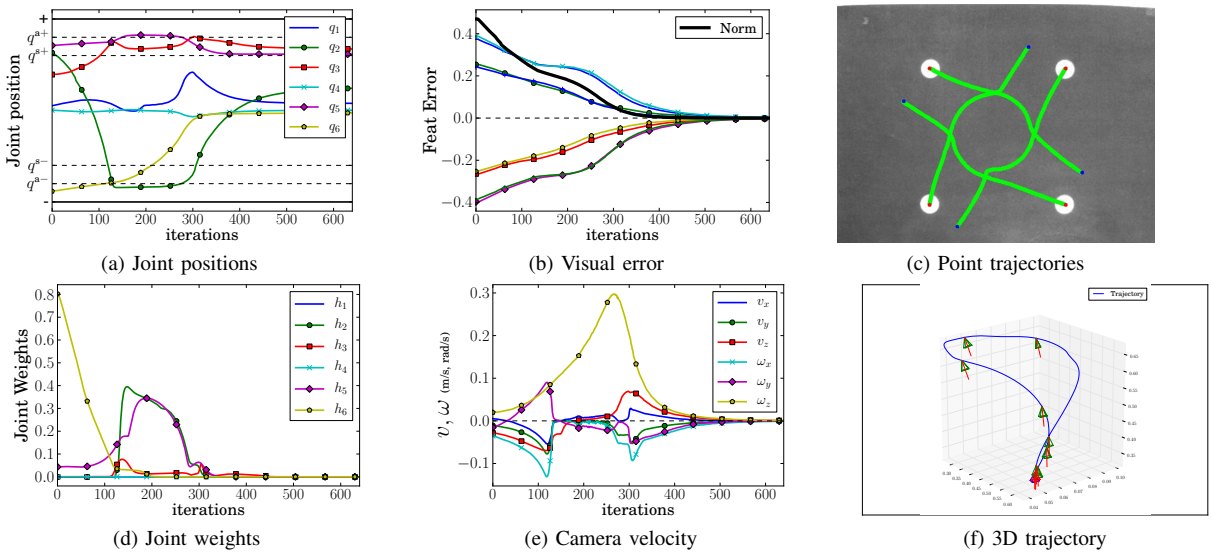
(a) Joint positions (b) Joint weights

Fig. 4. Positions and weights for the robot joints while escaping from a local minimum. Oscillations in the weight induce a slight oscillation of the robot end-effector.

increased by a multiplicative factor α, allowing non-optimal motion in terms of the sensor-based task. During normal convergence, α is slowly set back to 1. However, this strategy may induce oscillations. We will see in the next section that these oscillations are very small in practice.

IV. Simulation Results

Simulations are carried for a 6-DOFs Gantry robot, the control law are implemented using ViSP software [12]. Main task is a visual servoing (VS) using 4 image points [4]. We use the cartesian coordinates of the current image points as visual features. It is well-known that if the initial pose makes it necessary for the camera to perform a large rotation around its optical axis, this choice of visual features induces a large translational backward motion, hence potentially reaching joint limits. First we expose the basic behavior of the avoidance scheme, then the results of exhaustive simulations are presented. In the simulations, the robot desired pose belongs to the safe area for each joint.



(a) Joint positions (b) Visual error (c) Point trajectories

(d) Joint weights (e) Camera velocity (f) 3D trajectory

Fig. 5. Joint avoidance for a pure rotation. Visual error is smoothly decreasing during the task (b), the resulting trajectory is a backward motion followed by a forward motion (f).

A. Behavior of joints and weights

The behavior induced by the proposed control law is exposed in Fig. 3. Joint positions are represented in normalized scale on Fig. 3a, the key positions q^{a} and q^{s} are in dotted lines. The initial and desired pose differ from a $110\,\mathrm{deg}$ rotation around the optical axis, inducing a large motion on joint 1. The simulation allows defining the robot joint limits such that only joints 1 and 2 may reach their limits for this task. *Close to avoidance* scheme is used. Three runs are performed:

1) No avoidance (green curve): Fig. 3d shows that as expected, the camera moves only along v_z and ω_z, allowing a decoupled decrease of the error as seen in Fig. 3c. However, joint 1 passes the limit q^{-}.

2) Avoiding one limit (blue curve): the joint limits are defined such that only joint 1 may reach its limit. This time, the joint does not reach the limit but stops before $q^{\mathrm{a}-}$. Furthermore, although the joint is still in the unsafe area at iteration 75, one can notice in Fig. 3b that the corresponding weight is null. This reveals the avoidance scheme change from configuration C1 to configuration C2, with the VS task making the joint going towards the safe area.

3) Avoiding two limits (red curves): here the limits are defined such that joint 2 enters the unsafe area. Here, Fig. 3a shows joint 1 nearly reaches $q^{\mathrm{a}-}$ while joint 2 passes $q^{\mathrm{a}+}$, illustrating the coupling between the two. As in run #2, both weights are null at iteration 75, before the joints go back in the safe area. Once more, the avoidance configuration has changed from C1 to C2. Fig. 3e shows the visual features error for run #3. We can see the error is decreasing even during the avoidance (iterations 10-80). The corresponding camera velocity is represented on Fig. 3f. The initial motion is a pure translation and rotation along the z-axis as in Fig. 3d, then other velocity components appear when the avoidance is performed, in order to keep the visual features error decreasing. Finally, we can notice the joint weights are very small (less than 0.02), yet they are sufficient to avoid the limits. A small weight barely disturbs the sensor-based task, as can be observed by comparing Fig. 3c and Fig. 3e.

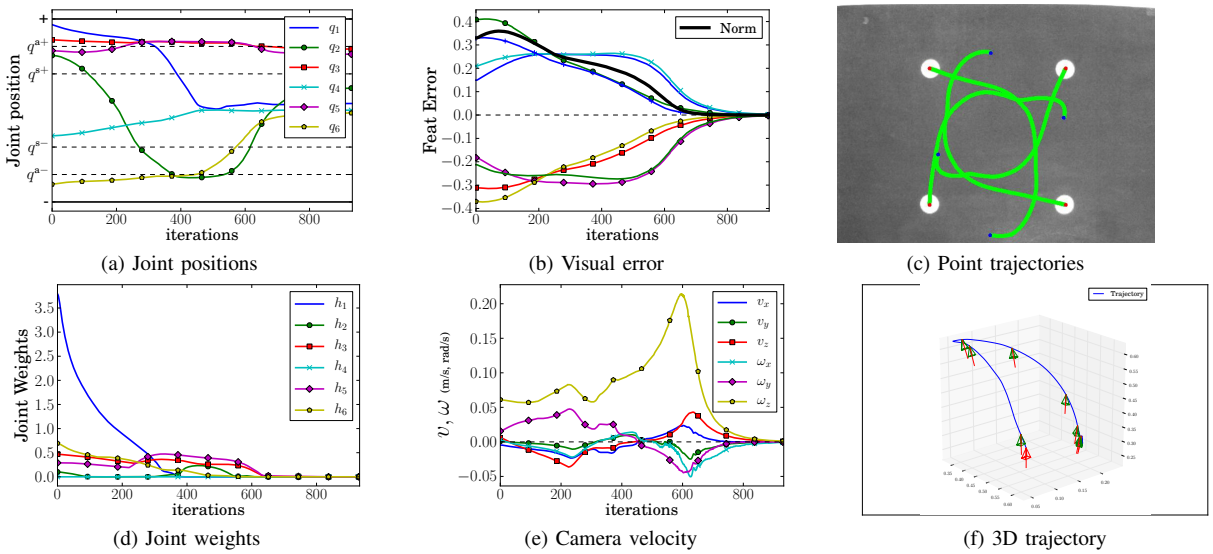
B. Exhaustive tests

In order to study the *strict avoidance* behavior, exhaustive runs have been carried from 690 different initial positions assuming the image plane is infinite. The initial positions have been computed using a space-filling design [7] that ensures all the joint space is explored and the 4 points are in front of the camera. In 636 cases the system converges directly to the global minimum. The other 54 cases make the system converge to a local minimum, that is always escaped from with the strategy exposed in Section III-C.2. Fig. 4 shows such a case: the weight of joint 3 varies following the proposed strategy, inducing very small oscillations on the other joints as seen in Fig. 4a.

V. Experiments

Experiments are carried on a 6-DOFs robot arm Adept Viper s850 (see the video accompanying this paper). The desired position is unsafe for some joints, hence the progress parameter defined in (27) is used. First, the initial position corresponds to a pure rotation around the z-axis. A second experiment is done with an initial position necessitating a more complex motion. A general view of both experiments can be found on the video accompanying this paper. *Strict avoidance* scheme is used.

1) Pure rotation: Here the initial and desired pose only differ from a $150\,\mathrm{deg}$ rotation around the z-axis. As in Section IV-A, the expected camera motion is a pure translation and rotation along the z-axis. However, as the initial pose is not safe for joints 2, 5 and 6, the avoidance scheme induces a different motion. Joint 6 (yellow curve on Fig. 5a) being near to the lower limit, the corresponding weight is high (0.8) at the beginning then quickly decreasing as the joint goes away from the limit. The visual error is decreasing, leaving



(a) Joint positions (b) Visual error (c) Point trajectories

(d) Joint weights (e) Camera velocity (f) 3D trajectory

Fig. 6. Joint avoidance for a complex motion. Visual error increases before converging to 0 (b), making the points have smooth trajectories in the image (c). Initial position is unsafe for 5 joints (a) and initial weight is very high for joint 1 (d).

smooth trajectories for the image points as seen on Fig. 5c. The large rotation around the z-axis that occurs at iterations 200-300 draws arcs of a circle in the image, then the points have a straight trajectory towards their desired position. As expected, the 3D trajectory (Fig. 5f) is a backward motion that adapts to the joint limits before converging to the desired position. The progress parameter makes it possible to reach the desired position, although it is unsafe for joints 2 and 3.

2) Complex motion: A new initial position is defined, such that a large rotation around the z-axis is still involved, together with other displacements. Here the initial position is unsafe for all joints except joint 4. Joint 1 starts very near to its upper limit, inducing a very high weight (3.7). As a consequence, and as many joints are in an unsafe position, the visual error increases during the first 100 iterations (Fig. 6b). Once the visual error starts to decrease, joints 3 and 5 stay around q^{a+} without any oscillation, while joint 2 quickly reaches q^{a-} without approaching the lower limit, with a corresponding weight of about 0.2. After iteration 630, all weights are null and the system converges with a classical IBVS behavior: exponential decoupled decrease of the visual error, straight line trajectories in the image. Once more, the 3D trajectory starts with a large backward motion (Fig. 6f) , before heading to the desired position.

VI. Conclusions

The framework of multi-sensor data fusion has been applied to joint limits avoidance. The measurement of the robot joint positions are considered as additional features, and treated with a particular varying-feature-set scheme that ensures the continuity of the control law. This leads to an optimal weighting in the case of single joint avoidance, that can be also applied when several joints are in an unsafe position. We have considered the configurations where the desired position is unsafe. A strategy has also been exposed to escape from local minima. Simulations and experiments illustrate the validity of the proposed approach.

References

[1] G. Antonelli, G. Indiveri, and S. Chiaverini, "Prioritized closed-loop inverse kinematic algorithms for redundant robotic systems with velocity saturations," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. IEEE, 2009.

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

[3] T. Chan and R. Dubey, "A weighted least-norm solution based scheme for avoiding joint limits for redundant joint manipulators," *IEEE Trans. Robot. Autom.*, vol. 11, no. 2, 1995.

[4] F. Chaumette and S. Hutchinson, "Visual servo control. i. basic approaches," *IEEE Robot. Autom. Mag.*, vol. 13, no. 4, 2006.

[5] C. Cheah, D. Wang, and Y. Sun, "Region-reaching control of robots," *IEEE Trans. Robot.*, vol. 23, no. 6, 2007.

[6] A. Comport, E. Marchand, and F. Chaumette, "Statistically robust 2-d visual servoing," *IEEE Trans. Robot.*, vol. 22, no. 2, 2006.

[7] K. Fang, R. Li, and A. Sudjianto, *Design and modeling for computer experiments*. CRC Press, 2006.

[8] 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. Robot.*, vol. 21, no. 6, 2005.

[9] O. Kermorgant and F. Chaumette, "Multi-sensor data fusion in sensor-based control: application to multi-camera visual servoing," in *IEEE Int. Conf. on Robotics and Automation*, Shanghai, China, 2011.

[10] N. Mansard and F. Chaumette, "Directional redundancy: a new approach of the redundancy formalism," in *44th IEEE Conf. on Decision and Control*. IEEE, 2006.

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

[12] 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.

[13] M. Marey and F. Chaumette, "New strategies for avoiding robot joint limits: Application to visual servoing using a large projection operator," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Taipei, Taiwan, October 2010.

[14] 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, 1995.

[15] B. Siciliano and J. Slotine, "A general framework for managing multiple tasks in highly redundant robotic systems," in *IEEE Fifth Int. Conf. on Advanced Robotics*, 1991.

[16] T. Yoshikawa, "Basic optimization methods of redundant manipulators," *Laboratory Robotics and Automation*, vol. 8, no. 1, 1996.

[17] H. Zghal, R. Dubey, and J. Euler, "Efficient gradient projection optimization for manipulators with multiple degrees of redundancy," in *IEEE Int. Conf. on Robotics and Automation*, 2002.