2008 IEEE International Conference on
Robotics and Automation
Pasadena, CA, USA, May 19-23, 2008

On the Efficient Second Order Minimization and Image-Based Visual Servoing

Omar Tahri and Youcef Mezouar

Abstract— **This paper deals with the efficient second order minimization (ESM) and the image-based visual servoing schemes. In other word, it deals with the minimization based on the pseudo-inverses of the mean of the Jacobians or on the mean of Jacobian Pseudo-inverses. Chronologically, it has been noted in [16] that the (ESM) improves generally the system behavior compared to the case where only the simple Jacobian Pseudo-inverses is used. Subsequently, a mathematical explanation has been given in [11]. In this paper, the proofs given in [11] are considered to deal with their validity. It will be shown that there is a limitation to the the validity of this method and some precautions should be taken, for adequate application of it. In other words, we will show that the use of ESM does not necessary ensures a better system behavior, especially in the cases where large rotational motions are considered.**

I. INTRODUCTION

Visual servoing techniques are very effective since they close the control loop over the vision sensor. This yields a high robustness to disturbances as well as to calibration errors. Several kinds of visual servoing can be distinguished, according to the space where the visual features were defined. In position-based visual servo (PBVS) [19], the features are defined in the 3D space. The control scheme using PBVS ensures a nice decoupling between the degrees of freedom (dofs). For this reason, adequate 3D trajectories can be obtained, such as a geodesic for the orientation and a straight line for the translation. However, position-based visual servoing may suffer from potential instabilities due to image noise [4]. On the opposite, in image-based visual servo (IBVS) [8], the visual servo is performed in the image. A compromise can be obtained by combining features in image and partial 3D data [12].

In 2D visual servoing, the behavior of the features in the image is generally satisfactory. On the other hand, the robot trajectory in 3D space is quite unpredictable and may be really unsatisfactory for large rotational displacements [4]. In few words, we recall that the time variation $\dot{\mathbf{s}}$ of the visual features $\mathbf{s}$ can be expressed linearly with respect to the relative camera-object kinematics screw $\mathbf{v}$:

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

where $\mathbf{L}_{\mathbf{s}}$ is the interaction matrix related to $\mathbf{s}$ [8]. The control scheme is usually designed to reach an exponential decoupled decrease of the visual features to their desired value $\mathbf{s}^*$, from which we deduce if we consider an eye-in-hand system observing a static object:

$$\mathbf{v}_{\mathbf{c}} = -\lambda \widehat{\mathbf{L}_{\mathbf{s}}}^{+}(\mathbf{s} - \mathbf{s}^*) \tag{2}$$

where $\widehat{\mathbf{L}_{\mathbf{s}}}$ is a model or an approximation of $\mathbf{L}_{\mathbf{s}}$, $\widehat{\mathbf{L}_{\mathbf{s}}}^{+}$ the pseudo-inverse of $\widehat{\mathbf{L}_{\mathbf{s}}}$, λ a positive gain tuning the time to convergence, and $\mathbf{v}_{\mathbf{c}}$ the camera velocity sent to the low-level robot controller. If the initial error is large, such control may produce an erratic behavior: convergence to local minima, inappropriate camera trajectory due to strong coupling. In fact, the difference of behaviors in image space and 3D space is due to the non linearities in the interaction matrix. In principle, an exponential decoupled decrease will be obtained simultaneously on the visual features and on the camera velocity (that would give a perfect behavior) only if the interaction matrix is constant.

To address this issue, a first approach consists of selecting features sharing good properties. In this way, recently, the analytical form of the interaction matrix related to any image moments corresponding to planar objects has been computed. This makes possible to consider planar objects of any shape [5], [17]. If a collection of points is measured in the image, moments can also be used [17]. In both cases, moments allow the use of intuitive geometrical features, such as the center of gravity or the orientation of an object. By selecting an adequate combination of moments, it is then possible to determine partitioned systems with good decoupling and linearizing properties [5], [17]. For instance, using such features, the control of the the three translational degrees of freedom is completely partitioned. Furthermore, the block of the interaction matrix corresponding to the translational velocity is a constant block diagonal. This has improved widely the 3D behavior of the system. Another solution is to use a path planing step jointly with the servoing one [13], [1], [14], [6]. In such approach, the basic idea is to sample the initial errors in order to ensure that the error at each iteration remains small.

Another way to improve IBVS behavior consists on taking into account the strong non-linearities in the relation from the image to the work space. Indeed, the function mapping 3D features to their projection in the image is neither linear nor invertible. The control law (2) was developed using a first-order Taylor expansion of the projection function to estimate locally the variation of the camera 3D pose from the features variations. Lapreste et al in [10] presents a method for estimating the control matrix in visual servoing using approximation up to the second order of the projection function.

O. Tahri and Y. Mezouar are with LASMEA, Université Blaise Pasacal, 63177 AUBIERE, France firstname.lastname@lasmea.univ-bpclermont.fr

The proposed method is based on Hessian approximation. The main drawbacks is that this method introduces a lot of supplementary parameters. The work we are interested in is that proposed by Malis in [11]. In the latter, a "second order method" based only on first derivative and thus without Hessian estimation is proposed to enhance the IBVS. The same idea was extended to image tracking of planar object in [2], [3], for instance. In this paper, the validity of such approaches is discussed. However, the discussion will only focus on image-based visual servoing application rather than tracking one. The tracking application is not concerned, since a small displacement are considered in general. The methods given in [2], [3] are still valid in general and ensure better results in term of convergence rate and percentage. In the following, it will be shown that the use of such method in IBVS does not ensure necessary a better behavior, worse it could be a cause of unstable system control. Furthermore, corrected formulas of these control will be proposed and validated.

II. Mathematical background of the "efficient second order minimization"

In this section, the mathematical background of the ESM given in [11] are recalled.

A. Starting point

Instead of the first order Taylor series, the ESM is based on the second-order Taylor series of $\mathbf{s}(\mathbf{x})$:

$$\mathbf{\Delta s} = -\mathbf{J}(\mathbf{x_1})\mathbf{\Delta x} + \frac{1}{2}\mathbf{M}(\mathbf{x_1}, \mathbf{\Delta x})\mathbf{\Delta x} + \mathbf{0_{s2}}(\mathbf{\Delta x^3}) \quad (3)$$

$$\mathbf{\Delta s} = -\mathbf{J}(\mathbf{x_2})\mathbf{\Delta x} - \frac{1}{2}\mathbf{M}(\mathbf{x_2}, \mathbf{\Delta x})\mathbf{\Delta x} + \mathbf{0_{s1}}(\mathbf{\Delta x^3}) \quad (4)$$

where $\mathbf{x_1}$ and $\mathbf{x_2}$ define the camera frame positions, $\mathbf{J}(\mathbf{x_1})$ and $\mathbf{J}(\mathbf{x_2})$ are the Jacobien matrices (n by 6 matrices), $\mathbf{0_{s1}}$ and $\mathbf{0_{s2}}$ are the reminders, $\mathbf{M}(\mathbf{x1}, \mathbf{\Delta x})$ and $\mathbf{M}(\mathbf{x2}, \mathbf{\Delta x})$ are matrices containing all the n Hessian matrices of the $(n \times 1)$ vector function $\mathbf{s}(\mathbf{x})$:

$$\mathbf{M}(\mathbf{x1}, \mathbf{\Delta x}) = (\mathbf{\Delta x^T H_1}(\mathbf{x1}),\ \mathbf{\Delta x^T H_2}(\mathbf{x1}), \ldots,\ \mathbf{\Delta x^T H_n}(\mathbf{x1}))$$
$$\mathbf{M}(\mathbf{x2}, \mathbf{\Delta x}) = (\mathbf{\Delta x^T H_1}(\mathbf{x2}),\ \mathbf{\Delta x^T H_2}(\mathbf{x2}), \ldots,\ \mathbf{\Delta x^T H_n}(\mathbf{x2}))$$

Starting from this, [11] designed two control schemes called respectively Mean of Jacobian Pseudo-inverses (MJP) and Pseudo-inverse of the mean of the Jacobians (PMJ).

B. Mean of Jacobian Pseudo-inverses

Multiplying both sides of equation (3) by $\mathbf{J}^+(\mathbf{x_1})$ and both sides of equation (4) by $\mathbf{J}^+(\mathbf{x_2})$ we obtain:

$$\mathbf{\Delta x} = -\mathbf{J}^+(\mathbf{x_1})\mathbf{\Delta s} + \frac{1}{2}\mathbf{J}^+(\mathbf{x_1})\mathbf{M}(\mathbf{x_1}, \mathbf{\Delta x})\mathbf{\Delta x} + \mathbf{0'_{s2}}(\mathbf{\Delta x^3}) \quad (5)$$

$$\mathbf{\Delta x} = -\mathbf{J}^+(\mathbf{x_2})\mathbf{\Delta s} - \frac{1}{2}\mathbf{J}^+(\mathbf{x_2})\mathbf{M}(\mathbf{x_2}, \mathbf{\Delta x})\mathbf{\Delta x} + \mathbf{0'_{s1}}(\mathbf{\Delta x^3}) \quad (6)$$

Let the matrix $\mathbf{J}^+(\mathbf{y})\mathbf{M}(\mathbf{y}, \mathbf{\Delta x})$ be a function of $\mathbf{y}$. Consider its first-order Taylor series about $\mathbf{x_1}$, evaluated at $\mathbf{x_2}$:

$$\mathbf{J}^+(\mathbf{x_2})\mathbf{M}(\mathbf{x_2}, \mathbf{\Delta x}) = \mathbf{J}^+(\mathbf{x_1})\mathbf{M}(\mathbf{x1}, \mathbf{\Delta x}) + (\mathbf{0_{J^+}})(\mathbf{\Delta x^2}) \quad (7)$$

where the reminder $\mathbf{O_{J^+}}$ is quadratic in $\mathbf{\Delta x}$. Computing the mean of equations (5) and (6), and plugging equation (7) into the mean, we obtain:

$$\mathbf{\Delta x} \approx -\frac{1}{2}(\mathbf{J}^+(\mathbf{x_2}) + \mathbf{J}^+(\mathbf{x_1}))\mathbf{\Delta s} + 0_{MJP}(\mathbf{\Delta x}^3) \quad (8)$$

where:

$$\mathbf{O_{MJP}}(\mathbf{\Delta x^3}) = \mathbf{O'}(\mathbf{s_1})(\mathbf{\Delta x^3}) + \mathbf{O'}(\mathbf{s_2})(\mathbf{\Delta x^3}) + \mathbf{O_{J^+}}(\mathbf{\Delta x^2})$$

is the total remainder which is cubic in $\mathbf{\Delta x}$. In conclusion, the mean of first-order approximations of the displacement is a second-order approximation of the displacement:

$$\mathbf{\Delta x} \approx \frac{1}{2}(\mathbf{J}^+(\mathbf{x1}) + \mathbf{J}^+(\mathbf{x2}))\mathbf{\Delta s} \quad (9)$$

C. Pseudo-inverse of the mean of the Jacobians

Consider the second-order Taylor series of the Jacobian $\mathbf{J}(\mathbf{x})$ about $\mathbf{x_2}$ and evaluated at $\mathbf{x_1}$:

$$\mathbf{J}(\mathbf{x_1}) = \mathbf{J}(\mathbf{x_2}) + \mathbf{M}(\mathbf{x_2}, \mathbf{\Delta x}) + \mathbf{O_J}(\mathbf{\Delta x^2}) \quad (10)$$

where O_J is the remainder. This formula provides an estimation to the second-order of matrix $\mathbf{M}(\mathbf{x2}, \mathbf{\Delta x})$:

$$\mathbf{M}(\mathbf{x2}, \mathbf{\Delta x}) = \mathbf{J}(\mathbf{x_1}) - \mathbf{J}(\mathbf{x_2}) - \mathbf{O_J}(\mathbf{\Delta x}^2)\mathbf{\Delta x}) \quad (11)$$

Plugging this equation into equation (4) we obtain:

$$\mathbf{\Delta s} = \frac{1}{2}(\mathbf{J(x1)} + \mathbf{J(x2)})\mathbf{\Delta x} + \mathbf{0_{PMJ}}(\mathbf{\Delta x}^3) \quad (12)$$

where $(\mathbf{0_{PMJ}}\mathbf{\Delta x}^3) = \mathbf{O_{s_2}}(\mathbf{\Delta x}^3)) + O_J(\mathbf{\Delta x}^2)\mathbf{\Delta x}$ is the total remainder which is cubic in $\mathbf{\Delta x}$. As a consequence, a second-order approximation of $\mathbf{s}(\mathbf{x})$ is again obtained using only first derivatives:

$$\mathbf{\Delta s} \approx \frac{1}{2}(\mathbf{J}(\mathbf{x_1}) + \mathbf{J}(\mathbf{x_2}))\mathbf{\Delta x} \quad (13)$$

The displacement can be obtained by computing the pseudo-inverse of the mean of the Jacobians:

$$\mathbf{\Delta x} \approx 2(\mathbf{J}(\mathbf{x_1}) + \mathbf{J}(\mathbf{x_2}))^+\mathbf{\Delta s} \quad (14)$$

The whole results recalled in the two above paragraphs are valid from a mathematical point view. However, their application in the context of visual servoing is not straightforward and some precautions should be taken. In fact, some special properties of visual servoing have not been taken into account. More details will be given in the next paragraph.

D. From second order minimization to IBVS

The first problem to be considered in the "proofs" given in the above paragraphs is the definition of $\mathbf{\Delta x}$. In visual servoing, naturally, $\mathbf{\Delta x}$ is nothing but a displacement between two camera configurations with respect to an object. In [11] it has been defined as follows:

$$\mathbf{\Delta x} \approx \mathbf{v}\Delta t \quad (15)$$

Let $\mathbf{v_d}$ be the velocity vector computed using (2) to move the desired camera frame (referenced by $\mathbf{x_2}$) to the current one (referenced by $\mathbf{x_1}$). Let also $\mathbf{v_c}$ be the velocity vector

to move the current camera frame (referenced by $\mathbf{x_2}$) to its desired position (referenced by $\mathbf{x_1}$) using (2). In other words, $\mathbf{v_d}$ and $\mathbf{v_c}$ are defined as follow:

$$\mathbf{v_d} = \lambda {\mathbf{L_{s^*}}}^{+}(\mathbf{s} - \mathbf{s}^*) \tag{16}$$

$$\mathbf{v_c} = -\lambda {\mathbf{L_s}}^{+}(\mathbf{s} - \mathbf{s}^*) \tag{17}$$

From the above definitions, it has been considered that $\mathbf{L_{s^*}}$ and $\mathbf{L_s}$ are equal to $\mathbf{J(x_2)}$ and $\mathbf{J(x_1)}$ respectively. This leads straightforwardly to the two following control laws:

$$\mathbf{v_{MJP}} = -\frac{1}{2}\lambda({\mathbf{L_s}}^{+} + {\mathbf{L_{s^*}}}^{+})(\mathbf{s} - \mathbf{s}^*) \tag{18}$$

$$\mathbf{v_{PJM}} = -2\lambda(\mathbf{L_s} + \mathbf{L_{s^*}})^{+}(\mathbf{s} - \mathbf{s}^*) \tag{19}$$

In fact the definition of $\mathbf{\Delta x}$ used to develop the two above control laws is not consistent with (3) and (4). Indeed, in (3) and (4), it is assumed that the displacements from the position $\mathbf{x_1}$ to the the position $\mathbf{x_2}$ and the displacement from $\mathbf{x_1}$ to $\mathbf{x_2}$ have the same value $\|\mathbf{\Delta x}\|$, but a different sign, which leads to (15):

$$\mathbf{v_d}dt \approx -\mathbf{v_c}dt \tag{20}$$

If a rotational motion is considered, the orientation of the desired and the current frames are different. Since $\mathbf{v_d}$ and $\mathbf{v_c}$ are respectively expressed in the current and the desired frames, (20) is in general not valid. Indeed, although $\mathbf{v_d}$ and $\mathbf{v_c}$ represent the same displacement, they have not got the same orientation. Thus the motion between the two camera positions need to be taken into account. In other word, $\mathbf{v_d}$ and $\mathbf{v_c}$ have to be expressed in the current frame and applied to the latter. This means we have to use $-\mathbf{T}\,\mathbf{v_d}$ instead of $\mathbf{v_d}$. Where, $\mathbf{T}$ is the tensor transformation matrix from the desired frame to the current one:

$$\mathbf{T} = \begin{pmatrix} {}^{\mathbf{c}}\mathbf{R_d} & {}^{\mathbf{c}}\mathbf{R_d}[\mathbf{t}]_\times \\ \mathbf{0_{3\times 3}} & {}^{\mathbf{c}}\mathbf{R_d} \end{pmatrix} \tag{21}$$

${}^{\mathbf{c}}\mathbf{R_d}$ is the rotation matrix between the current and the desired position, $\mathbf{t}$ is the translation vector, and $[]_\times$ is the skew-symmetric matrix associated to the vector cross-product. Finally the following appropriate second order minimization can be obtained:

$$\mathbf{v_{MJP}} = -\frac{1}{2}\lambda({\mathbf{L_s}}^{+} + \mathbf{T}{\mathbf{L}}^{+}_{\mathbf{s^*}})(\mathbf{s} - \mathbf{s}^*) \tag{22}$$

$$\mathbf{v_{PJM}} = -2\lambda(\mathbf{L_s} + \mathbf{L_{s^*}}\,\mathbf{T^{-1}})^{+}(\mathbf{s} - \mathbf{s}^*) \tag{23}$$

Note that (18) and (19) are nothing else that a rough approximation of (22) and (23). Where, $\mathbf{T}$ is approximated by the identity matrix $\mathbf{I_6}$. This could be valid only if the translational degrees of freedom are considered. The change of frame presented here has already been taken into account in [3] but for tracking purpose using Lie Algebra and for small displacements.

III. Validations

In this section, six features based on moment computed from a set of points will be used for simulation purpose.

A. Features vector

As a example of features in image, the image moments computed from a set of points are used. In this way, to control the three rotational degrees of freedom the features proposed in [17] are exploited. They are defined as follow:

$$\mathbf{s} = \begin{pmatrix} r_1 \\ r_2 \\ \alpha \end{pmatrix} \tag{24}$$

where $\alpha = \frac{1}{2}\arctan(\frac{2\mu_{11}}{\mu_{20}-\mu_{02}})$ determines the orientation of the principal axis of the ellipsoid defined by the central moment computed from the set of point, r_i, r_j are two invariants obtained by combining three kinds of moment invariants: invariant to translation, to 2D rotation and to scale. For instance, r_i, r_j can be chosen as:

$$r_1 = I_{n_1}/I_{n_3}\ ,\ r_2 = I_{n_2}/I_{n_3} \tag{25}$$

with:

$$\begin{cases} I_{n_1} = (\mu_{50}+2\mu_{32}+\mu_{14})^2 + (\mu_{05}+2\mu_{23}+\mu_{41})^2 \\ I_{n_2} = (\mu_{50}-2\mu_{32}-3\mu_{14})^2 + (\mu_{05}-2\mu_{23}-3\mu_{41})^2 \\ I_{n_3} = (\mu_{50}-10\mu_{32}+5\mu_{14})^2 + (\mu_{05}-10\mu_{23}+5\mu_{41})^2 \end{cases}$$

μ_{ij} are the centered moments defined by:

$$\mu_{ij} = \sum_{k=1}^{N}(x_k - x_g)^i(y_k - y_g)^j$$

where $(x,\ y)$ are the projection of 3D point onto the image using a perspective projection, N is the number of points, and (x_g, y_g) is the center of the set of points in the image. Complete details on how r_i and r_j have been determined can be found in [15]. The interaction matrix $\mathbf{L_s}$ related to the above three features with respect to rotational degrees of freedom has the following form [17]:

$$\mathbf{L_s} = \begin{bmatrix} r_{1_{wx}} & r_{1_{wy}} & 0 \\ r_{2_{wx}} & r_{2_{wy}} & 0 \\ \alpha_{wx} & \alpha_{wy} & -1 \end{bmatrix}$$

In the other hand, as in [18], [15], the invariant to rotations will be used to control the translational motions. For instance, the following polynomials are invariant to rotational motions [15]:

$$I_1 = m_{200}m_{020} - m_{200}m_{002} + m_{110}^2 + m_{101}^2 - m_{020}m_{002} + m_{011}^2 \tag{26}$$

$$\begin{aligned} I_2 = &- m_{300}m_{120} - m_{300}m_{102} + m_{210}^2 - m_{210}m_{030} - m_{210}m_{012} \\ &+ m_{201}^2 - m_{201}m_{021} - m_{201}m_{003} + m_{120}^2 - m_{120}m_{102} \\ &+ 3m_{111}^2 + m_{102}^2 - m_{030}m_{012} + m_{021}^2 - m_{021}m_{003} + m_{012}^2 \end{aligned} \tag{27}$$

$$\begin{aligned} I_3 = &\ m_{300}^2 + 3m_{300}m_{120} + 3m_{300}m_{102} + 3m_{210}m_{030} \\ &+ 3m_{210}m_{012} + 3m_{201}m_{021} + 3m_{201}m_{003} + 3m_{120}m_{102} \\ &- 3m_{111}^2 + m_{030}^2 + 3m_{030}m_{012} + 3m_{021}m_{003} + m_{003}^2 \end{aligned} \tag{28}$$

where:

$$m_{i,j,k} = \sum_{h=0}^{N} x_{s_h}^i\ y_{s_h}^j\ z_{s_h}^k \tag{29}$$

with $(x_s,\ y_s,\ z_s)$ are the coordinates of the projection of a 3D point onto the unit sphere [9], [15].

B. Simulations where translational motion is considered

In a first simulation only translational motion is considered. It will be seen that the classical version presented in [11] is still valid and improve the system behavior. For this, we compare the convergence success percentage using the MJP method or the classical one (i.e. using the current value of the interaction matrix) in the case where only a translational motion is considered. The convergence success percentage is defined by the percentage of the case when the system converge to the global minimum. In order to do so, thousands of random translational motion with different norms were generated. The percentage of the convergence success was computed with respect to the translation vector norm. Figure 1 shows the results using the MJP (continuous plot) and using the classical method (dashed plot). From this figure, it can be seen that the convergence percentage is noticeably better using the MJP. This was expected, since the MJP is valid if only a translational motion is considered.

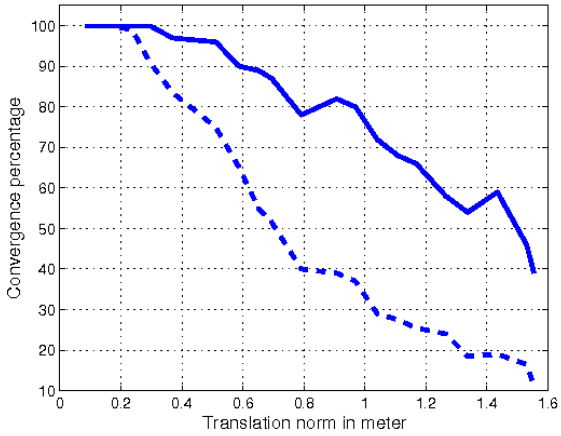


Fig. 1. Convergence percentage if only a translational motion is considered: dashed plot shows the result using the current value of the interaction matrix, continuous plot shows the result using MJP

In the second simulation, the displacement combining the translation and the rotation given respectively by (31) and (30) is considered. The obtained results using the MJP or the classical method are given on Figure 2. From the latter figure, it can be noticed that the obtained results using MJP are not better compared to the classical method results. Indeed, oscillations of two components of the translational velocities using MJP can be observed. The dashed plot corresponds to the translational velocity with respect to the optical axis. Note that this component did not suffer oscillation, since the considered rotational motion did not change the orientation of the optical axis. In fact the oscillation observed for the two others components can be explained by the different orientations of the current and the desired camera frames. In the next subsection, results for rotation motion only will be presented.

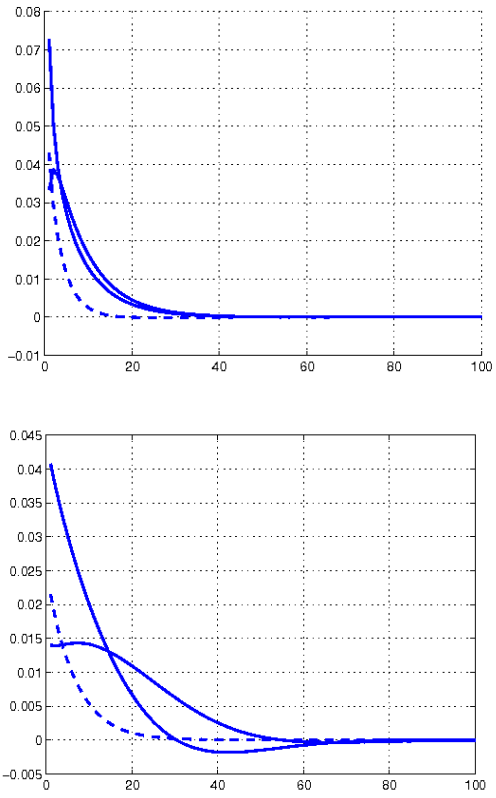


Fig. 2. Translational velocities (in m/s): result using the current value of the interaction matrix (top), result MJP (bottom)

$$\boldsymbol{\theta}\mathbf{u} = \begin{pmatrix} 0 & 0 & 80 \end{pmatrix}^o \tag{30}$$

$$\mathbf{t} = \begin{pmatrix} -0.4 & -0.4 & -0.15 \end{pmatrix} cm \tag{31}$$

C. The ESM behavior for large rotational motion

In this simulation, the rotational motion expressed as the rotation vector given by (32) has been considered.

$$\mathbf{u}\boldsymbol{\theta} = \begin{pmatrix} 21.82 & 0 & 87.00 \end{pmatrix} \tag{32}$$

Further, a random set of 10 coplanar points has been generated for the desired position. The interaction matrices computed for the the current and desired positions are given respectively by (33) and (34). It can be noticed the large difference between the interaction matrix values.

$$\mathbf{L_s} = \begin{pmatrix} 67.80 & 17.01 & -0.00 \\ -12.00 & -10.49 & -0.00 \\ -0.65 & -0.16 & -1.00 \end{pmatrix} \tag{33}$$

$$\mathbf{L_{s^*}} = \begin{pmatrix} 9.0053 & 43.78 & -0.00 \\ -6.42 & 6.38 & 0.00 \\ 0.00 & -0.04 & -1.00 \end{pmatrix} \tag{34}$$

This is the consequence of the large rotation around the optical axis (nearly 90^o). Figure 3 gives the plot of the obtained results using respectively the MJP, the PMJ and the

classical method based on the current value of the interaction matrix. The features errors plots are given on Figures 3.a, 3.c and 3.e. The velocities are plotted on 3.b, 3.d and 3.f. We can note that the results using the classical method are very satisfactory. On the other hand, it can also be seen that the use of MJP and PMJ does not improve the result obtained using the classical method. Contrary, from the corresponding plots, the use of these methods disturbed the system and several oscillations can be observed. For a larger rotational motion around the optical axis, the control law using the MJP and the PMJ becomes unstable and diverges.

The obtained results for a large rotational motion around the optical axis were expected. In fact, in visual servoing, the computed velocity is expressed in the current frame and applied to this frame. Further, the interaction matrix determines the feature variations with respect to the camera velocity. In other words, it determines the direction of the motion to apply. Thus, combining $\mathbf{L_{s^*}}$ and $\mathbf{L_s}$ in the control laws MJP, and PMJ is not safe business, since this does not take into account the difference between the camera frame orientations in its initial and desired positions. To take this into account, the tensor ${}^{\mathbf{d}}\mathbf{vd} = \mathbf{L_{s^*}}(\mathbf{s} - \mathbf{s}^*)$ should be expressed in the current frame. In next subsection, we will see how the interaction matrix entries behave with respect to rotational motion.

D. Interaction matrix entries variations with respect to rotation

As a significant example, the variations of the interaction matrix entries with respect to rotation around the optical axis is presented. Figure 4 gives the variation of the interaction matrix entries with respect to the rotation angle. In this Figure, the curves with dashed lines correspond to the component related to the optical axis. From the latter, it can be noticed that they are constant. This was expected, since a rotational motion around the optical axis does not change the orientation of this axis. Thus, the variation of the selected features with respect to rotational motion around the optical axis is still constant. From, the same figure, it can also be seen that the variation of the other entries are sinusoidal functions. In fact, the interaction matrix after rotational motion is the product of this matrix by the rotation matrix for the considered features. Further, from the same figure, it can be seen that the matrix entries change their signs if the rotation angle is more than $\frac{\pi}{2}$. This mean although if $\mathbf{L_s}$ and $\mathbf{Ls}^*$ are not singular their sum might be singular. Thus, a control law using $\mathbf{L_s} + \mathbf{Ls}^*$ as proposed in [11] would make unstable the system behavior if a large rotational motion is performed.

E. The ESM behavior by taking into account the rotational motion

In the case where only a rotational motion is considered, ${}^{d}\mathbf{v_d}$ can be expressed in the current frame as follow:

$$ {}^{c}\mathbf{v_d} = -{}^{\mathbf{c}}\mathbf{R_d}\,{}^{d}\mathbf{v_d} $$

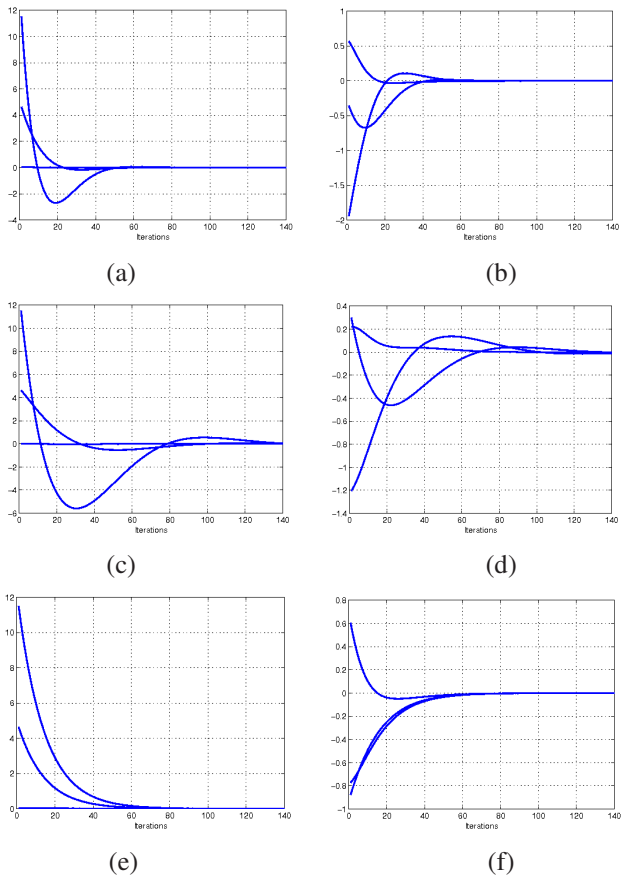


Fig. 3. Validations: a) feature errors using pseudo-inverses of the mean of the Jacobians, b) velocities (degree/s) using pseudo-inverses of the mean of the Jacobians, c) feature errors using the mean of pseudo-inverses of the Jacobians, d) velocities (degree/s) errors using the mean of pseudo-inverses of the Jacobians, e) feature errors using the pseudo-inverse of the current Jacobian, f) velocities (degree/s) using the pseudo-inverse of the current Jacobian

${}^{\mathbf{c}}\mathbf{R_d}$ is the rotation matrix between the two camera positions. For instance, an estimation of ${}^{\mathbf{c}}\mathbf{R_d}$ can be obtained using a model-based pose estimation method [7] (if the object model is available), or a model-free pose estimation [12](in the case where no object model is available). This leads to two control laws:

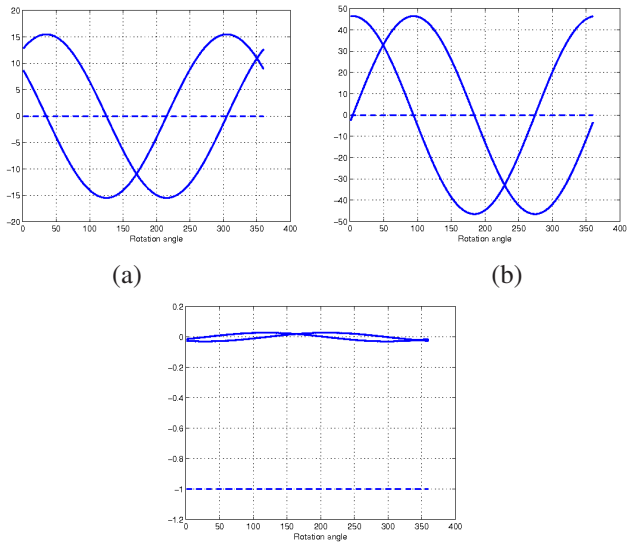
$$ \mathbf{v_{MJP}} = -\frac{1}{2}\lambda(\mathbf{L_s}^{+} + {}^{\mathbf{c}}\mathbf{R_d}\mathbf{L_{s^*}})^{+})(\mathbf{s} - \mathbf{s}^*) \qquad (35) $$

$$ \mathbf{v_{PJM}} = -2\lambda(\mathbf{L_s} + \mathbf{L_{s^*}}\,{}^{\mathbf{d}}\mathbf{R_c})^{+}(\mathbf{s} - \mathbf{s}^*) \qquad (36) $$

In a second simulation, the two above control laws was used. The rotational motion given by (32) is considered. Figure 5 gives the plots of the obtained results. From this figure, it is clearly noticeable that the oscillations observed in the case where the classical ESM was used disappeared. Indeed, a satisfactory exponential decrease for both features errors and velocities were obtained.

IV. Conclusions and Discussions

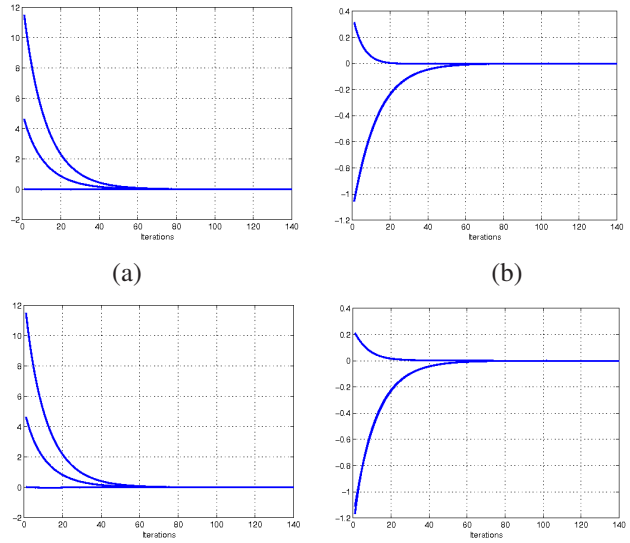
This paper dealt with the validity of the "Efficient second order minimization" for visual servoing application proposed in [11]. It has been shown that this method is not valid if a



(a) (b)

(c)

Fig. 4. Interaction matrix entries variations: a) result for r_1, b) result for r_2 a) result for α



(a) (b)

(c) (d)

Fig. 5. Validations: a) feature errors using (35), b) velocities (degree/s) using (35), c) feature errors using (36), d)velocities (degree/s) errors using (36)

large displacement has to be performed. More precisely, if a large rotational motion is considered, the use of the ESM does not necessary ensure a better behavior of the system than the classical method based on the current value of the interaction matrix. Contrary, the ESM can make unstable the system control and produces oscillations of the features errors and velocities. Worse, in some cases, it can cause system control divergence.

An adequate application of the ESM has to take into account the coordinates transformation $\mathbf{T}$ from the current and the desired frame of the camera. In general, this means that the object model is completely known. However, in the case where the used features ensures a complete decoupling between the rotational and the translational motions, the knowledge of the object model can be avoided. Indeed, the features used to perform the simulations in this paper ensure a decoupled control of the translation and the rotations. This means that only the rotation between the current and the desired camera positions has to be known. This can be computed using a model-free method for partial pose computation.

REFERENCES

[1] H. H. Abdelkader, Y. Mezouar, and P. Martinet. Path planning for image based control with omnidirectional cameras. In CDC'06, editor, *45th IEEE Conference on Decision and Control*, San Diego, California, USA, 13-15 December 2006.

[2] S. Benhimane and E. Malis. Real-time image-based tracking of planes using efficient second-order minimization. In *IEEE/RSJ International Conference on Intelligent Robots Systems*, Sendai, Japan, October 2004.

[3] S. Benhimane and E. Malis. Integration of euclidean constraints in template-based visual tracking of piecewise-planar scenes. In *IEEE/RSJ International Conference on Intelligent Robots Systems*, Beijing, China, October 2006.

[4] F. Chaumette. Potential problems of stability and convergence in imagebased and position-based visual servoing. In Springer-Verlag, editor, *The Confluence of Vision and Control*, volume 237 of *LNCIS*, pages 66–78, 1998.

[5] F. Chaumette. Image moments: A general and useful set of features for visual servoing. *IEEE Transaction on Robotics and Automation*, 20(4):713723, August 2004.

[6] N. Cowan, J. Dingarten, and D. Koditschek. Visual servoing via navigation functions. *IEEE Transactions on Robotics and Automation*, 2002.

[7] D. Dementhon and L. Davis. Model-based object pose in 25 lines of code. *International Journal of Computer Vision*, 15(1-2):123–141, June 1995.

[8] B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. *IEEE Transaction on Robotics and Automation*, 8:313–326, June 1992.

[9] T. Hamel and R. Mahony. Visual servoing of an under-actuated dynamic rigid body system: an image-based approach. *IEEE Transaction on Robotics and Automation*, 18(2):187–198, April 2002.

[10] J. T. Lapreste and Y. Mezouar. A hessian approach to visual servoing. In *International Conference on Intelligent Robots and Systems*, pages 998–1003, Sendai, Japan, September 28 October 2 2004.

[11] E. Malis. Improving vision-based control using efficient second-order minimization techniques. In *IEEE International Conference on Robotics and Automation*, New Orleans, Louisiana, April 2004.

[12] E. Malis, F. Chaumette, and S. Boudet. 2 1/2 d visual servoing. *IEEE Transaction on Robotics and Automation*, 15(2):238–250, 1999.

[13] Y. Mezouar and F. Chaumette. Path planning for robust image-based control. *IEEE Transaction on Robotics and Automation*, 18(4):534–549, August 2002.

[14] F. Schramm, F. Geffard, G. Morel, and A. Micaelli. Calibration free image point path planning simultaneously ensuring visibility and controlling camera path. In *IEEE International Conference on Robotics and Automation*, pages 2074–2079, Roma, April 2007.

[15] O. Tahri. *Utilisation des moments en asservissement visuel et en calcul de pose*. PhD thesis, University of Rennes, 2004.

[16] O. Tahri and F. Chaumette. Application of moment invariants to visual servoing. In *IEEE International Conference on Robotics and Automation*, pages 4276–4281, Taipeh, Taiwan, September 2003.

[17] O. Tahri and F. Chaumette. Point-based and region-based image moments for visual servoing of planar objects. *IEEE Transaction on Robotics*, 21(6):1116–1127, December 2005.

[18] O. Tahri, F. Chaumette, and Y. Mezouar. New decoupled visual servoing scheme based on invariants from projection onto a sphere. In *In IEEE International Conference on Robotics and Automation, ICRA'08*.

[19] W. Wilson, C. Hulls, and G. Bell. Relative end-effector control using cartesian position-based visual servoing. *IEEE Transaction on Robotics and Automation*, 12(5):684–696, October 1996.