HAL
archives-ouvertes.fr

Visual servoing set free from image processing

Christophe Collewet, Eric Marchand, François Chaumette

▶ **To cite this version:**

Christophe Collewet, Eric Marchand, François Chaumette. Visual servoing set free from image processing. IEEE Int. Conf. on Robotics and Automation, ICRA'08, May 2008, Pasadena, United States. 2008. <inria-00261398>

HAL Id: inria-00261398
https://hal.inria.fr/inria-00261398

Submitted on 6 Mar 2008

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Visual servoing set free from image processing

Christophe Collewet, Eric Marchand, François Chaumette

***Abstract*— This paper proposes a new way to achieve robotic tasks by visual servoing. Instead of using geometric features (points, straight lines, pose, homography, etc.) as it is usually done, we use directly the luminance of all pixels in the image. Since most of the classical control laws fail in this case, we turn the visual servoing problem into an optimization problem leading to a new control law. Experimental results validate the proposed approach and show its robustness regarding to approximated depths, non Lambertian objects and partial occlusions.**

I. INTRODUCTION

Visual servoing consists in using the information provided by a vision sensor to control the movements of a dynamic system [6]. Robust extraction and real-time spatio-temporal tracking of visual cues is usually one of the keys to success of a visual servoing task. This tracking process has been, to date, considered as a necessary step and is also one of the bottlenecks of the expansion of visual servoing. In this paper we show that such a tracking process can be totally removed and that no other information than the image intensity (the pure image signal) can be considered to control the robot motion.

Classically, to achieve a visual servoing task, a set of visual features has to be selected from the image allowing to control the desired degrees of freedom. A control law has also to be designed so that these visual features $\mathbf{s}$ reach a desired value $\mathbf{s}^*$, leading to a correct realization of the task. The control principle is thus to regulate to zero the error vector $\mathbf{s}-\mathbf{s}^*$. To build this control law, the knowledge of the interaction matrix $\mathbf{L_s}$ is usually required. For eye-in-hand systems, it links the time variation of $\mathbf{s}$ to the camera instantaneous velocity $\mathbf{v}$

$$\dot{\mathbf{s}} = \mathbf{L_s}\,\mathbf{v} \tag{1}$$

with $\mathbf{v} = (\boldsymbol{v}, \boldsymbol{\omega})$ where $\boldsymbol{v}$ is the linear velocity and $\boldsymbol{\omega}$ is the angular velocity. This interaction matrix plays an essential role. Indeed, if we consider the camera velocity as input of the robot controller, the control law is designed to try to obtain an exponential decoupled decrease of the error $\mathbf{s}-\mathbf{s}^*$

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

where λ is a proportional gain that has to be tuned to minimize the time-to-convergence, and $\widehat{\mathbf{L}}_{\mathbf{s}}^{+}$ is the pseudo-inverse of a model or an approximation of $\mathbf{L_s}$ [6].

Visual servoing explicitly relies on the choice of the visual features $\mathbf{s}$ (and then on the related interaction matrix); that is the key point of the approach. With a vision sensor providing 2D measurements $\mathbf{x}(\mathbf{r}_k)$ (where $\mathbf{r}_k$ is the camera pose at time k), potential visual features $\mathbf{s}$ are numerous, since 2D data (coordinates of feature points in the image, moments, ...) as well as 3D data provided by a localization algorithm exploiting the extracted 2D features can be considered. If the choice of $\mathbf{s}$ is important, it is always designed from visual measurements $\mathbf{x}(\mathbf{r}_k)$. A robust extraction, matching (between $\mathbf{x}(\mathbf{r}_k)$ and $\mathbf{x}^* = \mathbf{x}(\mathbf{r}^*)$) and real-time spatio-temporal tracking (between $\mathbf{x}(\mathbf{r}_{k-1})$ and $\mathbf{x}(\mathbf{r}_k)$) have proved to be difficult, as testified by the abundant literature on the subject.

In this paper we propose to radically modify the procedure by removing the extraction of geometric measurements and consequently the matching and tracking process. To achieve this goal we use as visual features the simplest feature that can be considered: the image intensity. The visual feature vector $\mathbf{s}$ is nothing but the image while $\mathbf{s}^*$ is the desired image. The error $\mathbf{s}-\mathbf{s}^*$ is then only the difference between the current and desired image (that is $\mathbf{I}-\mathbf{I}^*$ where $\mathbf{I}$ is a vector that contains image intensity of all pixels). If we assume that the observed scene is Lambertian, we can exhibit the analytical form of the interaction matrix related to the luminance. This interaction matrix can be derived from the *optical flow constraint equation* (OFCE) [5] as has been done in [9]. Using the image intensity as visual features, the classical control law, given by equation (2), at best converges with a slow and inappropriate camera motion or simply diverges, we thus turn the visual servoing problem into a minimization one as in [8] or [10]. We show that using this approach it is then possible to handle positioning tasks.

Considering the whole image as a feature has previously been considered in [2], [3], [11]. As in our case, the methods presented in [2], [3], [11] did not require a matching process. Nevertheless they differ from our approach in two important points. First, they do not use directly the image intensity but an eigenspace decomposition is performed to reduce the dimensionality of image data. The control is then performed directly in the eigenspace and not directly with the image intensity. This requires the off-line computation of this eigenspace (using a principal component analysis) and then, for each new frame, the projection of the image on this subspace. Second, the interaction matrix related to the eigenspace is not computed analytically but is learned during an off-line step. This learning process has two drawbacks: it has to be done for each new object and requires the acquisition of many images of the scene at various camera positions. Considering an analytical interaction matrix avoids these issues.

Finally, in [1], the authors present an homography-based

C. Collewet, E. Marchand and F. Chaumette are with INRIA Rennes - Bretagne Atlantique, IRISA, Lagadic team, Rennes, France.
firstname.name@irisa.fr.

approach to visual servoing. In this method the image intensity of a planar patch is first used to estimate the homography (using the ESM algorithm described in [1] for example) between current and desired image which is then used to build the control law. Despite the fact that, as in our case, image intensity is used as the basis of the approach, an important image processing step is necessary to estimate the homography. Furthermore, the visual features used in the control law rely on the homography matrix. Finally, a matching process between the current and desired image patches is required.

In the remainder of this paper we first reformulate the visual servoing problem as an optimization problem in Section II. The interaction matrix related to the luminance is then recalled in Section III. We then study in Section IV the chosen cost function and propose an optimization method suitable to our problem in Section V. Section VI shows experimental results for several positioning tasks.

II. Visual servoing as an optimization problem

Different control laws can be derived regarding the minimization technique one uses. Their goal is to minimize the following cost function

$$\mathcal{C}(\mathbf{r}) = (\mathbf{s}(\mathbf{r}) - \mathbf{s}(\mathbf{r}^*))^\top (\mathbf{s}(\mathbf{r}) - \mathbf{s}(\mathbf{r}^*)) \tag{3}$$

where $\mathbf{r}$ describes the current pose of the camera with respect to the object (it is an element of $\mathbb{R}^3 \times SO(3)$) and where $\mathbf{r}^*$ is the desired pose. Several methods are detailed in [8], we only give here the most interesting results while focusing on the differential approaches. In that case, a step of the minimization scheme can be written as follows

$$\mathbf{r}_{k+1} = \mathbf{r}_k \oplus t_k \mathbf{d}(\mathbf{r_k}) \tag{4}$$

where "$\oplus$" denotes the operator that combines two consecutive frame transformations, $\mathbf{r}_k$ is the current pose, t_k is a positive scalar (the descent step) and $\mathbf{d}(\mathbf{r_k})$ a direction of descent ensuring that (3) decreases if

$$\mathbf{d}(\mathbf{r_k})^\top \nabla\mathcal{C}(\mathbf{r}_k) < 0. \tag{5}$$

In that case, the following velocity control law can be derived considering that t_k is small enough

$$\mathbf{v} = \lambda_k \mathbf{d}(\mathbf{r_k}) \tag{6}$$

where λ_k is a scalar that depends on t_k and on the sampling rate. It is often chosen as a constant value. In the remainder of the paper we will omit the subscript k for the sake of clarity.

A. Steepest descent (gradient method)

The direction of descent is simply

$$\mathbf{d}(\mathbf{r}) = -\nabla\mathcal{C}(\mathbf{r}) \tag{7}$$

where

$$\nabla\mathcal{C}(\mathbf{r}) = \left(\frac{\partial \mathbf{s}}{\partial \mathbf{r}}\right)^\top (\mathbf{s}(\mathbf{r}) - \mathbf{s}(\mathbf{r}^*)). \tag{8}$$

Since we have $\dot{\mathbf{s}} = \frac{\partial \mathbf{s}}{\partial \mathbf{r}}\dot{\mathbf{r}} = \mathbf{L_s}\mathbf{v}$, we obtain the following control law

$$\mathbf{v} = -\lambda \mathbf{L_s}^\top (\mathbf{s}(\mathbf{r}) - \mathbf{s}(\mathbf{r}^*)). \tag{9}$$

For instance, this approach has been used in [4].

B. Gauss-Newton

When $\mathbf{r_k}$ lies in a neighborhood of $\mathbf{r}^*$, $\mathbf{s}(\mathbf{r})$ can be linearized around $\mathbf{s}(\mathbf{r_k})$ and plugged into (3). Then, after having zeroed its gradient, we obtain

$$\mathbf{d}(\mathbf{r}) = -\left(\left(\frac{\partial \mathbf{s}}{\partial \mathbf{r}}\right)^\top \left(\frac{\partial \mathbf{s}}{\partial \mathbf{r}}\right)\right)^{-1} \nabla\mathcal{C}(\mathbf{r}) \tag{10}$$

that becomes using (8)

$$\mathbf{v} = -\lambda \mathbf{L_s}^+ (\mathbf{s}(\mathbf{r}) - \mathbf{s}(\mathbf{r}^*)) \tag{11}$$

which is nothing but (2). It is the control law usually used.

C. Newton

If we locally approximate $\mathcal{C}(\mathbf{r})$ by its second order Taylor series expansion in $\mathbf{r_k}$ and cancel its gradient, we have

$$\mathbf{d}(\mathbf{r}) = -\left(\nabla^2\mathcal{C}(\mathbf{r})\right)^{-1} \nabla\mathcal{C}(\mathbf{r}) \tag{12}$$

with

$$\nabla^2\mathcal{C}(\mathbf{r}) = \left(\frac{\partial \mathbf{s}}{\partial \mathbf{r}}\right)^\top \left(\frac{\partial \mathbf{s}}{\partial \mathbf{r}}\right) + \sum_{i=1}^{i=\dim \mathbf{s}} \nabla^2 s_i \left(s_i(\mathbf{r}) - s_i(\mathbf{r}^*)\right). \tag{13}$$

This approach has shown its efficiency in [7] for example. Note that the vector $\mathbf{d}(\mathbf{r})$ is really a direction of descent if $\nabla^2\mathcal{C}(\mathbf{r}) > 0$ holds (see (5)). Note also that the Newton's and Gauss-Newton's approaches are equivalent in $\mathbf{r}^*$.

D. Levenberg-Marquardt

This method considers the following direction

$$\mathbf{d}(\mathbf{r}) = -\left(\mathbf{G} + \mu\ \mathrm{diag}(\mathbf{G})\right)^{-1} \nabla\mathcal{C}(\mathbf{r}) \tag{14}$$

where $\mathbf{G}$ is usually chosen as $\nabla^2\mathcal{C}(\mathbf{r})$ or more simply as $\left(\frac{\partial \mathbf{s}}{\partial \mathbf{r}}\right)^\top \left(\frac{\partial \mathbf{s}}{\partial \mathbf{r}}\right)$ leading in that last case to

$$\mathbf{v} = -\lambda \left(\mathbf{H} + \mu\ \mathrm{diag}(\mathbf{H})\right)^{-1} \mathbf{L_s}^\top (\mathbf{s}(\mathbf{r}) - \mathbf{s}(\mathbf{r}^*)) \tag{15}$$

with $\mathbf{H} = \mathbf{L_s}^\top \mathbf{L_s}$. The parameter μ makes possible to switch from a steepest descent like approach to a Gauss-Newton one thanks to the observation of (3) during the minimization process. Indeed, when μ is very high (15) behaves like (9)[1]. In contrast, when μ is very low (15) behaves like (11).

[1]More precisely, each component of the gradient is scaled according to the diagonal of the Hessian, which leads to larger displacements along the direction where the gradient is low.

III. LUMINANCE AS A VISUAL FEATURE

The visual features considered in this paper are the luminance I of each point of the image. In fact we have

$$\mathbf{s}(\mathbf{r}) = \mathbf{I}(\mathbf{r}) = (\mathbf{I}_{1\bullet}, \mathbf{I}_{2\bullet}, \cdots, \mathbf{I}_{N\bullet}) \tag{16}$$

where $\mathbf{I}_{k\bullet}$ is nothing but the k-th line of the image. $\mathbf{I}(\mathbf{r})$ is then a vector of size $N \times M$ where $N \times M$ is the size of the image.

As it has been shown in Section II, all the control laws require an estimation of the interaction matrix. In our case, as already stated, we are looking for the interaction matrix related to the luminance of a pixel in the image. Its computation is detailed in [9][2] and is recalled now.

The basic hypothesis assumes the temporal constancy of the brightness for a physical point between two successive images. This hypothesis leads to the so-called *optical flow constraint equation* (OFCE) that links the temporal variation of the luminance I to the image motion at point $\mathbf{x}$ [5].

More precisely, assuming that the point has a displacement $\mathbf{dx}$ in the time interval dt, the previous hypothesis leads to

$$I(\mathbf{x} + \mathbf{dx}, t + dt) = I(\mathbf{x}, t). \tag{17}$$

Written with a differential form, a first order Taylor series expansion of this equation around $\mathbf{x}$ gives

$$\nabla I^\top \dot{\mathbf{x}} + \dot{I} = 0. \tag{18}$$

with $\dot{I} = \partial I / \partial t$. It becomes then straightforward to compute the interaction matrix $\mathbf{L}_I$ related to I by plugging the interaction matrix $\mathbf{L}_\mathbf{x}$ related to $\mathbf{x}$ into (18). We obtain

$$\dot{I} = -\nabla I^\top \mathbf{L}_\mathbf{x} \mathbf{v}. \tag{19}$$

Finally, if we introduce the interaction matrices $\mathbf{L}_x$ and $\mathbf{L}_y$ related to the coordinates x and y of $\mathbf{x}$, we obtain

$$\mathbf{L}_I = -(\nabla I_x \ \mathbf{L}_x + \ \nabla I_y \mathbf{L}_y) \tag{20}$$

where ∇I_x and ∇I_y are the components along x and y of ∇I. Note that it is actually the only image processing step necessary to implement the presented method.

Of course, because of the hypothesis required to derive (17), (20) is only valid for Lambertian scenes, that is for surfaces reflecting the light with the same intensity in each direction. Besides, (20) is also only valid for a motionless lighting source with respect to the scene. These hypotheses can be seen as restrictive, however we will see in Section VI that (20) can be efficiently used even when Lambertian constraints are not always valid.

IV. ANALYSIS OF THE COST FUNCTION

Since the convergence of the control laws described in Section II highly depends on the cost function (3), we focus here on its shape.

To do that, we consider the vector $\mathbf{I}(\mathbf{r})$ given by (16). We write $\mathbf{r} = (\mathbf{t}, \boldsymbol{\theta})$ where $\mathbf{t} = (t_x, t_y, t_z)$ describes the translation part of the homogeneous matrix related to the transformation from the current to the desired frame, and where $\boldsymbol{\theta} = (\theta_x, \theta_y, \theta_z)$ describes its rotation part (pitch, yaw, roll).

As an example, Fig. 1b and Fig. 1c describe the shape of the cost function (3) in the subspace (t_x, θ_y) when the scene being observed is planar (see Fig. 1a) and when the desired pose is such that the image plane and the object plane are parallel at the depth Z^* = 80 cm. Let us point out that this is the most complex case (with its dual case (t_y, θ_x)). Indeed, it is well known that it is very difficult to distinguish in an image an x axis translational motion (respectively y) from a y axis rotational motion (respectively x). It explains why the cost function is low in a preferential direction, as clearly shown on Fig. 1b and Fig. 1c.

Simulations on various images have shown that the shape of the cost function (3) does not depend too much on the scene as soon as the image does not contain periodic patterns or strong changes of the spatial gradient. For the motions considered here, it always shows a narrow valley at the middle of a gentle slope plateau with non constant slope. Note that the direction of that valley depends on Z^*. It is in the direction of the θ_y axis when $Z^* \approx 0$, the direction of t_x when $Z^* \to +\infty$. Note also in Fig. 1b that (3) is only quasi convex, moreover on a very small domain.

Let us study more precisely (3) in a neighborhood of $\mathbf{r}^*$. To do that, we perform a first order Taylor series expansion of the visual features $\mathbf{I}(\mathbf{r})$ around $\mathbf{r}^*$

$$\mathbf{I}(\mathbf{r}) = \mathbf{I}(\mathbf{r}^*) + \mathbf{L}_{\mathbf{I}^*} \boldsymbol{\Delta}\mathbf{r} \tag{21}$$

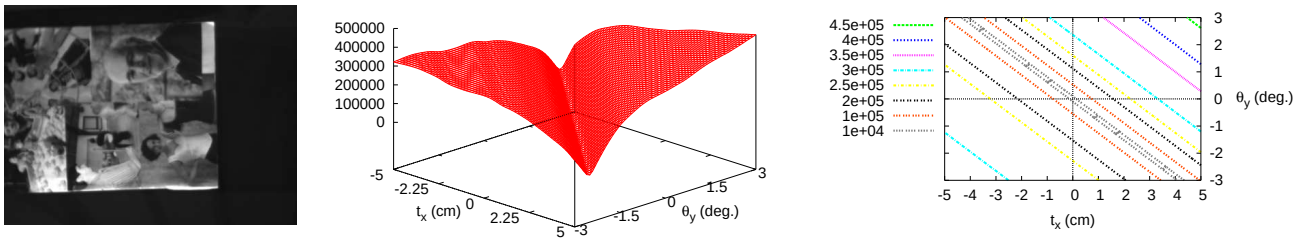
where $\boldsymbol{\Delta}\mathbf{r}$ denotes the relative pose between $\mathbf{r}$ and $\mathbf{r}^*$. Therefore, by plugging (21) and (16) into (3), the cost function can be approximated in a neighborhood of $\mathbf{r}^*$

$$\widehat{\mathcal{C}(\mathbf{r})} = \boldsymbol{\Delta}\mathbf{r}^\top \mathbf{H}^* \boldsymbol{\Delta}\mathbf{r} \tag{22}$$

with $\mathbf{H}^* = \mathbf{L}_{\mathbf{I}^*}^\top \mathbf{L}_{\mathbf{I}^*}$. In practice, because of the special form of the interaction matrix given in (20) (its translation part contains terms related to the depths), the eigenvalues of the matrix $\mathbf{H}^*$ are very different[3] (unfortunately, only numerical results can be obtained because of the complexity of this matrix). Consequently, in the subspace (t_x, θ_y) (respectively (t_y, θ_x)), the cost function is an elliptic paraboloid with a very high major axis with respect to its minor axis leading consequently to near parallel isocontours as shown on Fig. 1c. Moreover, the eigenvectors of $\mathbf{H}^*$ point out some directions where the cost function decreases slowly when its associated eigenvalue is low or decreases quickly when its associated eigenvalue is high. In the case of Fig. 1c, the eigenvector associated to the smaller eigenvalue corresponds to the valley where the cost varies slowly. In contrast, it varies strongly along an orthogonal direction, that is in a direction near $\nabla\mathcal{C}(\mathbf{r})$. We will use this knowledge about the cost function in the next section.

[2] In [9], this interaction matrix has been used to control the light source or the camera position in order to ensure optimal lighting condition of a scene.

[3] This result also holds for most of the geometrical visual features where a term related to the depth occurs in the translational part of the interaction matrix.



(a) (b) (c)

Fig. 1. Cost function: (a) Object being observed, (b) Shape of the cost function in the subspace (t_x, θ_y), (c) Isocontours in the subspace (t_x, θ_y).

V. Positioning tasks

As shown in Section II, several control laws can be used to minimize (3). We first used the classical control laws based on the Gauss-Newton approach and the ESM approach [8]. Unfortunately, they all failed, either because they diverged or because they led to unsuitable 3D motion. Therefore, a new control law has been derived.

Indeed, since the general form of the cost function is known (see Fig. 1b), we propose the following algorithm to converge towards the global minimum. The camera is first moved to reach the valleys and next along the axes of the valleys towards the desired pose. The first step can be easily done by using a gradient approach. However, as seen on Fig. 1c, the direction of $\nabla \mathcal{C}(\mathbf{r})$ is constant but its amplitude on the plateau is not constant (see Fig. 1b) since the slope varies. We have thus to tune the parameter λ involved in (9) to ensure smooth 3D velocities. A simpler approach to achieve this goal consists in using the following control law

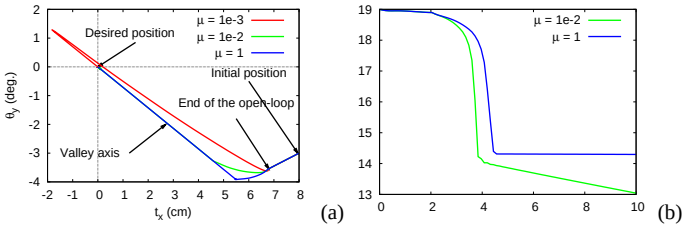
$$\mathbf{v} = -v_c \frac{\nabla \mathcal{C}(\mathbf{r_{init}})}{\| \nabla \mathcal{C}(\mathbf{r_{init}}) \|}. \qquad (23)$$

That is, a constant velocity with norm v_c is applied in the steepest descent computed at the initial camera pose. Consequently, this first step behaves as an open-loop system. To turn into a closed-loop system, we first detect roughly the bottom of the valley from a 3rd order polynomial filtering of $\mathcal{C}(\mathbf{r})$ and then apply the control law (15). Rather to control the parameter μ as in the Levenberg-Marquardt algorithm, we use a constant value as detailed below. We denote MLM this method in the remainder of the paper. Moreover, instead of using for the matrix $\mathbf{H}$ the Hessian of the cost function, we use its approximation $\mathbf{L_I}^\top \mathbf{L_I}$. The resulting control law is then given by

$$\mathbf{v} = -\lambda \left(\mathbf{H} + \mu \text{diag}(\mathbf{H})\right)^{-1} \mathbf{L_I}^\top \left(\mathbf{I}(\mathbf{r}) - \mathbf{I}(\mathbf{r}^*)\right) \qquad (24)$$

with $\mathbf{H} = \mathbf{L_I}^\top \mathbf{L_I}$.

We now detail how μ is tuned. Fig. 2a shows the paths obtained with the MLM algorithm in the case where $\mathbf{r_{init}} = (8$ cm, 4 cm, -10 cm, 3°, -3°, -5°$)$ for various choices of μ. If a high value is used, after the open-loop motion, the bottom of the valley is easily reached (see Fig. 2a when $\mu = 1$) since (24) behaves in this case like a steepest descent approach. But in this case, since the valley is narrow, the convergence rate towards the global minimum (following the direction of the axis of the valley) is low (see Fig. 2b). In



(a) (b)

Fig. 2. Influence of μ. (a) Path in the subspace (t_x, θ_y) for $\mathbf{r} = (8$ cm, 4 cm, -10 cm, 3°, -3°, -5°$)$, (b) Logarithm of the cost function versus time in second.

contrast, if μ is low, (24) behaves like a Gauss-Newton (GN) approach and the convergence is no more ensured (see the large motion on Fig. 2a when $\mu = 10^{-3}$). As can be seen, an intermediate value ($\mu = 10^{-2}$) has to be chosen to ensure an optimal path (Fig. 2a) and a high convergence rate (Fig. 2b). Therefore, this value has been chosen in the experiments described in the next section.

VI. Experimental results

For the first experiment a pure Lambertian and planar object has been used (a tablecloth) to be sure that the interaction matrix given in (20) is valid. The initial pose was $\mathbf{r_{init}} = (20$ cm, 0 cm, 0 cm, 0°, 11°, 0°$)$. In that case, we are not very far from the valley. The desired pose was so that the object and CCD planes are parallel. The interaction matrix has been computed at each iteration but assuming that all the depths are constant and equal to $Z^* = 80$ cm, which is a coarse approximation. Fig. 3a depicts the behavior of the cost function using the GN method while Fig. 3b depicts the behavior of the cost function using the MLM method. The initial and final images are reported respectively on Fig. 3c and Fig. 3d; Fig. 3e and Fig. 3f depict respectively the error cue $\mathbf{I} - \mathbf{I}^*$ at the initial and final positions. First, as can be seen on Fig. 3a and b, both the control laws converge since the cost functions vanish. As can be seen, the time-to-convergence with the GN method is very high wrt the one of the MLM method. Note that it was also very difficult to tune the gain λ for the GN method, a compromise between oscillations at the end of the motion and relative high velocities at the beginning had to be managed. Therefore, it has been set to 5 at the beginning of the motion and to 1 near the convergence. For the MLM method a constant gain $\lambda = 4$ has been used. On the other hand, we can clearly see

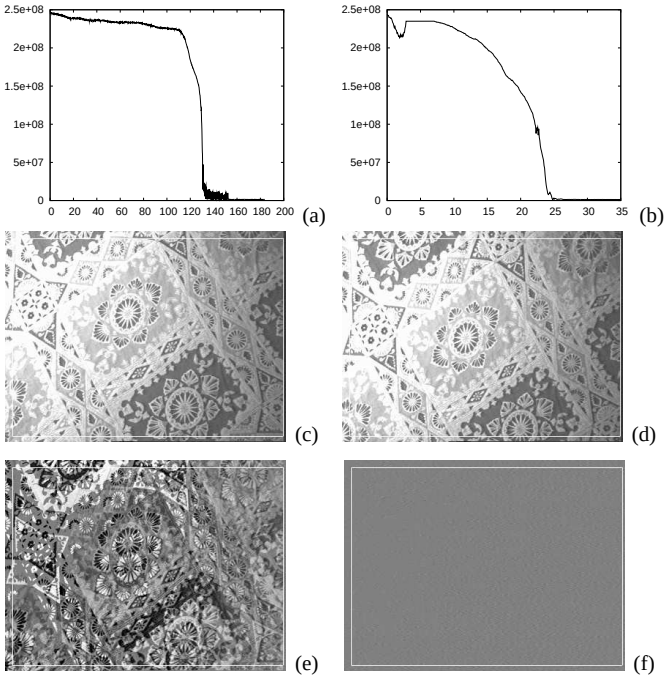


Fig. 3. First experiment. Lambertian object (x axis in second). (a) Cost function using the GN method, (b) Cost function using the MLM method, (c) Initial image, (d) Final image, (e) $\mathbf{I} - \mathbf{I}^*$ at the initial position, (f) $\mathbf{I} - \mathbf{I}^*$ at the final position.

Fig. 4. The non Lambertian object.

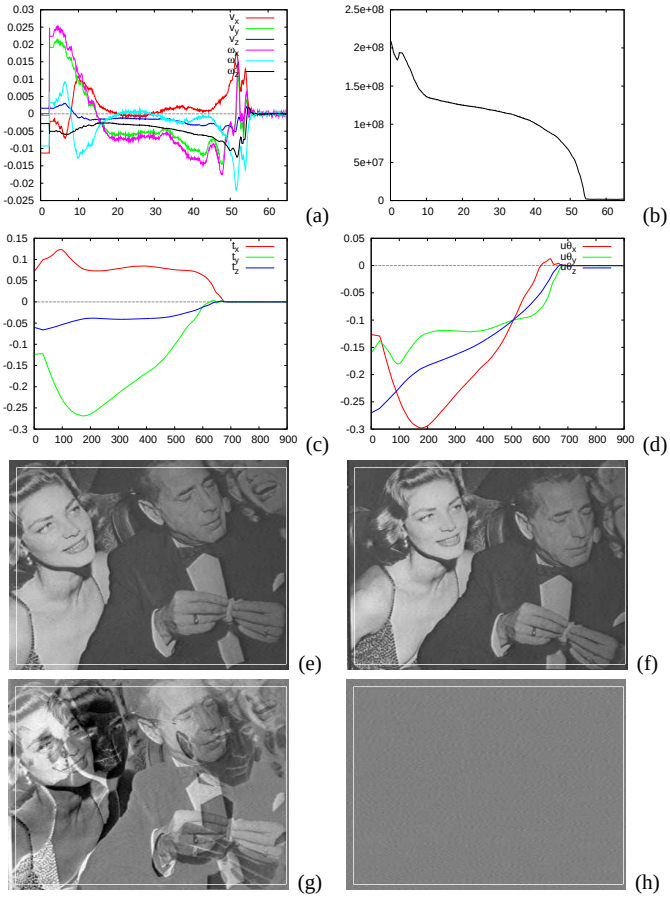


Fig. 5. Second experiment. MLM method (x axis in second for (a) and (b), frame number for (c) and (d)). (a) Camera velocities (m/s or rad/s), (b) Cost function, (c) Translational part of $\boldsymbol{\Delta}\mathbf{r}$ (m), (d) Rotational part of $\boldsymbol{\Delta}\mathbf{r}$ (rad.), (e) Initial image, (f) Final (and desired) image, (g) $\mathbf{I} - \mathbf{I}^*$ at the initial position, (h) $\mathbf{I} - \mathbf{I}^*$ at the end of the motion.

the first step of the MLM algorithm (until $t \approx 2.5$ s) where an open-loop is used, the cost decreases and increases before the bottom of the valley can be detected by the filter (Fig. 3b).

The second experiment is much more complex than the previous one. First, the desired pose has been changed. The image plane and the object plane are no more parallel: a $5°x$ axis rotation has been performed. Thus the desired depths all have different values. Nevertheless, we keep $Z = 80$ cm for all points in the interaction matrix to show the robustness wrt the scene structure variation. Moreover, a non Lambertian object has been used, it is a photograph covered by glass and, as shown in Fig. 4, specularities can clearly be seen. The initial pose is also more complicated since it involves larger motion to reach the desired position. Indeed, we have $\mathbf{r_{init}} = (20$ cm, 10 cm, 5 cm, 10°, 11°, 15°$)$. The behavior of the MLM control law is depicted in Fig. 5. More precisely, Fig. 5a depicts the camera velocity; Fig. 5b the behavior of the cost function; Fig. 5c the translational part of the pose $\boldsymbol{\Delta}\mathbf{r}$ between $\mathbf{r}$ and $\mathbf{r}^*$ and Fig. 5d its rotational part (the rotations are represented by a unit rotation axis vector and a rotation angle around this axis). As can be seen, despite the non Lambertian object and all the approximations we used, the control law converges without any problem. The camera velocities are however noisy. That is because the cost function is not smooth. Nevertheless, they have a small effect on the camera trajectory as seen in Fig. 5c and 5d. Finally, the final positioning error is very low since we have $\boldsymbol{\Delta}\mathbf{r} =$ (-0.1 mm, -0.1 mm, -0.1 mm, -0.01°, -0.01°, -0.01°). It is because $\mathbf{I} - \mathbf{I}^*$ is very sensitive to the pose $\mathbf{r}$.

The next experiment deals with partial occlusions. The desired object pose is unchanged but the initial pose is $\mathbf{r_{init}} = (20$ cm, 10 cm, 5 cm, 10°, 11°, 5°$)$. After having moved the camera to its initial position, an object has been added to the scene, so that the initial image is now the one shown in Fig. 6a and the desired image is still the one shown in Fig. 5f. Moreover, the object introduced in the scene is also moved by hand, as seen in Fig. 6b and Fig. 6c, which highly increases the occluded surface. Despite that, the control law still converges (see Fig. 6f). Of course, since the desired image is not the true one, the cost function does not vanish at the end of the motion (see Fig. 6f and Fig. 6h). Nevertheless, the positioning error is not affected by the occlusions since the final positioning error is $\boldsymbol{\Delta}\mathbf{r}$ = (-0.1 mm, -0.21 mm, -0.1 mm, -0.02°, 0.02°, -0.01°) and it is very similar with

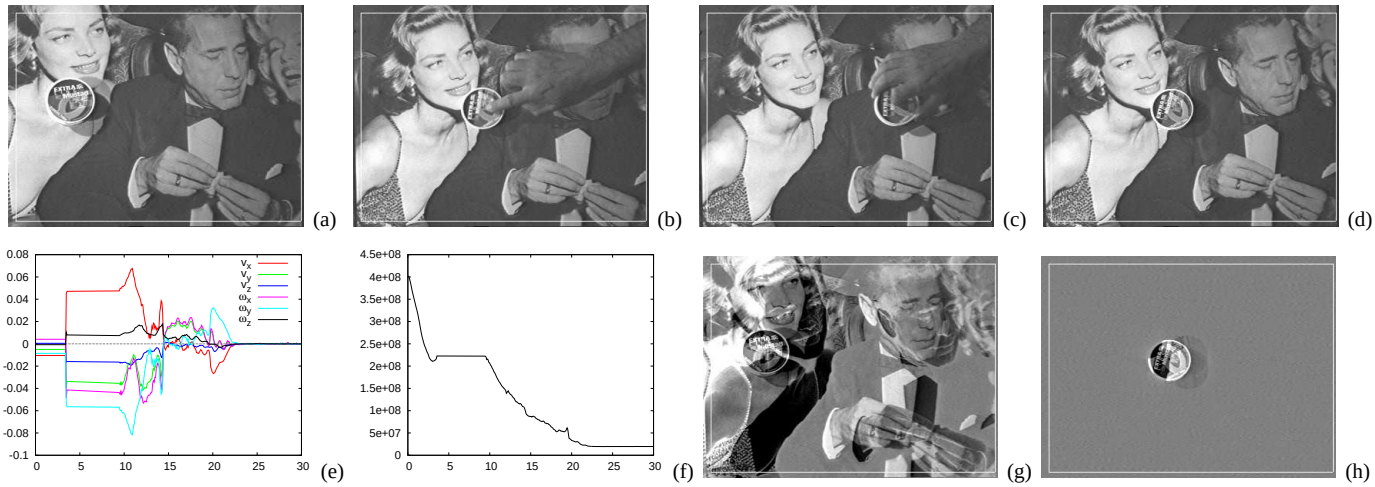


Fig. 6. Third experiment. Occlusions. MLM method (x axis in second). (a) Initial image, (b) Image at t $\approx$ 11 s, (c) Image at t $\approx$ 13 s (d) Final image, (e) Camera velocities (m/s or rad/s), (f) Cost function, (g) $\mathbf{I} - \mathbf{I}^*$ at the initial position, (h) $\mathbf{I} - \mathbf{I}^*$ at the end of the motion.

the previous experiment. This very nice behavior is due to the high redundancy of the visual features we use.

The goal of the last experiment is to show the robustness of the control law wrt the depths. For this purpose, a non planar scene has been used as shown on Fig. 7. It shows that large errors in the depth are introduced (the height of the castle tower is around 30 cm). The initial and desired poses are unchanged. Fig. 8 depicts this experiment. Here again, the control law still converges and the positioning error is still low since we have $\Delta\mathbf{r}$ = (0.2 mm, -0.2 mm, 0.1 mm, -0.01°, -0.02°, 0.01°).

VII. Conclusion and Future Works

We have shown in this paper that it is possible to use the luminance of all the pixels in an image as visual features in visual servoing. To the best of our knowledge this is the first time that visual servoing has been handled without any image processing (except the image spatial gradient required for the computation of the interaction matrix) nor learning step. To do that, we proposed a new control law since the classical ones used with geometrical features fail when using the luminance. Our approach has been validated on positioning tasks. The positioning error is very low. Supplementary advantages are that our approach is not sensitive to partial occlusions and to coarse approximations of the depths required to compute the interaction matrix. Finally, even if it has been computed in the Lambertian case, experiments on a non Lambertian object has shown that very low positioning errors can be reached.

Future work will concern target tracking. To do that, since the relative pose between the target and the lighting will be no more constant, an important issue will be in the determination of the interaction matrix in the non Lambertian case.

Fig. 7. The non planar scene.

References

[1] S. Benhimane and E. Malis. Homography-based 2d visual tracking and servoing. *Int. Journal of Computer Vision*, 26(7):661–676, July 2007. Special IJCV/IJRR issue on vision for robots.

[2] K. Deguchi. A direct interpretation of dynamic images with camera and object motions for vision guided robot control. *Int. Journal of Computer Vision*, 37(1):7–20, June 2000.

[3] K. Deguchi and T. Noguchi. Visual servoing using eigenspace method and dynamic calculation of interaction matrices. In *IEEE Int Conf on Pattern Recognition*, page A7E.3, 1996.

[4] K. Hashimoto and H. Kimura. LQ optimal and non-linear approaches to visual servoing. In K. Hashimoto, editor, *Visual Servoing*, volume 7, pages 165–198. World Scientific Series in Robotics and Automated Systems, Singapour, 1993.

[5] B.K.P. Horn and B.G. Schunck. Determining optical flow. *Artificial Intelligence*, 17(1-3):185–203, August 1981.

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

[7] J.-T. Lapresté and Y. Mezouar. A Hessian approach to visual servoing. In *IEEE/RSJ Int. Conf. on Intelligent Robots and System, IROS'04*, Sendai, Japan, October 2004.

[8] E. Malis. Improving vision-based control using efficient second-order minimization techniques. In *IEEE Int. Conf. on Robotics and Automation, ICRA'04*, volume 2, pages 1843–1848, New Orleans, April 2004.

[9] E. Marchand. Control camera and light source positions using image gradient information. In *IEEE Int. Conf. on Robotics and Automation, ICRA'07*, pages 417–422, Roma, Italia, April 2007.

[10] K. Miura, J. Gangloff, and M. De Mathelin. Robust and uncalibrated visual servoing without Jacobian using a simplex method. In *IEEE/RSJ Int. Conf. on Intelligent Robots and System, IROS'02*, Lausanne, Switzerland, October 2002.

[11] S.K. Nayar, S.A. Nene, and H. Murase. Subspace methods for robot vision. *IEEE Trans. on Robotics*, 12(5):750 – 758, October 1996.

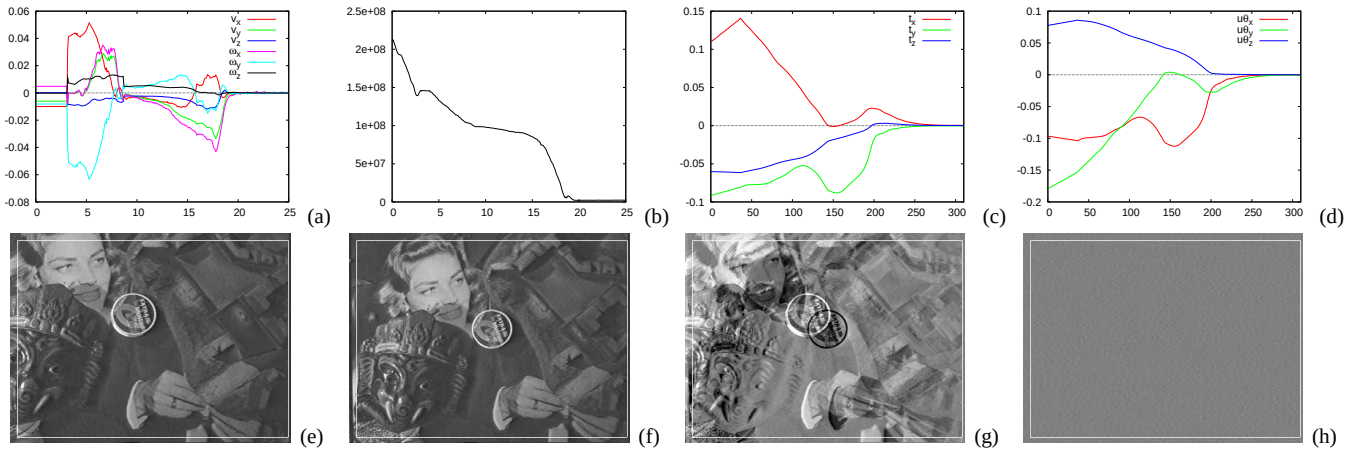


Fig. 8. Fourth experiment. Robustness wrt depths. MLM method (x axis in second for (a) and (b), frame number for (c) and (d)). (a) Camera velocities (m/s or rad/s), (b) Cost function, (c) Translational part of $\mathbf{\Delta r}$ (m), (d) Rotational part of $\mathbf{\Delta r}$ (rad.), (e) Initial image, (f) Final image, (g) $\mathbf{I} - \mathbf{I}^*$ at the initial position, (h) $\mathbf{I} - \mathbf{I}^*$ at the end of the motion.