Direct visual servoing based on multiple intensity histograms

Quentin Bateux, Eric Marchand

Direct visual servoing based on multiple intensity histograms

Quentin Bateux, Eric Marchand

Abstract— **Classically Visual servoing considered the regulation in the image of a set of visual features (usually geometric features). Recently direct visual servoing scheme, such as photometric visual servoing, have been introduced in order to consider every pixel of the image as a primary source of information and thus avoid the extraction and the tracking of such geometric features. Previous works proposed methods to use directly the image intensities in the definition of the control law, by using for example mutual information. In this paper, we propose a method to extend these works by using a global descriptor, namely intensity histograms, on the whole or multiple sub-sets of the images in order to achieve control of a 6 degrees of freedom (DoF) robot. The results are then demonstrated through experimental validations.**

I. INTRODUCTION

The goal of visual servoing is the control of the dynamic of a system by using visual information provided by one or multiple cameras [2]. The goal is to regulate an error defined in the image space to zero. Usually, as presented in [11][2][3] this error, to be minimized, is based on visual features that correspond to geometric features (points, lines,...) and requires extraction and matching of those features in the current and desired image. This real-time spatio-temporal tracking of these visual features [13] is a non trivial task and also one of the bottlenecks of the expansion of visual servoing.

Recently, to avoid this tracking and matching process, direct approaches have been proposed [4], where the feature matching and extraction step has been removed. Within this method, the control is built by minimizing the error between the current and desired images. Only pure photometric information (i.e., image intensity) is considered here. Although photometric visual servoing [4] is very efficient and very precise it may suffer from several drawbacks in non controlled environment. Since this is a pixel-wise comparison between the images it may be sensitive to illuminations variations and occlusions... leading to less precise positioning results. A way to improve this method is to use a global approach in order to work with a more compact representation of the visual information. To solve these problems, [9] proposes to dynamically adapt the reference image to the illumination conditions of the current image acquired by the camera. The reference image is then replaced by an expected image computed using a probabilistic expectation operator (that take into account illumination condition in both current and learned desired image). Another direct method [8] considered the mutual information as the similarity measure between desired and current images. This information theoretic approach allows to servo the camera in order to maximize the quantity of information shared by the current and desired image (that is the mutual information which is built by a measure of image entropy).

In this paper we propose to represent the image by intensity histograms and to build a control based on such visual features. Histograms have been widely considered in computer vision for tracking, matching, detection or images retrieval. In tracking, the Mean Shift tracking algorithm [6] and its sequel proved the feasibility of an histogram-based tracking scheme. A color histogram is used to describe the region to track. Based on the Bhattacharyya coefficient, tracking is accomplished by iteratively finding the local minima of the distance measure functions using the mean shift algorithm. Only histogram computation is considered. An extended version of the mean-shift, considering histogram and based on non-linear optimization, has also been proposed in [10]. In shape recognition or detection and image indexing, the use of methods that consider histogram based descriptors have benefited greatly to some applications. Histograms of gradients have been widely considered in key-points detection and matching method such as Sift [12] of pedestrian detection such as in [7]. Image indexation or retrieval algorithms, such as "Gist" [14], also consider histogram to globally describe the image.

In this paper, we extend the direct visual servoing approach by using gray-level intensity histograms. The basic idea is to compute the histogram of the the images acquired for the current and desired pose and control the camera in order to minimize an adequate distance between these two histograms. The idea is to use for our features a distance measure between histograms computed on both the images taken on initial and desired poses. This approach allows us to benefit from the properties of histograms, such as increased robustness to noise, and flexibility by choosing the number of classes considered in the computation of the histogram.

This paper is structured as follows. The next section provides elements in visual servoing, Section 3 develops the structure of the minimization algorithm involved in the presented method. Then Section 4 and 5 will present respectively the theoretical basis of the presented method, and then the experimental developments that validate the method.

II. DIRECT VISUAL SERVOING

A. Positioning Task

The aim of a positioning task is to reach a desired pose of the camera $\mathbf{r}^*$, starting from an arbitrary initial pose. To

Quentin Bateux and Eric Marchand are with Université de Rennes 1, IRISA, Inria Rennes, France, `Quentin.Bateux@irisa.fr`, `Eric.Marchand@irisa.fr`

achieve that goal, one needs to define a cost function that reflects, in the image space, this error. Most of the time this cost function f is an error measure which needs to be minimized. Considering the actual pose of the camera $\mathbf{r}$ the problem can therefore be written as an optimization process:

$$\widehat{\mathbf{r}} = \arg\min_{\mathbf{r}} \rho(\mathbf{r}, \mathbf{r}^*) \tag{1}$$

where $\widehat{\mathbf{r}}$, the pose reached after the optimization process (servoing process), is the closest possible to $\mathbf{r}^*$ (optimally $\widehat{\mathbf{r}} = \mathbf{r}^*$). For example, considering a set of geometrical features $\mathbf{s}$, the task will typically have to minimize the difference between $\mathbf{s(r)}$ and the desired configuration $\mathbf{s^*}$ which leads to:

$$\widehat{\mathbf{r}} = \arg\min_{\mathbf{r}}(\mathbf{s(r)} - \mathbf{s}^*). \tag{2}$$

This visual servoing task is achieved by iteratively applying a velocity to the camera. This requires the knowledge of the interaction matrix $\mathbf{L_s}$ of $\mathbf{s(r)}$ that links the variation of $\dot{\mathbf{s}}$ to the camera velocity and which is defined as:

$$\dot{\mathbf{s}}(\mathbf{r}) = \mathbf{L_s}\mathbf{v} \tag{3}$$

where $\mathbf{v}$ is the camera velocity.

This equation leads to the expression of the velocity that needs to be applied to the robot. The control law is classically given by:

$$\mathbf{v} = -\lambda\mathbf{L_s^+}(\mathbf{s(r)} - \mathbf{s}^*) \tag{4}$$

where λ is a positive scalar and $\mathbf{L_s^+}$ is the pseudo-inverse of the interaction matrix.

As pointed out in [5], such control is very similar to a Gauss-Newton minimization method which is widely considered to solve non-linear optimization problems. In this approach, the determination of the direction of descent will affect widely the performances of the servoing, as it will affect the radius of convergence of this method. In order to keep as much flexibility as we can, we will alternatively consider a control inspired from the Levenberg-Marquardt method, which can be expressed as:

$$\mathbf{v} = -(\mathbf{H} + \mu\,\mathrm{diag}(\mathbf{H}))^{-1}\mathbf{L_s^\top}(\mathbf{s(r)} - \mathbf{s(r}^*)) \tag{5}$$

with $\mathbf{H} = \mathbf{L_s^\top}\mathbf{L_s}$ where μ is a gain determining the behavior of the control law, from a steepest gradient behavior (μ small) to a Newton's behavior (μ large).

III. Histogram-based Visual Servoing

Histograms have been considered as a powerful way to provide a global description of a set of values, especially for tracking applications or image indexing where they have been used to perform comparison between two images (or part of images) in a global way. Since pixel-wise comparison (SSD) may not be robust enough in case of illumination offset or noise, such global descriptor appears to be an interesting choice.

A. Histogram: definition and overview

The main idea behind histogram computation is to classify each pixel of the image into a corresponding *bin*, according (in the case considered in this paper) to the intensity value of this pixel. Each bin consists of a scalar value that is incremented each time a corresponding element is found, and once each pixel of the image has been tested, the bins values will be normalized according to the number of pixels in the image. This leads to the following classical expression of an intensity histogram:

$$\mathbf{p_I}(i) = \frac{1}{N_\mathbf{x}} \sum_\mathbf{x}^{N_\mathbf{x}} \delta(\mathbf{I(x)} - i) \tag{6}$$

where $\mathbf{x}$ is a 2D pixel position in the image plane, the pixel intensity $i \in [0, 255]$ if we use images with 256 gray-levels, $N_\mathbf{x}$ the number of pixels in the image $\mathbf{I(x)}$, and $\delta(\mathbf{I(x)} - i)$ the Kronecker's function defined such as:

$$\delta(x) = \begin{cases} 1 & \text{if } x = 0 \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

$\mathbf{p_I}(i)$ is nothing but the probability for a pixel of the image $\mathbf{I}$ to have the intensity i.

B. Distance between histograms

As stated, our goal here is to find a way to use such histogram as a visual feature within a visual servoing control law. In order to use intensity histograms as our cues, we need to be able to compare them with each other. As defined in equation (1) a correlation function $\rho(.)$ has to be defined. The traditional way of doing this is to compare the histograms bin-wise, by using for example the Matusita distance, which can be expressed as [1]:

$$\rho(\mathbf{I}, \mathbf{I}^*) = \sum_i^{N_c} \left(\sqrt{\mathbf{p_I}(i)} - \sqrt{\mathbf{p_{I^*}}(i)} \right)^2 \tag{8}$$

where N_c is the number of bins considered in the histograms.

C. Computation of the interaction matrix

The main problem that arises with this formulation of the histogram is that the Kronecker's function ($\delta(.)$ used in equation (6)) is non-differentiable, which will prove problematic for our purposes, since we need to compute the interaction matrix [2] (or image Jacobian [11]) for this new error function (the Matusita distance given in equation (8)).
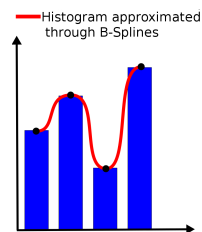

Fig. 1. Smoothing and approximating an histogram by using a B-spline

To solve this problem, as in [8], we choose here to use B-splines of at least second order (which can be differentiated

once) to approximate and smooth our bins , as illustrated in Fig. 1 . This technique yields the following formulation of an histogram:

$$\mathbf{p_I}(i) = \frac{1}{N_x} \sum_{\mathbf{x}}^{N_{\mathbf{x}}} \phi(\bar{\mathbf{I}}(\mathbf{x}) - \mathbf{i}) \tag{9}$$

where $N_{\mathbf{x}}$ is the number of pixels, $\mathbf{x} = \{x, y\}$ is a single image pixel, $\phi(.)$ is a B-spline differentiable once, $\bar{\mathbf{I}}$ is the image reduced to N_c intensity values (each histogram will then contain N_c bins). For 8 bits images, we have

$$\bar{\mathbf{I}}(\mathbf{x}) = \frac{N_c}{255} \mathbf{I}(\mathbf{x}).$$

A detailed description of B-Spline functions is given by Unser *et al.* in [15] but here are interesting properties of B-Spline in this case: the integral of the function is 1, so the result does not have to be re-normalized and the computation of the derivatives is easily obtained. In the following experiments a B-Spline of order 2 as been selected:

$$\phi(t) = \begin{cases} t+1 & \text{if } t \in [-1, 0] \\ -t+1 & \text{if } t \in [0, 1] \\ 0 & \text{else} \end{cases} \tag{10}$$

As shown in the previous part, the interaction matrix can be seen as the Jacobian of the feature vector according to each pose parameter $\mathbf{r} = \{t_x, t_y, t_z, r_x, r_y, r_z\}$. This definition leads to the following expression: $\mathbf{L}_\rho = \frac{\partial \rho(\mathbf{I}, \mathbf{I}^*)}{\partial \mathbf{r}}$ where $\mathbf{I}^*$ and $\mathbf{I}$ are the images recorded at the desired pose and at the current pose respectively, and $\rho(\mathbf{I}, \mathbf{I}^*)$ the Matusita distance between the two histograms.

Using chain rule derivation, we can obtain the following expressions:

$$\mathbf{L}_\rho = \frac{\partial}{\partial \mathbf{r}} \left[\sum_i^{N_c} \left(\sqrt{\mathbf{p_{\bar{I}}}(i)} - \sqrt{\mathbf{p_{\bar{I}^*}}(i)} \right)^2 \right] \tag{11}$$

$$= 2 \sum_i^{N_c} \left(\frac{\partial}{\partial \mathbf{r}} \sqrt{\mathbf{p_{\bar{I}}}(i)} \left(\sqrt{\mathbf{p_{\bar{I}}}(i)} - \sqrt{\mathbf{p_{\bar{I}^*}}(i)} \right) \right)$$

with

$$\frac{\partial}{\partial \mathbf{r}} \sqrt{\mathbf{p_{\bar{I}}}(i)} = \frac{1}{2\sqrt{\mathbf{p_{\bar{I}}}(i)}} \frac{\partial \mathbf{p_{\bar{I}}}(i)}{\partial \mathbf{r}} \tag{12}$$

where

$$\frac{\partial \mathbf{p_{\bar{I}}}(i)}{\partial \mathbf{r}} = \frac{\partial}{\partial \mathbf{r}} \left(\sum_{\mathbf{x}}^{N_{\mathbf{x}}} \left(\phi(\bar{\mathbf{I}}(\mathbf{r}, \mathbf{x}) - i) \right) \right)$$

$$= \sum_{\mathbf{x}}^{N_{\mathbf{x}}} \left(\frac{\partial}{\partial \mathbf{r}} \left(\phi(\bar{\mathbf{I}}(\mathbf{r}, \mathbf{x}) - i) \right) \right) \tag{13}$$

and from [8],

$$\frac{\partial}{\partial \mathbf{r}} \left(\phi \left(\bar{\mathbf{I}}(\mathbf{r}, \mathbf{x}) - i \right) \right) = \frac{\partial}{\partial i} \left(\phi \left(\bar{\mathbf{I}}(\mathbf{r}, \mathbf{x}) - i \right) \right) \frac{\partial \bar{\mathbf{I}}(\mathbf{r}, \mathbf{x})}{\partial \mathbf{r}}$$

$$= \frac{\partial}{\partial i} \left(\phi \left(\bar{\mathbf{I}}(\mathbf{r}, \mathbf{x}) - i \right) \right) \mathbf{L_{\bar{I}}} \tag{14}$$

with, from [5], the interaction matrix that links the variation of the image intensity to the camera velocity is given by:

$$\mathbf{L_{\bar{I}}} = - \left(\nabla_{\mathbf{x}} \bar{\mathbf{I}} \mathbf{L_x} + \nabla_{\mathbf{y}} \bar{\mathbf{I}} \mathbf{L_y} \right) \tag{15}$$

where $\mathbf{L_x}$ and $\mathbf{L_y}$ are the lines corresponding to the x and y-coordinates of $\mathbf{L_x}$ is the interaction matrix corresponding to a single 2D point $\mathbf{x} = (x, y)$ in the image, and is defined in [2] as:

$$\mathbf{L_x} = \begin{bmatrix} \frac{-1}{Z} & 0 & \frac{x}{Z} & xy & -(1+x^2) & y \\ 0 & \frac{-1}{Z} & \frac{y}{Z} & 1+y^2 & -xy & -x \end{bmatrix}$$

Finally, after simplification, we get the following $\mathbf{L}_\rho$ interaction matrix for our control law:

$$\mathbf{L}_\rho = \sum_i^{N_c} \left(\frac{1}{N_{\mathbf{x}}} \left(1 - \frac{\sqrt{\mathbf{p_{\bar{I}^*}}(\mathbf{i})}}{\sqrt{\mathbf{p_{\bar{I}}}(\mathbf{i})}} \right) \sum_{\mathbf{x}}^{N_{\mathbf{x}}} \left(\frac{\partial}{\partial i} \left(\phi(\bar{\mathbf{I}}(\mathbf{r}, \mathbf{x}) - i) \right) \mathbf{L_{\bar{I}}} \right) \right)$$

D. Using Kernels in histogram

In works such as tracking problems [6], [10], that are related to our current problematic, a weighting kernel function as been successfully introduced in the computation of the histogram, in order to give more weight to the pixels in the center of the image and less to those on the borders, as the latter are less prone to contain relevant information. The use of this kernel can be here justified since some pixels will enter the camera field of whereas other will disappears. The kernel will then allow to give more importance to the pixels at the center of image. The histogram computation expression becomes then:

$$\mathbf{p_{\bar{I}}}(i) = C \sum_{\mathbf{x}}^{N_{\mathbf{x}}} K(\frac{\mathbf{x} - \mathbf{c}}{h}) \delta(i - \bar{\mathbf{I}}(\mathbf{x})) \tag{16}$$

with C a normalization constant such as $C = \frac{1}{N_x \sum_{\mathbf{x}}^{N_{\mathbf{x}}} K(\frac{\mathbf{x}-\mathbf{c}}{h})}$, and K an isotropic kernel of bandwidth h, centered on $\mathbf{c}$, the center of the image. In our application, we will take K as a gaussian kernel, expressed such as:

$$K(\frac{\mathbf{x} - \mathbf{c}}{h}) = e^{-\frac{||\mathbf{x}-\mathbf{c}||_2^2}{h}}. \tag{17}$$

E. Multiple kernels extension

As expected from previous work using intensity histograms distances [6], processing a single histogram on the whole image fails to control more than translations on the x/y axis. A solution that as been proposed in tracking to extend the sensibility of histogram-based method to more degrees of freedom has been to use multiple kernels throughout the image, as in [10]. Here we adapt this idea by dividing the image in multiple areas and associating a histogram to each area, then stacking the resulting error vectors and interaction matrix. The Fig. 2 shows an example of the application of this method using 9 kernels on the image : each sub-part will be used to compute its own histogram, and thus giving us a new distance to use as a distinct feature when we compare to the 9 histograms computed on the desired pose image. The global interaction matrix then becomes:

$$\mathbf{L}_\rho = \begin{bmatrix} \mathbf{L}_{\rho_1} & \mathbf{L}_{\rho_2} & \dots & \mathbf{L}_{\rho_n} \end{bmatrix}^{\mathbf{T}} \tag{18}$$

where $\mathbf{L}_{\rho_i}$ is the interaction matrix given by the i-th histogram distance, using the control law defined in the previous section. By using enough kernels on the image, it becomes them possible to control every six degrees of freedom of the robot, as it will be shown in the next section.

After several experiments, the separation of the image into 5 horizontally and vertically (25 sub-parts) proved to be an adequate choice, as less areas can prove problematic to a good decoupling of the DoF, and more can leads to increasing the sensibility to small changes in the scene and thus to an harder determination of the parameters to be tuned (mainly regarding the determination of the gain and the number of bins in the histograms).

Fig. 2. Illustration of multi-kernels approach with 2 horizontal and vertical separations (9 areas)

Regarding the effect of the multiple kernel approach on the cost function, Fig. 3 shows the mapping of the cost function in the case where we use only one kernel on the whole image to calculate our histogram. It can be seen that even for the two DoF t_x and t_y, there is no clear minimum, which can only prevent our system to reach the desired pose. On the other hand, if we separate the image into 5 along x and y, and if we create a distance feature from each of the generated sub-parts, the cost function show a clear minimum, with a rather large valley, as seen on the Fig. 4.
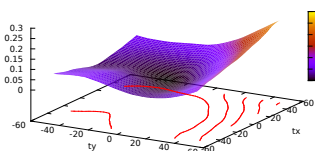
Fig. 3. Cost function map for single kernel approach

IV. EXPERIMENTAL RESULTS

To validate the proposed approach, several experiments have been performed using a camera mounted on a 6 DOF gantry robot. Independently from the experiment, the computation time remains low. The control law is computed at video rate. A velocity is computed and sent to the robot every 40 ms for a 320 × 240 input image using a 2.4-GHz computer. All implementation are done using the ViSP library [13].
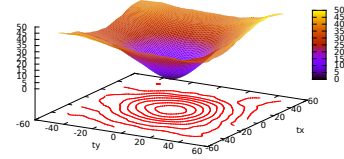
Fig. 4. Cost function map for 25 kernels approach

In order to validate our approach, we have done several experiments to test the overall efficiency of this method. The goal of these experiments is always the same, we indicate a desired pose to the robot by taking the associated image, then we move the robot to another pose, and the camera velocities are computed at nearly video rate according to the proposed control law in order to reach the desired pose. Four main experiments have been performed, testing this method in nominal conditions (with 4 DoF in the preliminary simulation and 6 DoF on a real robot), in the case of non-planar scene and in the presence of occlusions added during the experiment.

1) Simulation for a 4 DoF positioning task: For the first experiment, in order to validate the proposed control law, we consider a 4 DoF positioning task in simulation. Simulated image of a planar environment is provided by the rendering engine provided in ViSP [13]. In this test case, the camera is set at a distance of 0.2m of the image plane, and the initial offset in the camera position was $r_{init} = (1cm, -1cm, -2.5cm, 0°, 0°, 10°)$. As shown in the Fig. 5, the control law succeeds to converge, allowing further testing on the more complex 6DoF test case, applying the control law to the servoing of an actual robot.
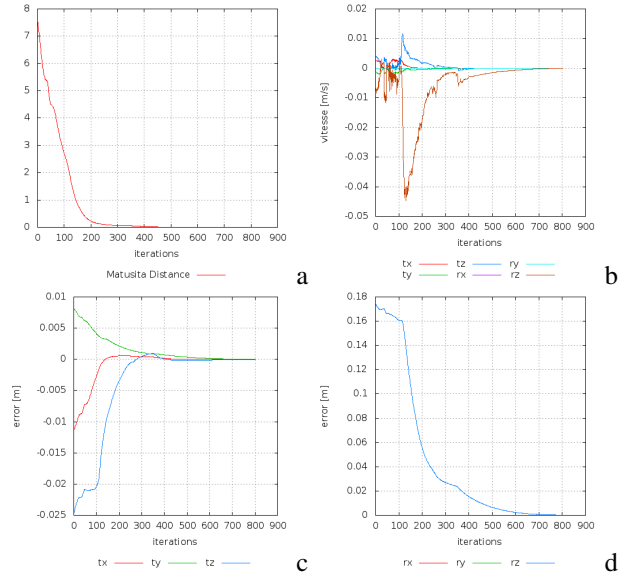
Fig. 5. Simulation with 4 DoF. Camera velocities in m/s and rad/s in (a). (b) Matusita distance. (c) Positioning error: translational part of $\triangle r$, (d) rotational part of $\triangle r$.

2) 6 DoF positioning task Nominal conditions: For the second experiment, we used a planar scene. The depth has been considered as a constant during all the experiments with Z = 0.80m in equation (16). The initial displacement with respect to the desired position (which is parallel to the scene plane) was chosen in this experiment was $r_{init} = (-2cm, 16cm, 6cm, 16°, 1°, 0°)$. Fig. 6 shows the results of this experiment. It is interesting to note that with this initial pose, the plane of the picture and the one of the camera are not parallel, which leads to non-uniform relative depth for the pixel of the image acquired. Although the depth is assumed to be constant, our system exhibits a correct behavior, despite this strong hypothesis in the computation of the interaction matrix. The Fig. 6(a) depicts the velocities of the camera ; Fig. 6(b) the evolution of the Matusita distance ; Fig. 6(c) and 6(d) the pose errors (resp. the translational and rotational parts) between r and r^* ; Fig. 6(e), 6(f) show the initial and final image acquired by the camera ; Fig. 6(g), 6(h) depicts the initial and final values of $\mathbf{I} - \mathbf{I}^*$. The convergence shows good properties regarding the evolution of the Matusita distance, with a satisfying exponential decrease, which can be found also in the evolution of the pose errors.

3) Non-planar scene: In this experiment, we test the sensibility that can be induced by our hypothesis of planar scene by introducing a 3D object inside the scene. The object used can be seen in Fig 7(e) and 7(f). The initial pose that was chosen in this experiment was $r_{init} = (-1cm, 0cm, -10cm, 0°, 0°, -19°)$, which is also a way the test the sensibility of the constant depth that is used in our interaction matrix. Fig. 7 show the result of this experiment, and we can see that all the desirable properties of the first experiment are kept, proving the robustness of our method regarding the depths approximations, In this experiment we also considered only 16 bins in the histograms, which enlarges the converging area.

4) Occlusions during positioning: In this last experiment, we introduce an object that is external to the scene during the positioning process. This introduces an significant dissimilarity between the desired image and the one that has actually to be reached. Since it introduces a significant change in the intensity levels of many pixels, it also show that the proposed method is robust to local changes. The initial pose that was chosen in this experiment was $r_{init} = (-2cm, 16cm, 6cm, 16°, 1°, 0°)$, the same one as in the second experiment, in order to allow comparison in the behavior. The Fig. 8(b) displays the moment when we add an object in the scene. This object can clearly be seen in the Fig. 8(f) and 8(h). We can see that the system still converges, despite noisier computed velocities and trajectories.

5) Increasing the convergence range: In order to increase the convergence area in the case of larger noises such as changes in illumination or wider occlusions, experiments has shown that using a Gaussian filtering as preprocessing step provides significant stabilization of this method in most cases, but at the cost of the final precision. Still, this could prove a valuable solution if it were to be coupled with our current technique, based on unfiltered images, inside a two-
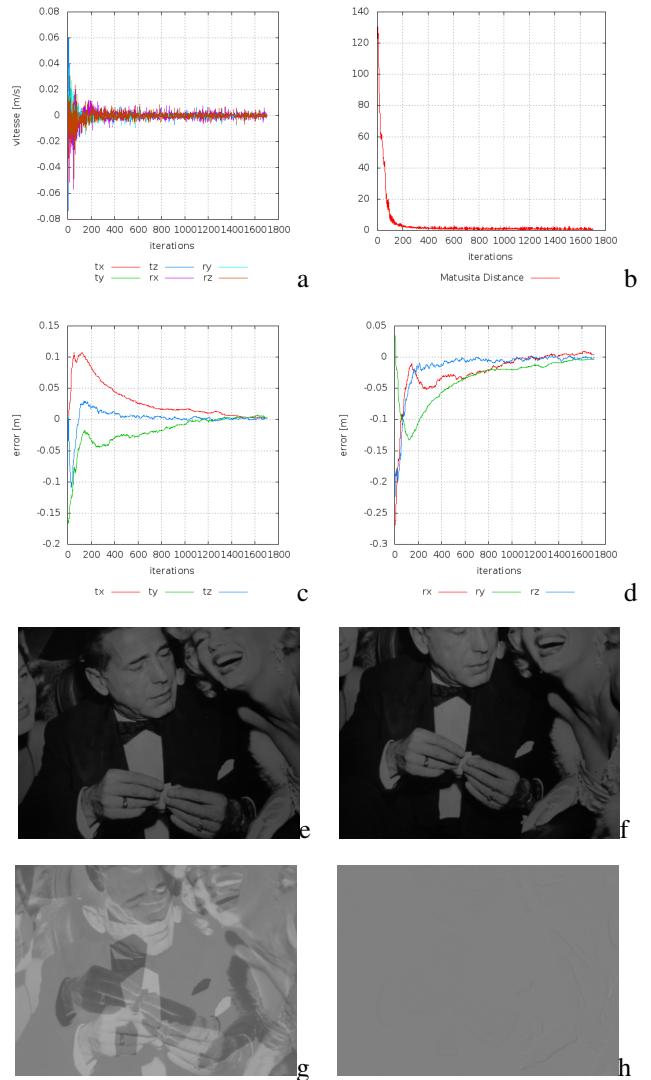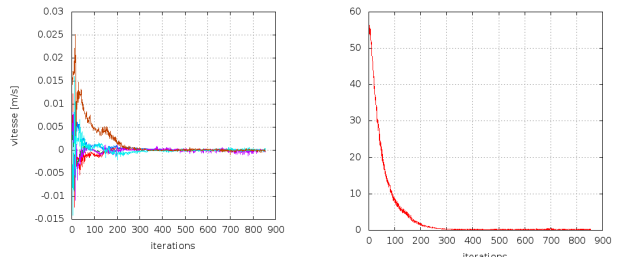

Fig. 6. Nominal conditions. Camera velocities in m/s and rad/s in (a). (b) Matusita distance. (c) Positioning error: translational part of $\triangle r$, (d) rotational part of $\triangle r$. (e) Initial image (128 gray levels used). (f) Desired image (128 gray levels used). (g) I - I* at initial position. (h) I - I* at the end of the motion

step method, where the first step could bring the system in the converging area, and then switching in order to get good final precision.

V. CONCLUSION AND PERSPECTIVES

In this paper, we have shown that it was possible to extend the current direct photometric visual servoing method by using more compact global descriptors: intensity histograms. We proposed a new control law that minimizes the Matusita distance between histograms. Interaction matrix related to this error function has been exhibited. We validated this approach for 6 DoF positioning tasks on various scene with planar and non-planar scenes. Future works will concern the elaboration of control laws based on more complex and more resilient global descriptors such as color histograms or histograms of oriented gradients [7].
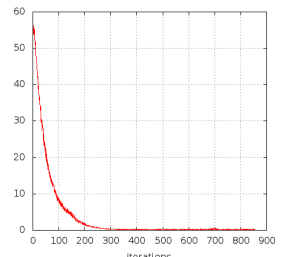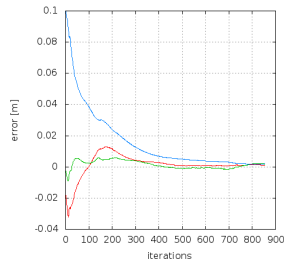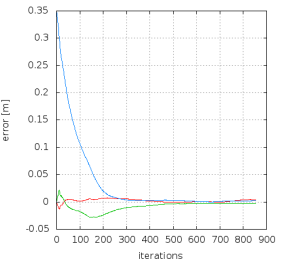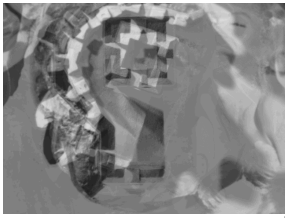
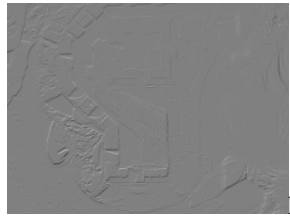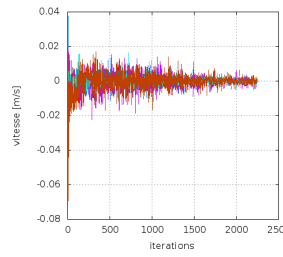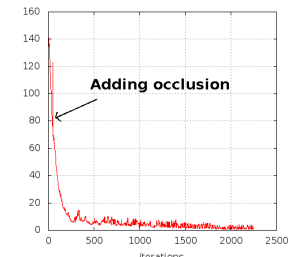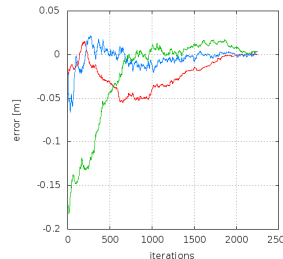Fig. 7. Positioning task with respect to non-planar scene. Camera velocities in m/s and rad/s in (a). (b) Matusita distance. (c) Positioning error: translational part of △r, (d) rotational part of △r. (e) Initial image (16 gray levels used). (f) Desired image (16 gray levels used). (g) I - I* at initial position. (h) I - I* at the end of the motion
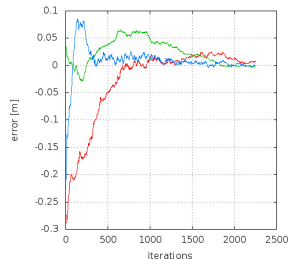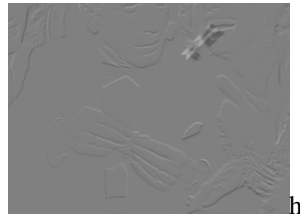

Fig. 8. Occlusion during camera motion. Camera velocities in m/s and rad/s in (a). (b) Matusita distance. (c) Positioning error: translational part of △r, (d) rotational part of △r. (e) Initial image (128 gray levels used). (f) Desired image (128 gray levels used). (g) I - I* at initial position. (h) I - I* at the end of the motion

REFERENCES

[1] S.-H. Cha and S. Srihari. On measuring the distance between histograms. *Pattern Recognition*, 35(6):1355–1370, 2002.

[2] F. Chaumette and S. Hutchinson. Visual servo control, Part I: Basic approaches. *IEEE Robotics and Automation Magazine*, 13(4):82–90, December 2006.

[3] F. Chaumette and S. Hutchinson. Visual servo control, Part II: Advanced approaches. *IEEE Robotics and Automation Magazine*, 14(1):109–118, March 2007.

[4] C. Collewet and E. Marchand. Photometric visual servoing. *IEEE Trans. on Robotics*, 27(4):828–834, August 2011.

[5] C. Collewet, E. Marchand, and F. Chaumette. Visual servoing set free from image processing. In *IEEE Int. Conf. on Robotics and Automation, ICRA'08*, pages 81–86, Pasadena, CA, May 2008.

[6] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *IEEE Trans. on PAMI*, 25(5):564–577, May 2003.

[7] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Conf. on Computer Vision and Pattern Recognition, CVPR'05*, pages 886–893, 2005.

[8] A. Dame and E. Marchand. Mutual information-based visual servoing. *IEEE Trans. on Robotics*, 27(5):958–969, October 2011.

[9] B. Delabarre and E. Marchand. Visual servoing using the sum of conditional variance. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'12*, pages 1689–1694, Vilamoura, Portugal, October 2012.

[10] G. Hager, M. Dewan, and C. Stewart. Multiple kernel tracking with ssd. In *IEEE Conf. on Computer Vision and Pattern Recognition, CVPR'04*, pages 790–797, June 2004.

[11] 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.

[12] D. Lowe. Distinctive image features from scale-invariant keypoints. *Int. Journal of Computer Vision*, 60(2):91–110, 2004.

[13] E. Marchand and F. Chaumette. Feature tracking for visual servoing purposes. *Robotics and Autonomous Systems*, 52(1):53–70, June 2005. special issue on "Advances in Robot Vision", D. Kragic, H. Christensen (Eds.).

[14] A. Oliva and A. Torralba. Building the gist of a scene: The role of global image features in recognition. *Progress in brain research*, 155:23–36, 2006.

[15] M. Unser, A. Aldroubi, and M. Eden. B-spline signal processing: Part i-theory. *IEEE Trans. Signal Processing*, 41(2):821–833, February 1993.