I R I S A
INSTITUT DE RECHERCHE EN INFORMATIQUE ET SYSTÈMES ALÉATOIRES

PUBLICATION
INTERNE
N° 1698

REAL TIME PLANAR STRUCTURE TRACKING: A CONTOUR AND TEXTURE APPROACH

MURIEL PRESSIGOUT AND ÉRIC MARCHAND

ISSN 1166-8687

IRISA
CAMPUS UNIVERSITAIRE DE BEAULIEU - 35042 RENNES CEDEX - FRANCE

INSTITUT DE RECHERCHE EN INFORMATIQUE ET SYSTÈMES ALÉATOIRES
Campus de Beaulieu – 35042 Rennes Cedex – France
Tél. : (33) 02 99 84 71 00 – Fax : (33) 02 99 84 71 71
http://www.irisa.fr

Real Time Planar Structure Tracking: a Contour and Texture Approach

Muriel Pressigout * and Éric Marchand **

Systèmes cognitifs
Projet Lagadic

Publication interne n° 1698 — Mars 2005 — 23 pages

Abstract:

Robustness and accuracy are major issues in real-time object tracking in image sequences. This paper describes a reliable tracking for markerless objects based on the fusion of visual cues and on the estimation of a 2D transformation. The parameters of this transformation are estimated using a non-linear minimization of a unique criterion that integrates information on both texture and edges of the tracked object. The proposed tracker is then more robust and succeeds in conditions where methods based on a single cue fail. The tracker can deal with polygonal shaped objects but also with those which can be modeled by a B-spline. In the latter case, NURBS are used to reduce time processing. The efficiency and the robustness of the proposed method are tested on image sequences as well as during image-based visual servoing experiments.

Key-words: edge-based tracking, template matching, robust motion estimation

(Résumé : tsvp)

* Muriel.Pressigout@irisa.fr
** Eric.Marchand@irisa.fr

Centre National de la Recherche Scientifique
(UMR 6074) Université de Rennes 1 – Insa de Rennes

Institut National de Recherche en Informatique
et en Automatique – unité de recherche de Rennes

Suivi temps-réel d'objets planaires: approche contour et texture

Résumé : Le suivi temps-réel d'un objet dans une séquence d'images reste un problème sensible au niveau de la précision des résultats et de la prise en compte d'occultations. La méthode décrite dans cet article permet un suivi plus efficace d'objets planaires sans utiliser de marqueurs spécifiques. Elle se base sur la fusion d'informations visuelles et sur l'estimation d'une transformation 2D. Les paramètres de cette transformation sont estimés par une minimisation itérative d'un critère hybride qui intègre à la fois des informations sur la texture et sur le contour de l'objet suivi. L'algorithme est alors plus robuste et permet d'achever un suivi correct quand l'utilisation d'un seul type d'information n'aurait pas suffit à obtenir un résultat satisfaisant. Ce suivi hybride a été développé pour des objets dont le contour peut être modélisé par des lignes mais aussi par une B-Spline. Dans ce dernier cas, l'implémentation est réalisée en utilisant les NURBS pour diminuer fortement le temps de calcul. L'efficacité de ce suivi a été testée sur des séquences d'images mais aussi lors d'expériences d'asservissement visuel avec une caméra montée sur un robot.

Mots clés : suivi de contour, template matching, estimation du mouvement robuste

1 Introduction

Object tracking in image sequences is an important issue for research and applications related to visual servoing and more generally for robot vision. A robust and real-time spatio-temporal tracking process of visual cues is indeed one of the keys to success of a visual servoing task. Fiducial markers have been used for a long time since they ensure reliable and fast tracking. However, as these features are not present in realistic environments, it is no longer possible to be limited to such techniques. For the time-being, most of the available tracking techniques can be divided into two main classes: edge-based and texture-based tracking. The former approach focuses on tracking 2D or 3D features such as geometrical primitives (points, segments, circles,...), object contours, 3D object, *etc.* The latter explicitly uses the texture or the luminance information that represents the tracked object.

Edge-based trackers rely on the high spatial gradients outlining the contour of the object or some geometrical features of its pattern (points, lines, circles, distances, splines,...). When 2D tracking is considered, such edge points enable to estimate the geometrical features parameters whose values define the position of the object [12]. Snakes or active contours are also based on high gradients and can be used to outline a complex shape [3]. If a 3D model of the object is available [7, 8], edge-based tracking is closely related to the pose estimation problem and is therefore suitable for any visual servoing approach. This implicit 3D information improves robustness and performance by being able to predict the hidden movement of the object and acts to reduce the effects of outlier data in the tracking process. As only planar structures are considered in this paper, these latter methods that require a 3D model of the scene are not further detailed. In general, edge-based techniques have proved to be very effective for applications that require a fast tracking process. On the other hand, they may fail in the presence of highly textured environments.

Previous approaches rely mainly on the analysis of intensity gradients in the images. When the scene is too complex (due for example to the presence of high texture or to the lack of specific object contours), other approaches are required. Another possibility is to directly consider the brightness values and to perform 2D matching on an area (a part of the image or in some cases the whole image) without any feature extraction: we then refer to template-based tracking or motion estimation (according to the problem formulation) [11, 22]. The goal of such algorithms is to estimate a set of parameters that describes the transformation or the displacement of the considered area by minimizing a given correlation criterion. It is possible to solve this problem using efficient minimization techniques that are able to consider quite complex 2D transformations (such as affine or homographic motions). Such an approach has been proposed in [11]. In this work, Hager and Belhumeur define an Jacobian matrix that links the variation of the motion parameters to the variation of the brightness value. An extension of this approach has been proposed in [16] where the pseudo-inverse of the Jacobian matrix is learned offline. Whereas in [11] an affine motion estimation is considered, homography estimation is considered in [2, 4, 16]. Furthermore, [2] uses a second order minimization based on Lie Algebra in order to speed up the minimisation process. [20, 21] propose to update the texture template used in such methods to improve

the tracking. Let us note that these methods are closely related to classical image motion estimation algorithms [22]. Such tracking techniques are also fast and reliable when a "good" texture is present on the tracked object but fail otherwise.

One can notice that these two classes of approaches have complementary advantages and drawbacks. In order to develop algorithms robust to aberrant measurements and to potential occlusions, it is interesting to take into account visual information related to these different types, *i.e.* exploiting either edge-based or texture-based features to track the object. Some approaches rely on probabilistic frameworks. In [24], the authors consider a texture-based approach to find the projected contour of a 3D object. The standard gradient-based detection is substituted by a method which computes the most probable location of the texture boundary. Some classical single-cue trackers, such as the condensation algorithm, have been extended to multi-cue tracking [15]. 2D visual cues fusion using voting has also been studied in [17] and considered for visual servoing applications. However, this work is not directly related to edge and texture fusion. Some other methods use sequentially the edge-based tracker and the texture-based one, in order to combine robustness and accuracy, as in [1, 6, 18]. However in these cases motion estimation is mostly used to achieve a better position of the edge (and therefore to ensure the robustness of the tracker). Thus these latter approches do not take benefit of several advantages from using them simoultaneously, like dealing with a wider range of objects and larger motions. Let us note that in a different context (using 3D model), edge and texture information have also been fused for pose computation in [26] where a model-based approach that considers both 2D-3D matching against a keyframe and 2D-2D matching temporal matching is proposed.

The method presented in this paper integrates simultaneously both contour and texture. Our goal was to define a unique state vector that describes both the appearance of the template as well as its edge boundaries. Considering this state vector, we are able to compute the parameters of a 2D transformation that minimizes the error between a current multi-cue template and the displaced reference one. Both edges and texture tracking algorithms can be seen as optimization algorithms: the hybrid tracking algorithm fuses the motion estimation of edge locations and texture points in an unique non-linear minimization process. A similar approach has been proposed in [19]. In this latter work, the template matching algorithm is handled using the texture-based tracking algorithm proposed in [16] (the Jacobian is learned offline) whereas ours uses an explicit formulation of the Jacobian [11]. Furthermore, although the edge and texture points are exploited in a similar manner, the feature selection is different. In [19], edge and texture points are classified according to the eigen-values of the signal autocorrelation matrix. Sharp edges of the texture are then likely to be classified as edge points. As a consequence, the remaining points that are classified as texture points hold little information since they belong to smooth gradient varying regions. As it will be further described, we choose another classification that is to our point of view a better representation of the object: the edge location are evenly sampled along the geometrical features outlining contours of the object and the texture points are the sharp edges of the object pattern. As said in [25], better features enables a better tracking.

The paper is organized as follows. The tracker is presented in Section 2. Subsection 2.1 describes the general framework of the object tracking which is based on a 2D transformation estimation and the 2D transformation estimation is developped in Subsection 2.2. Since data are likely to be corrupted with noise, M-estimation is introduced in the minimization process via an iteratively re-weighted least squares implementation as explained in Subsection 2.3. Details about the edge-based and texture-based features are given in Subsection 2.4. Finally, several experiments in Section 3 will illustrate the behavior of the tracker on real image sequences and also during visual servoing experiment.

2 Hybrid tracking based on 2D transformation estimation

2.1 2D transformation

The tracking of the object is performed by estimating the 2D transformation that best describes its motion in the image. This transformation can be described by M parameters stored in a vector θ. The 2D transformation θ_t of the object between the first image $\mathbf{I}^0$ and image $\mathbf{I}^t$ is such that, if $\mathbf{x}_0 = (x_0, y_0)$ is a point in $\mathbf{I}^0$ belonging to the object and $\mathbf{x}_{\theta_t}$ its corresponding point in $\mathbf{I}^t$, then :

$$\mathbf{x}_{\theta_t} = \mathbf{\Psi}_{\theta_t}(\mathbf{x}_0) \tag{1}$$

where $\mathbf{\Psi}_{\theta_t}$ is the 2D transformation described by θ_t (see Figure 1). We note θ_t the current values of the parameters and $\widehat{\theta}_t$ their estimated values.

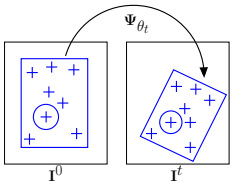


Figure 1: Estimating the object motion

Since planar structures are considered, the 2D transformation is an homography. $\mathbf{x}$ being expressed in homogeneous coordinates, one has[1,2] :

$$\mathbf{x}_{\theta_t}^{h\top} \sim \begin{pmatrix} \theta_0 & \theta_1 & \theta_2 \\ \theta_3 & \theta_4 & \theta_5 \\ \theta_6 & \theta_7 & \theta_8 \end{pmatrix} \mathbf{x}_0^{h\top} \tag{2}$$

[1] $\sim$ is the notation for the equality up to scale.
[2] the superscript h is added when the homogeneous coordinates are used

Therefore the parameters to be estimated are :

$$\theta_t = \left(\ \theta_0, \theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6, \theta_7, \theta_8 \ \right) \tag{3}$$

It is not required to choose a specific representation for the homography (for example setting θ_8 to 1) since the method proposed in this paper is invariant to the scale factor.

2.2 Transformation estimation

Let $\mathbf{m} = (m^1, \ldots, m^N)$ denote the vector of dimension N that stores the value of the N visual features. $\mathbf{m}^i$ can be either a brightness value or a distance between a image point and a 2D geometrical feature, *etc.* Its value measured in $\mathbf{I}^t$ is noted $\mathbf{m}_t = (m_t^1, \ldots, m_t^N)$ and its current value in $\mathbf{I}^t$ estimated according to the 2D transformation estimation θ_t is denoted by $\mathbf{m}_{\theta_t} = (m_{\theta_t}^1, \ldots, m_{\theta_t}^N)$.

The objective of our method is to estimate the 2D transformation that verifies (1). This is achieved by minimizing the error between the current value $\mathbf{m}_{\theta_t}$ and its value $\mathbf{m}_t$ observed in the current image $\mathbf{I}^t$:

$$\widehat{\theta}_t = \underset{\theta_t}{\operatorname{argmin}} \ \| \ \mathbf{m}_{\theta_t} - \mathbf{m}_t \ \|^2 \tag{4}$$

One has $\widehat{\theta}_t = \widehat{\theta}_{t-1} + \widehat{\mu}_t$, $\widehat{\theta}_{t-1}$ being the 2D transformation parameters estimated for the previous image from the first image. The problem is then to estimate the value $\widehat{\mu}_t$ that minimizes :

$$\widehat{\mu}_t = \underset{\mu_t}{\operatorname{argmin}} \ \underbrace{\| \ \mathbf{m}_{\widehat{\theta}_{t-1}+\mu_t} - \mathbf{m}_t \ \|^2}_{\mathbf{e}} \tag{5}$$

The error $\mathbf{e}$ as defined in (5) is regulated to zero by updating $\widehat{\mu}_t$ with an iterative minimization process based on a first order approximation. For the k-th iteration, one has:

$$\begin{array}{rrcl} & \widehat{\mu}_{t(k)} & = & \widehat{\mu}_{t(k-1)} + \widehat{\delta\mu}_{(k)} \\ \text{with} & \widehat{\delta\mu}_{(k)} & = & -\lambda \mathbf{J}^{+}_{\mathbf{m}_{\theta_{t\,(k-1)}}} \mathbf{e} \end{array} \tag{6}$$

where $\theta_{t(k-1)} = \widehat{\theta}_{t-1} + \widehat{\mu}_{t(k-1)}$. The subscript k that denotes the iteration number is suppressed from now to simplify the notations. $\mathbf{J}_{\mathbf{m}_{\theta_t}}$ is the Jacobian matrix of $\mathbf{m}$ with respect to the M current 2D transformation parameters θ_t. It is a $N \times M$ matrix storing the N Jacobian matrices $\mathbf{J}_{\mathbf{m}^i_{\theta_t}}$ of each visual feature $\mathbf{m}^i_{\theta_t}$:

$$\mathbf{J}_{\mathbf{m}_{\theta_t}} = \begin{bmatrix} \mathbf{J}_{\mathbf{m}^1_{\theta_t}} \\ \ldots \\ \mathbf{J}_{\mathbf{m}^N_{\theta_t}} \end{bmatrix} \qquad \text{with} \qquad \mathbf{J}_{\mathbf{m}^i_{\theta_t}} = \frac{\partial \mathbf{m}^i_{\theta_t}}{\partial \theta_t} \tag{7}$$

Since the tracking must be as fast as possible, an interesting choice for the Jacobian matrix is the one computed at the first iteration of the minimization process. Since the 2D transformation is small between images, it is an sufficient solution. In the result section, it will be shown that even large motions can be handled with this choice.

2.3 Robust estimation

The minimization process is sensible to outliers that can be due to occlusions or noise. M-estimators are therefore introduced in (6) to eliminate these data:

$$\widehat{\delta\mu} = -\lambda(\mathbf{D}\mathbf{J}_{\mathbf{m}_{\theta_t}})^{+}\mathbf{D}\mathbf{e} \tag{8}$$

where $\mathbf{D}$ is a $N \times N$ diagonal matrix such as:

$$\mathbf{D} = diag(w_1, ..., w_N) \tag{9}$$

The N weights w_i reflect the confidence in each visual feature $m^i_{\theta_t}$ and are usually given by [13]:

$$w_i = \frac{\psi(\delta_i/\sigma)}{\delta_i/\sigma} \tag{10}$$

with $\psi(u)$ the influence function and δ_i the normalized residue given by $\delta_i = \Delta_i - Med(\Delta)$ (where $Med(\Delta)$ is the standard deviation of the inlier data). Various influence functions have been studied in the literature. The Tukey's hard re-descending function is considered here since it completely rejects outliers.

2.4 Hybrid features

The approach described above is valid for any visual feature $\mathbf{m}$ as long as the associated Jacobian matrix $\mathbf{J}_{\mathbf{m}}$ is available. This provides a general framework to fuse different kinds of visual features. Thus one can take benefit simoultaneously from the advantages of each one to enlarge the convergence area of the tracker. Both edge-based and texture-based features are exploited in our hybrid tracker. Each of them has different properties in the tracking process. The first ones rely on the edge locations extracted in the current image, the second ones on the brightness values of the pattern.

The edges extraction gives helpful information about the object location in the image It relies on the current image analysis and therefore is less dependent on the past than the features based on the brightness value. The use of the edge-based features enables the current contour $\mathcal{C}_{\theta_t}$ estimated using parameters θ_t to lie on the edge locations extracted in the current image. As it will be further detailed in part 2.4.1, the edge-based features are point-to-contour distance $d_{\perp}\big(\mathcal{C}_{\theta_t}, \mathbf{x}^i_t\big)$. Using only such features is equivalent to estimate θ_t exploiting only the edge information (see Figure 2(a)).

The texture-based features are the brightness values $I^t(\mathbf{x}^i_{\theta_t})$ sampled at the texture points $\mathbf{x}^i_{\theta_t}$. The tracking then depends on past information (a reference template). The details about these features are given in part 2.4.2. Using only such features is equivalent to estimate θ_t exploiting only the pattern information (see Figure 2(b)).

When the two kinds of feature are used, exploiting both information extracted in the current image (edge locations) and information linking the past and the current image

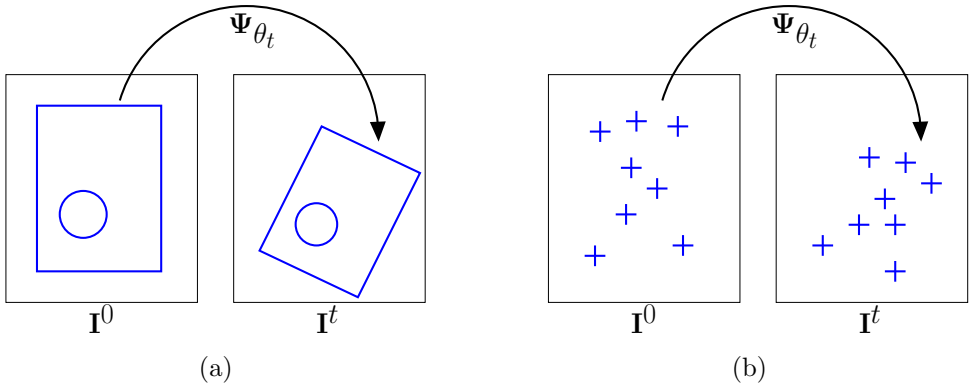


Figure 2: (a) Estimating the object motion using edge information,
(b) Estimating the object motion using texture information

(brightness value) enables an efficient spatio-temporal tracking. If there are N_c edge locations and N_t texture points, the feature vector $\mathbf{m}$ will be of size $N = N_c + N_t$, storing both of them :

$$\mathbf{m}_{\theta_t} = (m^1_{\theta_t}, ..., m^{N_c}_{\theta_t}, m^{N_c+1}_{\theta_t}, ..., m^{N_c+N_t}_{\theta_t}) \tag{11}$$

where $\left(m^i_{\theta_t}\right)_{i \leq N_c}$ is the feature associated with the i-th edge location and $\left(m^i_{\theta_t}\right)_{N_c < i = N_c + j}$ is the feature associated with the j-th texture point, *i.e.* :

$$m^i_{\theta_t} = \begin{cases} d_\perp\left(\mathcal{C}_{\theta_t}, \mathbf{x}^i_t\right) & \text{if} \quad i \leq N_c \\ I^t(\mathbf{x}^i_{\theta_t})) & \text{if} \quad i > N_c \end{cases} \tag{12}$$

The error associated with a texture point (brightness value) and the one associated with the edge locations (point-to-contour distance) are of a different order of magnitude. Therefore a normalization must be performed to take into account the information given by the different cues. The weights in (9) are now :

$$w'_i = \begin{cases} \frac{w_i}{max_{j=1...N_c}(|\mathbf{e}_j|)} & \text{if} \quad i \leq N_c \\ \frac{w_i}{max_{j=N_c+1...N}(|\mathbf{e}_j|)} & \text{if} \quad i > N_c \end{cases} \tag{13}$$

where $max_{j=1...N_c}(|\mathbf{e}_j|)$ (resp. $max_{j=N_c+1...N}(|\mathbf{e}_j|)$) is the maximal absolute value stored in the point-to-contour distance (resp. brightness difference) vector and w_i is the weight computed by the M-estimators.

The two following subsections are dedicated to the presentation of two kinds of features.

2.4.1 Edge-based features

the contours of the object and the geometrical features in the pattern of the object are considered here. These edges are tracked from an image to another one using a search along the contour normal. The points evenly sampled along the contours used for this low-level tracking process are called the edge locations and are denoted $\mathbf{x}_t^i$ in $\mathbf{I}^t$ (see Figure 3).

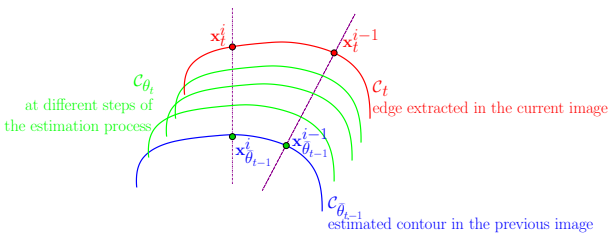


Figure 3: Edge-based tracking

As there is no matching correspondence between the edge locations in $\mathbf{I}^{t-1}$ and those in $\mathbf{I}^t$ (see Figure 3), a basic point-to-point distance minimization may lead to a mistaken motion estimation. In [10], the Iterative Closest Point algorithm is used: at each iteration of the minimization process, the point matching is updated before estimating the transformation parameters. To avoid the matching step in the proposed method, the point-to-contour distance $d_\perp\big(\mathcal{C}_{\theta_t}, \mathbf{x}_t^i\big)$ is the feature used in the minimization process (8), where $\mathcal{C}_{\theta_t} = \mathbf{\Psi}_{\theta_t}(\mathcal{C}_{\widehat{\theta}_{t-1}})$ is the contour estimated from the current parameters of the 2D transformation θ_t. Referring to (5), $m^i_{\theta_t} = d_\perp\big(\mathcal{C}_{\theta_t}, \mathbf{x}_t^i\big)$ whereas $m_t^i = d_\perp\big(\mathcal{C}_t, \mathbf{x}_t^i\big)$ which obviously is equal to zero since $\mathbf{x}_t^i$ is located on the contour $\mathcal{C}_t$ observed in $\mathbf{I}^t$ (although $\mathcal{C}_t$ is never really computed).

For the $\mathbf{J}_{d_\perp\left(\mathcal{C}_{\theta_t}, \mathbf{x}_t^i\right)}$ computation, let us express $d_\perp\big(\mathcal{C}_{\theta_t}, \mathbf{x}_t^i\big)$ as a function of the current contour parameters ϵ_j (for example lines or curves parameters) that depend on θ_t. One then has :

$$\mathbf{J}_{d_\perp\left(\mathcal{C}_{\theta_t}, \mathbf{x}_t^i\right)} = \sum_j \frac{\partial d_\perp\big(\mathcal{C}_{\theta_t}, \mathbf{x}_t^i\big)}{\partial \epsilon_j} \frac{\partial \epsilon_j}{\partial \theta_t} \tag{14}$$

From this latter equation, one can easily obtain the analytical form of the Jacobian matrix.

The computation of $d_\perp\big(\mathcal{C}_{\theta_t}, \mathbf{x}_t^i\big)$ depends on the geometrical features outlining the contours. The framework has been applied to objects outlined by lines as well as by a NURBS (Non Uniform Rational B-Spline) [23]. In the case of a polygonal object, let us note $\mathbf{l}^j_{\theta_t}$ the lines modeling its contours according to the current 2D transformation parameters. The distance $\mathbf{m}_{\theta_t}$ is then given by :

$$m^i_{\theta_t} = d_\perp\big(\mathbf{l}^j_{\theta_t}, \mathbf{x}_t^{hi}\big) = \mathbf{l}^j_{\theta_t}.\mathbf{x}_t^{hi\top} \tag{15}$$

where $\mathbf{x}_t^{hi}$ are the edge locations extracted in image $\mathbf{I}^t$ using homogeneous coordinates and $\mathbf{l}_{\theta_t}^j = (a_{\theta_t}^j, b_{\theta_t}^j, c_{\theta_t}^j)$ the normalized line parameters estimated using the current parameters of the 2D transformation..

For more complex shapes, NURBS (Non Uniform Rational B-Spline) [23] can be used rather than B-Spline. NURBS are invariant to perpective transformation thanks to a weight associated with each control point of the curve. The curve $C_{\theta_t}(s) = (x_{\theta_t}(s), y_{\theta_t}(s))^\top$ is then defined by :

$$C_{\theta_t}(s) : \begin{cases} x_{\theta_t}(s) & = & \frac{\sum_j \alpha_{\theta_t}^j w_{\theta_t}^j N_j(s)}{\sum_j w_{\theta_t}^j N_j(s)} \\ y_{\theta_t}(s) & = & \frac{\sum_j \beta_{\theta_t}^j w_{\theta_t}^j N_j(s)}{\sum_j w_{\theta_t}^j N_j(s)} \end{cases} \tag{16}$$

where $\mathbf{Q}_{\theta_t}^j = (\alpha_{\theta_t}^j, \beta_{\theta_t}^j)^\top$ are the control points of the curve and $N_j(s)$ are the B-spline basis functions. $w_{\theta_t}^j$ is the weight associated with the control point $\mathbf{Q}_{\theta_t}^j$. It could be interpreted as the third homogeneous coordinate of the control point, which enables to deal easily with homography saving time-consuming least-square estimations [23]. The distance between a point and the curve is approximated by the distance between the point and the line tangent to the NURBS. The minimization problem is then similar to the polygonal object case since a distance between a point and a line is considered. In the result section, both cases are presented.

If only such edge-based features are used in the minimization process (8), the output of the tracker is accurate when the tracked object is not textured. However, jittering happens. Furthermore, it requires a good initialization and it is sensitive to texture/cluttered environment.

2.4.2 Texture-based features

the features considered here are the classical one in template-based matching, the brightness values of the object pattern :

$$m_{\theta_t}^i = I^t(\mathbf{x}_{\theta_t}^i) \tag{17}$$

where $I^t(\mathbf{x}_{\theta_t}^i)$ is the current brightness value sub-sampled at the location $\mathbf{x}_{\theta_t}^i = \mathbf{\Psi}_{\theta_t}(\mathbf{x}_0^i)$. The locations $\mathbf{x}^i$ are called the texture points. With the constant illumination assumption, one has $\mathbf{m}_t = \mathbf{m}_0$ where $\mathbf{m}_0$ is the template sub-sampled in the first image. To improve the tracking as it is proposed in [25], $\mathbf{x}_0^i$ are Harris points.

The Jacobian matrix of $\mathbf{m}_{\theta_t}^j$ is [11] :

$$\mathbf{J}_{I_{\theta_t}^i} = \frac{\partial I^t(\mathbf{x}_{\theta_t}^i)}{\partial \theta_t} = \nabla \mathbf{I}^t(\Psi_{\theta_t}(\mathbf{x}_0^i))^\top \frac{\partial \Psi_{\theta_t}(\mathbf{x}_0^i)}{\partial \theta_t} \tag{18}$$

where $\nabla \mathbf{I}^t(\mathbf{x})$ is the spatial gradient of $\mathbf{I}^t$ at the location $\mathbf{x}$. From (1), one gets easily $\partial \Psi_{\theta_t}(\mathbf{x}_0^i)/\partial \theta_t$ (see [11] for the complete derivation and speed-up computing).

If only such texture-based features are used in the minimization process (8), the output ofthe tracker is robust to large 2D transformation and to occlusions. It is very smooth as the tracker uses information about the whole object. On the other side, the drawbacks of such an approach are that it requires a well-textured object and it is sensitive to changes of illumination.

3 Results

To initialize a tracking, the contour selection is performed by an operator. From there, the Harris points are selected automatically. They are chosen such that they cover as much as possible the whole pattern.

The four following subsections present some tracking results on video sequences. Our hybrid tracker is compared to an edge-based one and a texture-based one [11]. These two latter ones are similar to our hybrid tracker but using only the kind of feature associated with. The same amount of data is used for each tracker: if $2n$ features are tracked using a single cue tracker, then n of each kind of features are tracked using our hybrid tracker. In the first image of these tracking experiments, the edge locations and texture points used in the minimization process are displayed (red crosses for inliers and green ones for outliers). The object position in each image is given by the current contour in red.

The objects to track in the two first experiments are polygonal shaped and in the two following ones they outlined by a NURBS.

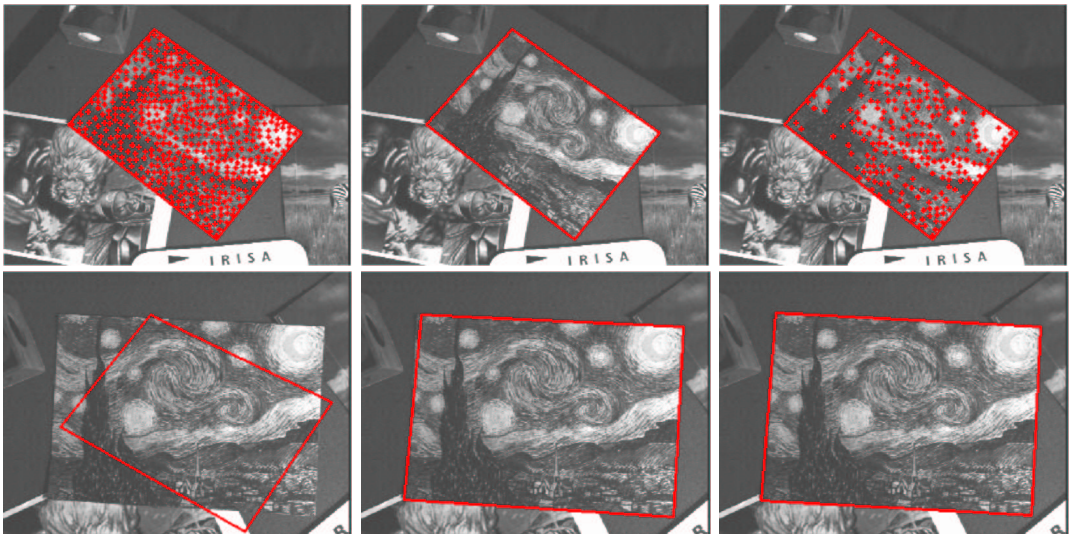
In the last subsection, a tracking performed during a visual servoing task is presented.

3.1 The Van Gogh "starry night" sequence

In this first sequence (about 45 images), large displacements are considered. Inter-frame displacement may reach about 14 pixels as shown in Figure 5(a). The initial and final images for each tracker are shown in Figure 4. The texture-based tracker loses the object and the edge-based one gives quite good results but some little imprecisions are sometimes observed (see Figure 5(b)). The only tracker that gives a good position of the object is the hybrid one. 470 points are tracked in each case. The hybrid tracker runs at an average rate of 13 Hz.

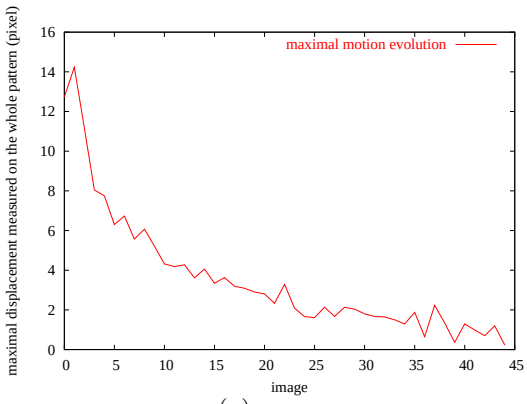
3.2 The mouse pad sequence

In this experiment (600 images), the background is highly textured. The initial images, some intermediate and the final ones are shown in Figure 6. As it could be expected with a textured background, the edge-based tracker is disturbed by the neighbouring contours and finally loses the object. The texture-based and the hybrid ones succeed to track correctly the mouse pad.

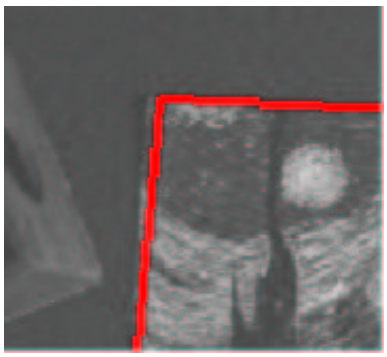


(a) Texture-based (b) Edge-based (c) Hybrid

Figure 4: Van Gogh sequence: polygonal outline tracking. Initial and final images. The texture-based tracker fails to track the object. On the contrary, the two other ones succeed.



(a)

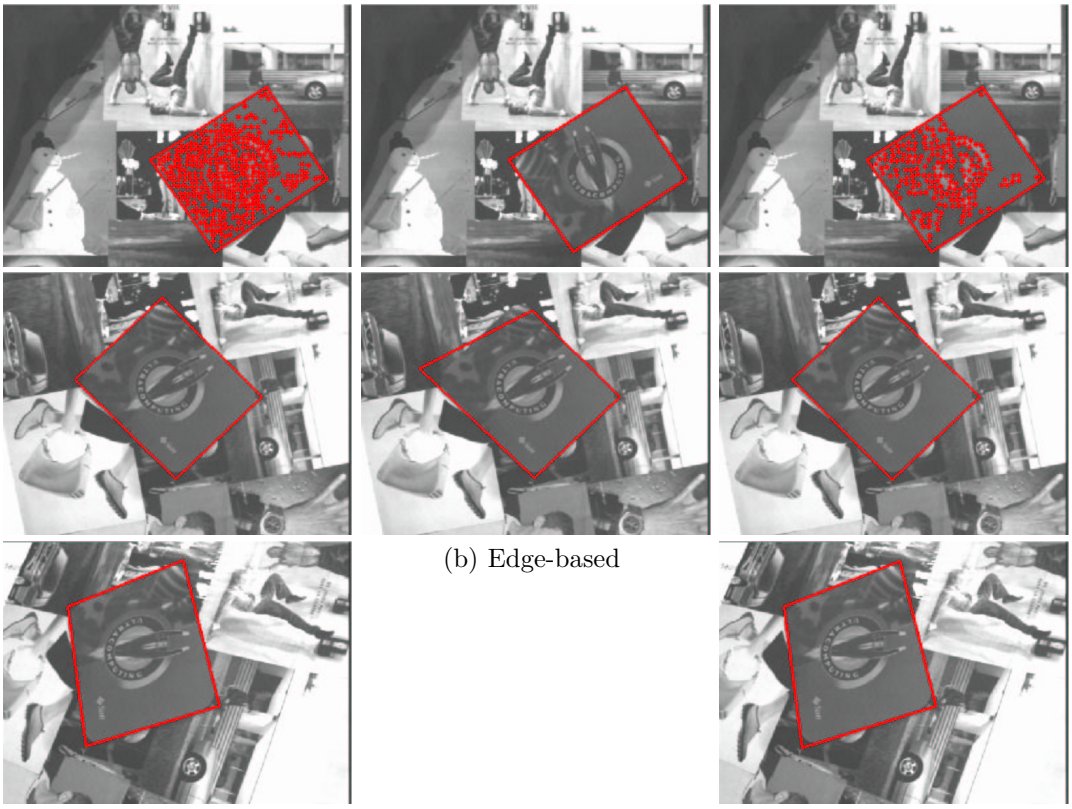


(b)

Figure 5: Van Gogh sequence: polygonal outline tracking. (a) Maximum motion between two successive frames. (b) Detail of the last image of Figure 4b: imprecisions are observed in the object position estimation when using the edge-based tracker.

The difference between the whole initial template and its reprojection in the current image is displayed for the three trackers in Figure 7. Let us note that although the texture-based tracker is based on the brightness difference minimization, this brightness difference on the whole template is smaller in the hybrid case. This means that the hybrid tracker estimates better the 2D transformation parameters thanks to the complementarity of each kind of features.

For this sequence, 340 points are tracked in each case. The hybrid tracker runs at an average rate of 16 Hz.



(a) Texture-based (b) Edge-based (c) Hybrid

Figure 6: Mouse pad sequence. Initial images, some intermediate and the final ones. The edge-based tracker fails to track the object. On the contrary, the two other ones succeed.

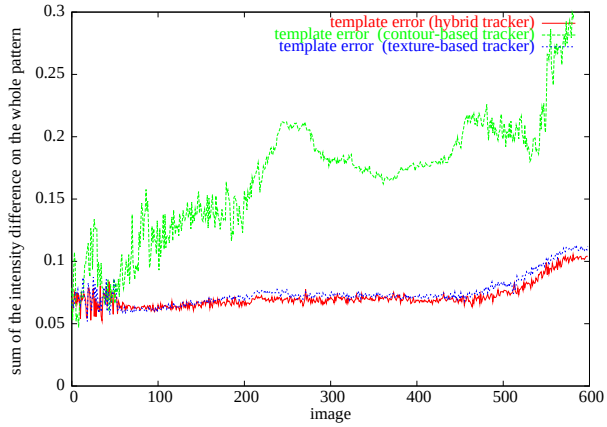


Figure 7: Mouse pad sequence. Difference between the initial template and its reprojection in the current image for the texture-based tracker (blue), the edge-based tracker (green) and the hybrid tracker (red)

3.3 The apple sequence

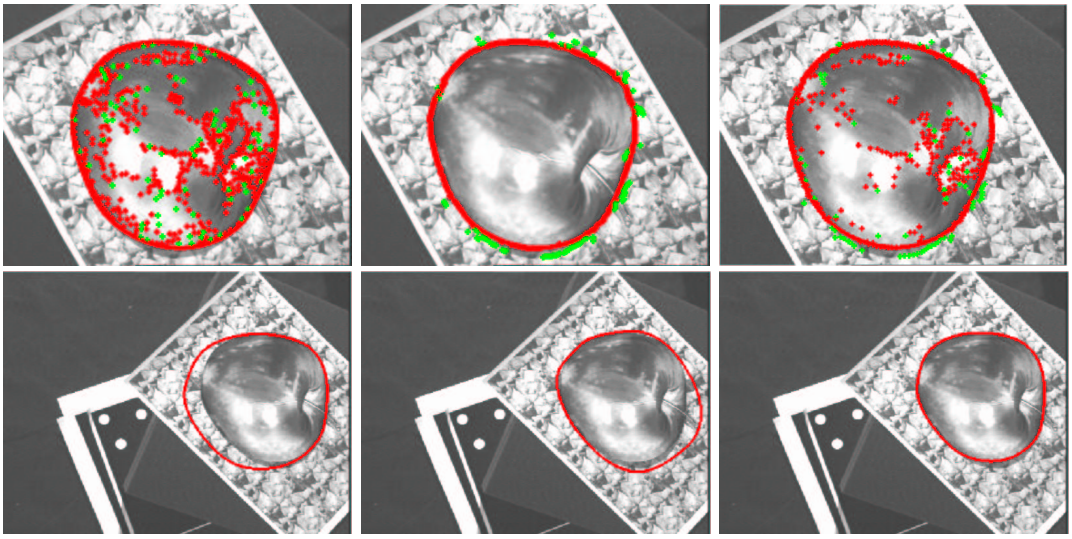
The framework described in this paper has been applied to objects outlined by a NURBS. Figure 8 is an example of such a tracker. The object to track is a picture of an apple. The challenge here is to obtain an accurate contour, which is quite difficult due to the background and the shadow. The only tracker that succeeds to track the object is the hybrid one. As previously, the selected features are shown in the first image. The red crosses are for the inliers ones and the green crosses for the features considered as outliers. For this sequence, 450 points are tracked in each case.

3.4 The vase sequence

In this sequence of 410 images, the picture of a vase is tracked. The difficulties here come from the complex environment and the large occlusions.

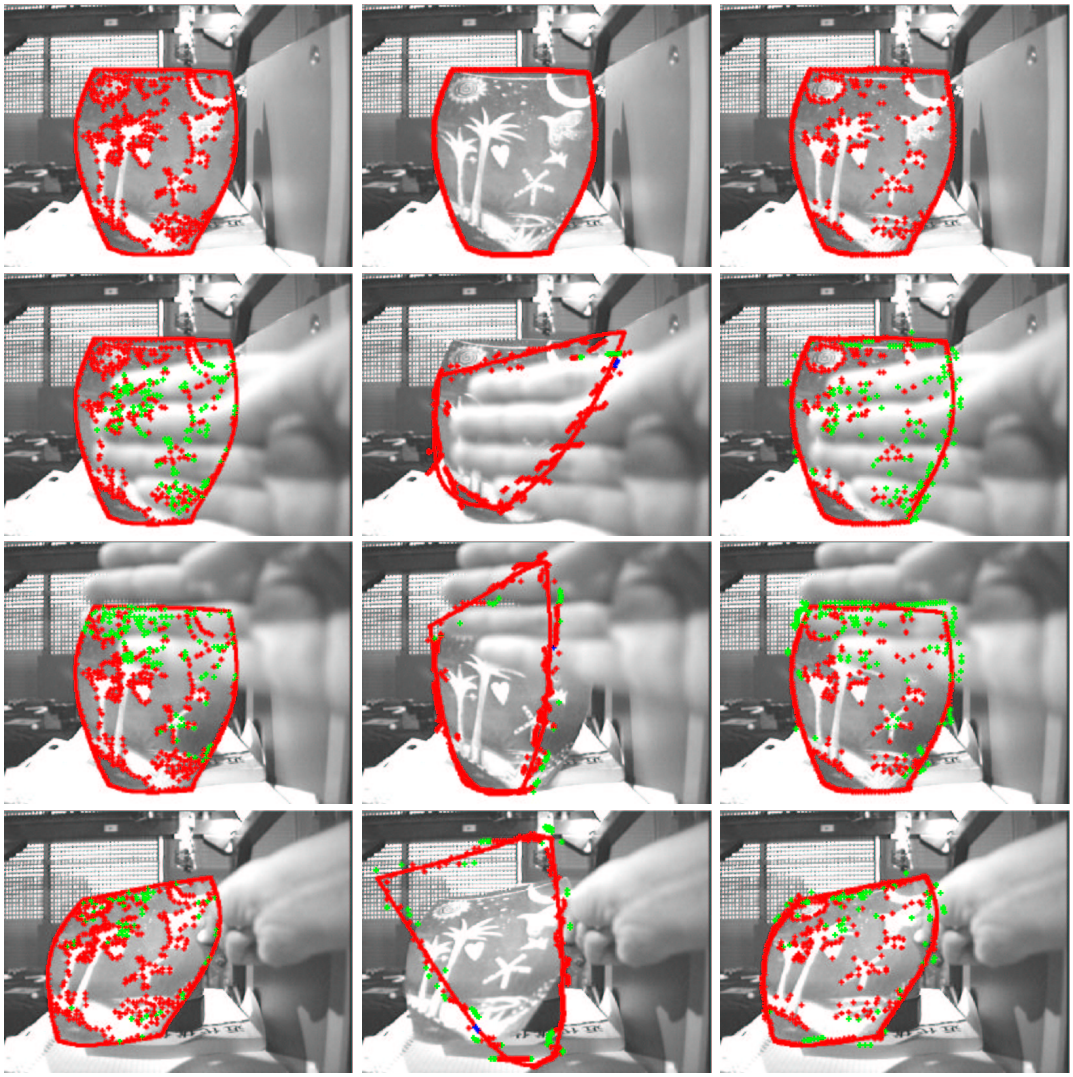
The initial, intermediate and final images of the sequence for each tracker are showned in Figure 9. As the background is highly textured, the edge-based tracker loses the object during the second occlusion. The two others ones succeeds to track the object. As observed in the second experiment, although the texture-based tracker and the hybrid one tracked correctly the object, when comparing the difference between the current template and the initial one reprojected in the current image (see Figure 10, one can see that the hybrid one best estimates the 2D transformation parameters. Let note that the two peaks are due to occlusions.

440 points were used for each tracker. The hybrid one runs at an average rate of 15 Hz.



(a) Texture-based (b) Edge-based (c) Hybrid

Figure 8: Apple sequence: NURBS tracking. Initial and final images. The hybrid tracker succeeds to track the object while the two other ones fail.



(a) Texture-based (b) Edge-based (c) Hybrid

Figure 9: Vase sequence: NURBS tracking. Initial, intermediate and final images. The edge-based tracker fails to track the object while the two other ones succeed.

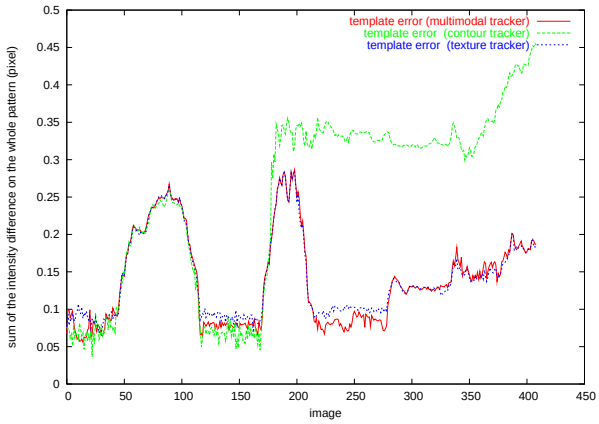


Figure 10: Vase sequence: NURBS tracking. Difference between the initial template and its reprojection in the current image for the texture-based tracker (blue), the edge-based tracker (green) and the hybrid tracker (red)

3.5 Visual servoing experiment

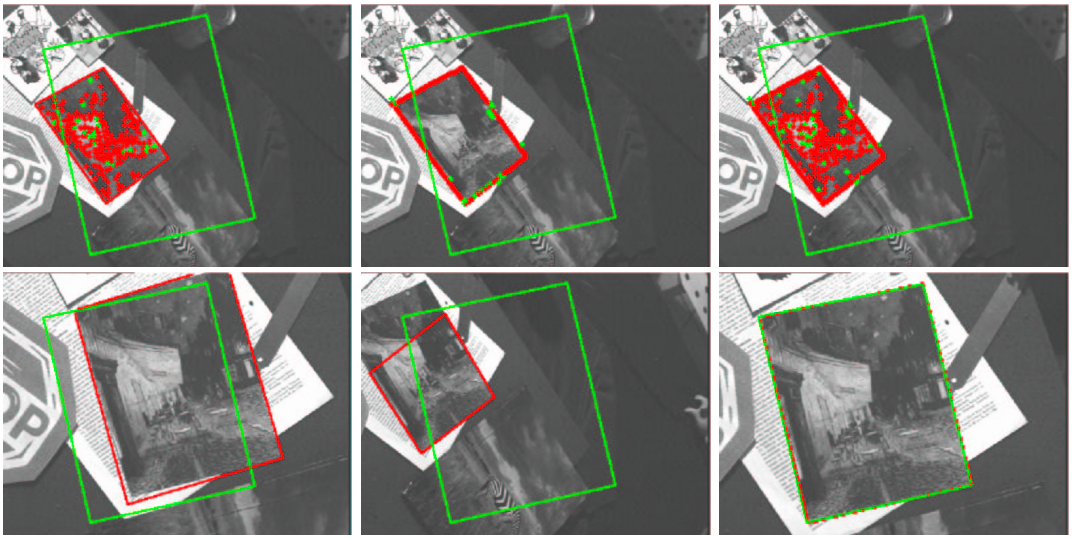
The hybrid tracker has been used successfully in image-based visual servoing . Visual servoing aims to control a system dynamically by exploiting the visual information captured by a visual sensor [9, 14]. The task is specified by a set of desired features associated to its desired position in the image. The velocity of a camera mounted on the end-effector of a 6 d.o.f robot is controlled such that the error between the desired features and the current value of the features gets minimized. The features chosen for this experiment are based on the image moments of the object as defined in [5].

The tracking process is a really important step in a visual servoing task. Using inertial moment as visual feature requires a very precise estimation of the object localization. If the tracking lacks of precision, the task may diverge or may be not accurately achieved.

The task is performed four times: once to test the output of each tracker when no occlusion occurs and once again to test the output of the hybrid tracker when multiple occlusions occur. Let us note that no information about the camera motion is used in the tracking process. The desired position of the object in the image is given by hand.

The initial and final images of the experiment performed without occlusion are shown in Figure 11. The desired position of the object in the image is drawn in green.

The visual servoing task was stopped when a part of the object is out of the image. Even if it may not be a failure, one can see the tracking was bad in the single cue cases. Although the tracker proposed in this paper is slower than the single-cue trackers, the experiments



(a) Texture-based (b) Edge-based (c) Hybrid

Figure 11: Visual servoing experiments without occlusion. Green rectangle: desired position of the object in the image. Initial and final images. Only the hybrid tracker performs a good tracking. The edge-based tracker completely diverges and the texture-based lacks of accuracy.

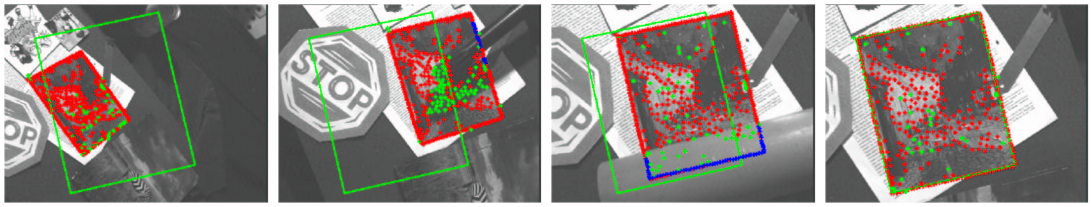
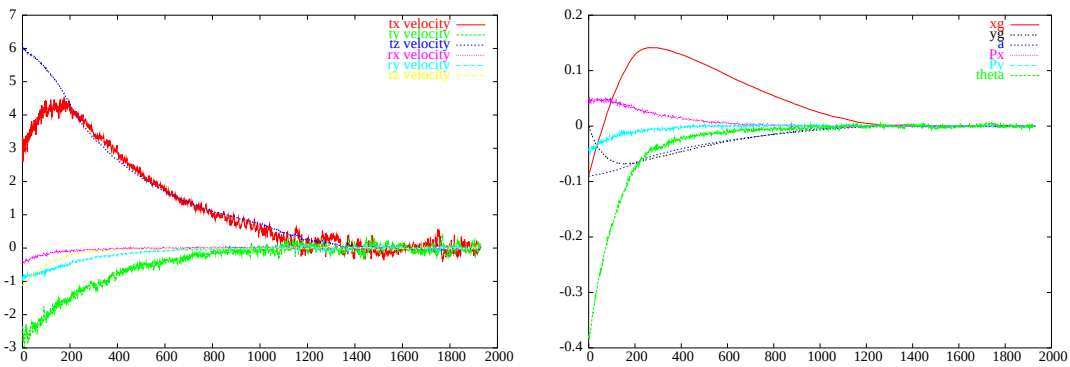


Figure 12: Visual servoing experiment with occlusions. Green rectangle : desired position of the object in the image. The green crosses are points associated with features considered as outliers (due to noise, occlusions or shadow) and the red ones are for the inliers ones.

show that it is better than the single-cue ones. Using only texture information is not accurate enough because of the object scale changes during the experiment. In such cases, the edge-based features are important to adjust more accurately the object position in the image.

The initial image, some intermediate and the final ones of the experiment performed with occlusions are shown in Figure 12. The green crosses are points associated with features considered as outliers (due to noise, occlusions or shadow) and the red ones are for the inlier ones. Hidden edge locations are represented in blue. One can see that the occluded parts are well detected.

The output of the hybrid tracker enables a good behavior of the camera and the positionning task is correctly achieved. In Figure 13(a), the evolution of the camera velocity is shown, as well as the error between the desired features and the current ones in Figure 13(b). The camera displacement is smooth and the accuracy of our tracker enables to achieve a very good positioning. In Figure 14, the desired position and the two final ones (without and with occlusions) obtained using the hybrid tracker are presented. The positionning is well achieved in both cases: the error on the camera pose is below 1 degree on rotation and 5 mm on translation when no occlusion occurs and below 1.5 degree on rotation axis and 10 mm on translation when occlusions occur.



(a) (b)

Figure 13: Visual servoing without occlusion using our hybrid tracker. (a) camera velocity, (b) error in the image of each visual feature

Axes	t_x	t_y	t_z	r_x	r_y	r_z
Desired pose	403.3	-51.3	300	17.8	0	5.2
Final pose (a)	408.7	-51.3	299.4	18.4	0	4.7
Final pose (b)	413.8	-48.9	300.2	18	-0.2	4

Figure 14: Visual servoing using our hybrid tracker: desired and final positions (a) case without occlusion, (b) case with occlusions (t_x, t_y and t_z are in mm while r_x, r_y and r_z are in degrees)

4 Conclusion

In this paper a reliable 2D tracker has been presented. It is based on a multi-cue template matching where an object is represented by the most relevant points of its brightness pattern and a regular sampling along its contours. By fusing the motion estimation of the edge locations and the texture points, the proposed approach enables an accurate tracking of textured objects in a textured background. It relies on a non-linear minimization process. Furthermore the minimization handles outlier rejection and the tracking is therefore robust to noise and partial occlusion.

Different experiment situations have been described which point out the robustness of the tracker to the nature of the tracked object, to the complexity of the environment or to the motion range. Since this tracker is precise and fast enough, it has been used successfully in visual servoing experiments where single cue trackers have failed or were not accurate enough.

The 2D transformation presented in this paper is valid only for planar objects. There does not exist any 2D transformation that accounts for a generic 3D object motion. Therefore to handle such cases, we are now interested in hybrid 3D tracking, by fusing pose computation and motion estimation. Such a tracker should be more robust and smoother than traditional 3D trackers.

References

[1] B. Bascle, P. Bouthemy, N. Deriche, and F. Meyer. Tracking complex primitives in an image sequence. In *Int. Conf. on Pattern Recognition, ICPR'94*, pages 426–431, Jerusalem, October 1994.

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

[3] A. Blake and M. Isard. *Active Contours*. Springer Verlag, April 1998.

[4] J. Buenaposada and L. Baumela. Real-time tracking and estimation of plane pose. In *IAP Int. Conf. on Pattern Recognition, ICPR'02*, volume 2, pages 697–700, Québec, Canada, August 2002.

[5] F. Chaumette. Image moments: a general and useful set of features for visual servoing. *IEEE Trans. on Robotics*, 20(4):713–723, August 2004.

[6] N. Chiba and T. Kanade. A tracker for broken and closely-spaced lines. In *ISPRS Int. Society for Photogrammetry and Remote Sensing Conf.*, pages 676 – 683., Hakodate,Japan, 1998.

[7] A.I. Comport, E. Marchand, and F. Chaumette. Robust model-based tracking for robot vision. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'04*, volume 1, pages 692–697, Sendai, Japan, September 2004.

[8] T. Drummond and R. Cipolla. Real-time visual tracking of complex structures. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 27(7):932–946, July 2002.

[9] B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. *IEEE Trans. on Robotics and Automation*, 8(3):313–326, June 1992.

[10] A.W. Fitzgibbon. Robust registration of 2d and 3d point sets. *Image and Vision Computing*, 21(12-13):1145–1153, December 2003.

[11] G. Hager and P. Belhumeur. Efficient region tracking with parametric models of geometry and illumination. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(10):1025–1039, October 1998.

[12] G. Hager and K. Toyama. The XVision system: A general-purpose substrate for portable real-time vision applications. *Computer Vision and Image Understanding*, 69(1):23–37, January 1998.

[13] P.-J. Huber. *Robust Statistics*. Wiler, New York, 1981.

[14] 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.

[15] M. Isard and A. Blake. Icondensation: Unifying low-level and high-level tracking in a stochastic framework. In *European Conf. on Computer Vision (1), ECCV'98*, pages 893–908, Freiburg, Germany, 1998.

[16] F. Jurie and M. Dhome. Hyperplane approximation for template matching. *IEEE trans on Pattern Analysis and Machine Intelligence*, 24(7):996–1000, July 2002.

[17] D. Kragic and H. Christensen. Cue integration for visual servoing. *IEEE Trans. on Robotics and Automation*, 17(1):19–26, February 2001.

[18] E. Marchand, P. Bouthemy, F. Chaumette, and V. Moreau. Robust real-time visual tracking using a 2D-3D model-based approach. In *IEEE Int. Conf. on Computer Vision, ICCV'99*, volume 1, pages 262–268, Kerkira, Greece, September 1999.

[19] L. Masson, F. Jurie, and M. Dhome. Contour/texture approach for visual tracking. In *13th Scandinavian Conf. on Image Analysis, SCIA 2003*, volume 2749 of *Lecture Notes in Computer Science*, pages 661–668. Springer, 2003.

[20] I. Matthews, T. Ishikawa, and S. Baker. The template update problem. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26(6):810–815, June 2004.

[21] H.T. Nguyen, M. Worring, and R. van den Boomgaard. Occlusion robust adaptive template tracking. In *IEEE Int. Conf. on Computer Vision*, volume 1, pages 678–683, Vancouver, B.C.,Canada, 2001.

[22] J.-M. Odobez and P. Bouthemy. Robust multiresolution estimation of parametric motion models. *Journal of Visual Communication and Image Representation*, 6(4):348–365, December 1995.

[23] L. Piegl and W. Tiller. *The NURBS book (2nd ed.)*. Springer-Verlag New York, Inc., 1997.

[24] A. Shahrokni, To Drummond, and P. Fua. Texture boundary detection for real-time tracking. In *European Conf. on Computer Vision, ECCV'04, LNCS 3022*, volume 2, pages 566–577, Prague, Czech Republic, May 2004.

[25] J. Shi and C. Tomasi. Good features to track. In *IEEE Int. Conf. on Computer Vision and Pattern Recognition, CVPR'94*, pages 593–600, Seattle, Washington, June 1994.

[26] L. Vacchetti, V. Lepetit, and P. Fua. Stable 3–d tracking in real-time using integrated context information. In *IEEE Int. Conf. on Conf. on Computer Vision and Pattern Recognition, CVPR'03*, volume 2, pages 241–248, Madison, WI, June 2003.

Contents