IEEE International Conference on Computer Vision, ICCV'99,
Volume 1, Pages 262–268, Kerkira, Greece, September 1999

Robust real-time visual tracking using a 2D-3D model-based approach

Éric Marchand[1], Patrick Bouthemy[1], François Chaumette[1], Valérie Moreau[2]

[1]IRISA / INRIA Rennes
Campus universitaire de Beaulieu
35042 Rennes Cedex, France

[2]EDF - Pôle Industrie -
Division Recherche et Développement
6, Quai Watier
78401 Chatou Cedex, France

Abstract

We present an original method for tracking, in an image sequence, complex objects which can be approximately modeled by a polyhedral shape. The approach relies on the estimation of the 2D object image motion along with the computation of the 3D object pose. The proposed method fulfills real-time constraints along with reliability and robustness requirements. Real tracking experiments and results concerning a visual servoing positioning task are presented.

1 Introduction

Most of the available tracking techniques can be divided into two main classes: feature-based and model-based. The former approach tracks features such as geometrical primitives (points, segments, circles, ...), object contours [1, 10], regions of interest [8], ... The latter explicitly uses a model of the tracked objects. This model can be a CAD model [4, 6, 13, 16, 14] or a 2D template of the object [12]. This second class of methods usually provides with a more robust solution (for example, it can cope with partial occlusion of the objects). Both approaches may use Kalman filters to predict and estimate the position of the tracked primitives over time.

In our case, we aim at designing a tracking algorithm fulfilling the following properties or constraints: it should be fast and robust, it should not require any temporal evolution model, it should not involve any complex feature extraction (such as contour extraction). Therefore, we have developed a model-based 2D-3D approach that relies on the estimation of the 2D object motion and of the 3D pose of the object. It supplies a fast and robust tracking of complex objects which can be approximately modeled by a polyhedral shape. More precisely, in a first step, the object image motion is represented by a 2D affine motion model, and is estimated, using a robust statistical method, from the computation of the normal displacements evaluated along the projected object model contours. These normal displacements are determined with the algorithm described in [3]. The 2D affine motion model does not always match the real displacement of the object. A second step that consists in fitting the projection of the object model on the intensity gradients in the image is necessary. This is achieved using an iterative minimization of a non-linear energy function with respect to 3D pose parameters. The main advantages of this two-step method can be summarized as follows. The 2D motion estimation stage allows us to handle large displacements of the object, and to avoid a prediction step (that is often a questionable issue). The result of this stage is exploited to supply an appropriate initialization to the pose estimation. Our model-based tracking only requires a coarse calibration of the camera and a rough model of the object. Both 2D motion estimation and 3D pose estimation do not involve edge detection (we only consider gray level images). Both are robust to partial occlusions of the object. Finally, we can perform real-time tracking (currently, up to 10Hz on a PC 400MHz).

One of our goals is to use this tracker within visual servoing tasks. Visual servoing [9], that consists in controlling robot motion with respect to image information, is now used in industrial environment. However, if most of the control issues are now well known and robust control laws can be defined to perform positioning tasks, efficient image processing tools seem to be one of the main shortcomings to a wide use of these techniques. Indeed, to fit visual servoing requirements, image feature extraction must be robust, accurate, and computed in real-time (at least at the highest possible rate). Current techniques exploited in industrial environment use marked objects. However, in order to increase the versatility of visual servoing techniques, such a requirement must be alleviated. We will see that the developed tracking algorithm is perfectly suitable for positioning tasks. These systems are of interest for the Research and Development division of EdF (Électricité de France) to

achieve maintenance and monitoring tasks in hostile environment (nuclear power plant). The paper is organized as follows. Section 2 describes the 2D motion-based tracking stage that acts as an initialization to the 3D model-based tracking presented in Section 3. Experimental tracking results and real-time visual servoing tasks are reported in Section 4.

2 Motion-based tracking

We first consider that the global transformation between two successive projections of the object in the image plane can be represented by a 2D affine motion model. The goal of this first step is to estimate the parameters of this 2D transformation even in presence of large 2D displacements of the object image. Contrary to usual Kalman filter methods, this motion-based method does not require the introduction of a state model evolution (*e.g.*, a constant velocity model), and consequently the initialization of the variance of the noise of the state and measurement models.

2.1 Affine transformation model.

Let $\mathcal{X}^t = [X_1^t, \ldots, X_n^t]^T$ be a vector formed by the image coordinates X_i^t of points along the boundaries of the object model projection at time t. The object image shape $\mathcal{X}^{t+1}$ at time $t+1$ will be given by:

$$\mathcal{X}^{t+1} = \Psi_\Theta(\mathcal{X}^t) \tag{1}$$

where Ψ_Θ is a 2D affine transformation expressed as:

$$\begin{bmatrix} x_i^{t+1} \\ y_i^{t+1} \end{bmatrix} = \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix} \begin{bmatrix} x_i^t \\ y_i^t \end{bmatrix} + \begin{bmatrix} T_x \\ T_y \end{bmatrix} = \mathbf{W}(X_i^t)\Theta \tag{2}$$

with $\Theta = (a_1, a_2, a_3, a_4, T_x, T_y)^T$, $X_i^t = (x_i^t, y_i^t)^T$, $X_i^{t+1} = \Psi_\Theta(X_i^t)$, and

$$\mathbf{W}(X) = \begin{bmatrix} x & y & 0 & 0 & 1 & 0 \\ 0 & 0 & x & y & 0 & 1 \end{bmatrix}$$

This transformation is linear wrt. Θ, and displacement $d_i(X_i) = X_i^{t+1} - X_i^t$ can be written as:

$$d_i(X_i) = \mathbf{W}(X_i)\Theta' \tag{3}$$

where $\Theta' = \Theta - (1,0,0,1,0,0)^T$.

The part of the tracking algorithm concerned with the estimation of the 2D affine parameters is articulated into two sub-steps. The first one computes normal displacements evaluated along the projection of the object model contours using the so-called Moving Edges algorithm (ME) [3], while the second one exploits this normal displacement field to estimate $\widehat{\Theta}'$ using an extension of the robust multiresolution estimation technique introduced in [15]. We now describe these two sub-steps.

2.2 Computing normal displacements.

One of the advantages of the ME method is that it does not require any prior edge extraction. We only manipulates point coordinates and image intensities. Nevertheless, for convenience, we will still use the word "contour" to refer to the list of tracked points. The ME algorithm can be implemented with convolution efficiency, and can lead to real-time computation [3, 2]. We consider a list L^t of pixels along the contour of the projection of the object model (model fitting in the first image is performed in a semi-automatic mode). The process consists in searching for the "correspondent" P_i^{t+1} in image I^{t+1} of each point $P_i^t \in L^t$. We determine a 1D search interval $\{Q_i^j, j \in [-J, J]\}$ in the direction δ of the normal to the contour (see Figure 1). For each point P_i^t of the list L^t, and for each entire position Q_i^j lying in the direction δ^* (for computational issue, δ^* is the closest direction to δ in the set $\{0^o, 45^o, 90^o, 135^o\}$), we compute a criterion corresponding to the square root of a log-likelihood ratio ζ^j. This ratio is nothing but the absolute sum of the convolution values, computed at P_i^t and Q_i^j, using a pre-determined mask M_δ function of the orientation of the contour.

New position P_i^{t+1} is given by:

$$Q_i^{j^*} = \arg \max_{j \in [-J,J]} \zeta^j \text{ with } \zeta^j = \mid I^t_{\nu(P_i)} * M_\delta + I^{t+1}_{\nu(Q_i^j)} * M_\delta \mid$$

providing that ζ^{j^*} is greater than a threshold λ. $\nu(.)$ is the neighborhood of the considered pixel. Then, pixel P_i^{t+1} given by $Q_i^{j^*}$ is stored in L^{t+1}.

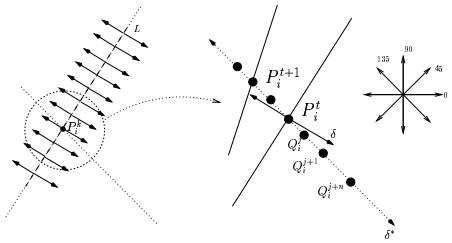

Figure 1. *Determining point positions in the next image using the ME algorithm*

At this step, we have a list of k pixels as well as their displacement component orthogonal to the object model contour: $(P_i^t, d_i^\perp)_{i=1\ldots k}$. This is performed for each new frame, it never requires the extraction of new contours. Since it is a local approach, it is robust to partial occlusions of the object and to missing measurements.

2.3 Affine transformation estimation.

Using $(P_i^t, d_i^\perp)_{i=1\ldots k}$, we can estimate the 2D affine transformation Θ'. Using equation (3), we have:

$$d_i^\perp = n_i^T \mathbf{d}(P_i) = n_i^T \mathbf{W}(P_i)\Theta' \quad (4)$$

where n_i is a unit vector orthogonal to the object model contour at point P_i. Relying on (4), we can use a robust estimator (a M-estimator ρ) to obtain $\widehat{\Theta}'$ as follows [15]:

$$\widehat{\Theta}' = \arg\min_{\Theta'} \sum_{i=1}^{n} \rho\left(d_i^\perp - n_i^T \mathbf{W}(P_i)\Theta'\right) \quad (5)$$

This robust statistical approach enables not to be affected by locally incorrect measures (due to shadows, local missmatching, occlusions, etc.)

3 Model-based tracking

Knowing the positions $\mathcal{X}^t$ of the projection of the contours of the tracked object at time t and the estimation $\widehat{\Theta}'$ of the 2D global affine motion parameters between t and $t+1$, we are able to compute the positions of points $\mathcal{X}^{t+1}$ at time $t+1$:

$$\mathcal{X}^{t+1} = \Psi_{\widehat{\Theta}}(\mathcal{X}^t)$$

However, the 2D affine transformation cannot completely account for the real transformation undergone by the projection of the object (due to perspective effects, important rotations, non shallow environment), and after a few iterations tracking may fail. To alleviate this problem the 2D affine displacement model was first augmented with 2D local deformations [7]. However, when adding local deformations, we cannot ensure 3D rigidity constraints. Moreover, this was highly time consuming. Therefore, we prefer to exploit a rough CAD polyhedral model of the object. We have to find the 3D rotation and the 3D translation (*i.e.*, pose Φ) that map the object coordinate system with the camera coordinate system. Once the pose parameters are available, we can easily determine visible and invisible faces of the object, which is of particular interest for tracking.

A number of methods to compute perspective from N points have been proposed. We need to estimate the pose of the object wrt. the camera from the positions $\mathcal{X}^{t+1}$ obtained after the first step of the algorithm described in Section 2. We use the method designed by Dementhon and Davis [5] followed by Lowe's method [14]. We therefore get a first estimate of the pose parameters Φ_{init}^{t+1} which has to be still updated to correspond as well as possible to the real new aspect of the object. This further step consists in fitting the projection of the object model on the intensity gradients in the image. This is achieved using an iterative minimization wrt. Φ of a non-linear energy function using Φ_{init}^{t+1} as initialization. Pose parameters $\widehat{\Phi}$ are given by:

$$\widehat{\Phi} = \arg\min_{\Phi} E(\Phi) \quad (6)$$

where the energy function $E(\Phi)$ is defined as:

$$E(\Phi) = -\int_{\Gamma_\Phi} \|\nabla I_{\pi_\Phi(s)}(t+1)\| ds \quad (7)$$

where Γ_Φ represents the visible part of the 3D object model contours for the pose Φ, and $\nabla I_{\pi_\Phi(s)}$ denoting the spatial gradient of the intensity function at image point $\pi_\Phi(s)$ along $\pi_\Phi(\Gamma_\Phi)$ where π_Φ is the perspective projection function.

The energy term defined in equation (7) is the simplest solution to this fitting problem. However, we may exploit other available information than the norm of the image gradient. Indeed, when projecting the object model for a given pose Φ, we are able to compute the expected direction of the projected contour at a 2D site $\varsigma = \pi_\Phi(s)$. If we denote $\mathbf{n}$ a unit vector corresponding to this expected direction, the dot product $\nabla I_\varsigma.\mathbf{n}$ should be equal to zero. Expression of the energy function can then be defined as:

$$E(\Phi) = \int_{\Gamma_\Phi} \frac{|\nabla I_\varsigma.\mathbf{n}|}{\|\nabla I_\varsigma\|^2} ds \quad (8)$$

where we only consider the sites ς where $\|\nabla I_\varsigma\| > \varepsilon$. This energy expression gives similar and even better results within textured environment.

The projection function π_Φ depends on the camera intrinsic parameters $\mathcal{I}$. The minimization of the energy function (6) requires that the camera calibration is available. Nevertheless, a rough knowledge of the camera parameters is sufficient. If calibration is wrong, the resulting estimation of Φ will be obviously biased, but the projection of the CAD model in the image, that is of interest here, is still correct. However, these parameters could also be estimated (or at least updated) on-line. In that case, the function to be minimized can be rewritten as follows:

$$\left(\widehat{\Phi}, \widehat{\mathcal{I}}\right) = \arg\min_{(\Phi,\mathcal{I})} \{E(\Phi, \mathcal{I})\} \quad (9)$$

In the general case, we have 11 parameters to estimate (if we consider the radial distortion). In practice, we have only performed experiments dealing with the on-line estimation of the radial distortion.

Discretization issue. Discretization of Γ_Φ can be considered in different ways. If we consider N_e visible contours to be discretized into N_p 2D sites $\varsigma = N_e N_p^e$, equation (7) can be rewritten as:

$$E(\Phi) = -\frac{1}{N_e} \sum_{e=1}^{N_e} \left(\frac{1}{N_p^e} \sum_{p=1}^{N_p^e} \|\nabla I_{\varsigma_p^e}\| \right) \quad (10)$$

We have now to determine the $N_e \times N_p^e$ 2D sites ς. The first approach is to discretize each contour into N_p^e sites s in the 3D space, and then, to project these sites in the image plane ($\varsigma = \pi_\Phi(s)$). ς_p^e is thus computed as:

$$\varsigma_p^e = \pi_\Phi \left(s_0^e + \frac{p}{N_p^e} (s_1^e - s_0^e) \right), \; p = 0 \dots N_p^e \qquad (11)$$

where s_0^e and s_1^e are the two 3D extremities of the contour e. A second approach can be considered. The discretization can be performed after the projection of the extremities of the visible object contour in the image. Indeed, as the considered objects are polyhedral, the projection of their contours are also segments and ς_p^e can be computed as:

$$\begin{aligned} \varsigma_p^e &= \pi_\Phi(s_0^e) + \frac{p}{N_p^e} (\pi_\Phi(s_1^e) - \pi_\Phi(s_0^e)), \; p = 0 \dots N_p^e \\ &= \varsigma_0^e + \frac{p}{N_p^e} (\varsigma_1^e - \varsigma_0^e) \end{aligned}$$

This discretization does not include the distortion term, but knowing K_d, the correct position of the points can be easily computed. These two approaches are similar in term of complexity when distortion is significant, but the second one avoids a discretization step in 3D. However, when we do not consider radial distortion, the latter approach is indeed more efficient, since we perform only two projections per segment while the former implies N_p^e projections. Furthermore, there exists efficient algorithms to compute all the pixels attached to a given segment (*e.g.*, using Bresenham algorithm).

Optimization algorithm. An important issue in this problem is the optimization procedure. Indeed, equations (7) and (8) are non linear, and involve numerous local minima. To solve this issue we resort to an explicit discrete search algorithm. Generalized Hough transform consisting in building a cumulative histogram in the pose space presents two main problems which are the size of the pose space ($\mathbb{R}^6$) and the presence of false peaks. Therefore, we have considered a recursive search algorithm. The method is inspired from an algorithm proposed for a fast block matching algorithm [11]. First, $\mathcal{F}(\phi)$ is minimized using large variation steps of the parameters. When the current minimum is found, the process is iterated with smaller variation steps around this value. In practice, the initial solution Φ_{init}^{t+1} is a proper initialization of this search algorithm. Therefore, we can bound the search space. It allows the algorithm to converge very quickly toward an appropriate minimum.

4 Experimental results

Experiments reported here after mainly involve various objects. These objects have been chosen since they can be considered as quite representative for the applications EDF is interested in. Indeed, disassembly and monitoring tasks are very important in the nuclear power plant context.

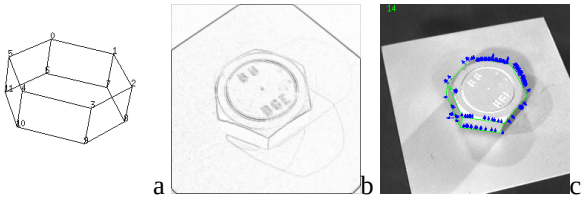

Figure 2. *Object of interest: the nut, (a) approximate CAD model of the nut, (b) intensity gradients in a typical image of the sequence, (c) normal displacement vectors used to compute the 2D affine motion model*

4.1 Nut tracking

Let us point out that the tracking of the nut silhouette in the image must deal with low intensity contrast (as it can be seen in the intensity gradient image of Figure 2.b), presence of cast shadows, mirror specularities, ... Moreover, the nut is not exactly polyhedral, since it presents no physically precisely defined ridges. Camera calibration is not precisely known. Despite these difficulties, the proposed method have proven its efficiency to track this object along long image sequences. All the images have been acquired on our robotics testbed (they have been processed off-line for results presented in Section 4.1 and 4.2, and on-line for visual servoing results presented in Section 4.4).

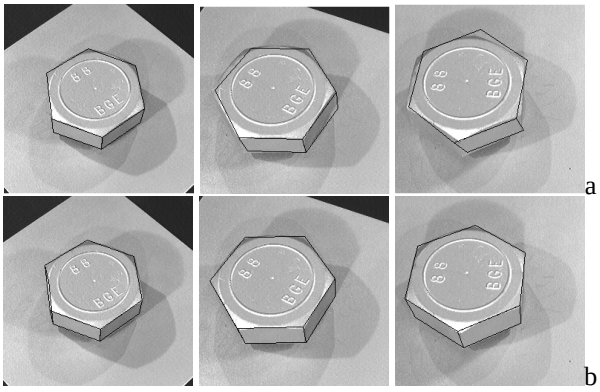

Figure 3. *Nut tracking: (a) tracking with only 2D motion estimation, (b) tracking with both 2D motion estimation and 3D pose computation*

Figure 3 contains results of the tracking of the nut along a sequence of 44 images. Figure 3.a shows the results of the tracking if we consider only the 2D motion estimation step.

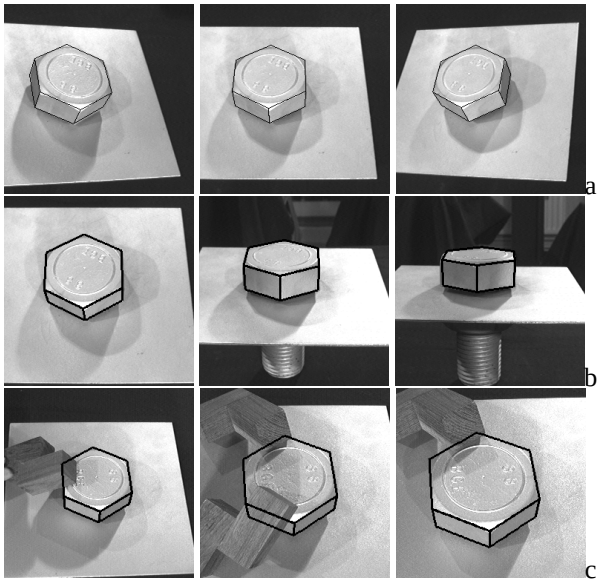

Figure 4. *Successful nut tracking experiments featuring various difficulties (see text for details)*

In that case, tracking is performed at video rate (25Hz). However, after a few images, the algorithm is no longer able to track accurately the object shape. The failure is mainly due to the fact that a 2D affine motion model cannot completely account for the projection of the 3D motion of the object. Figure 3.b reports the results of the tracking using both 2D motion estimation and 3D pose computation. In that case, tracking is performed at 10 Hz on a PC 400 Mhz under Linux OS.

We have also validated the performance of the tracking algorithm in the presence of various difficulties. In the experiment of Figure 4.a, a camera motion is performed around the y axis[1]. A face of the nut appears while another disappears. In Figure 4.b, the main difficulty is the very important rotation around the x axis. Furthermore, the illumination conditions are not constant along the sequence. In Figure 4.c, the difficulties lie in the occlusions of the nut. In Figure 7.d, the nut is tracked within a highly textured environment during a visual servoing experiment (see subsection 4.4).

4.2 Tracking a serial connector

We have evaluated our tracking algorithm on a still more complex object. In the experiments reported in Figure 5, we consider a serial port connector placed on a newspaper forming a "cluttered" background. We only built a rough approximate model. Here, we have also to deal with low intensity gradient images, specularities, and no precisely defined contours. The serial connector is successfully tracked over a 170 frames image sequence. The camera performs a large displacement around the object. A face of the object appears while another disappears. Tracking is performed at 3 Hz (this lower processing rate is mainly due to a the model of the object comprising more contours and leading to a higher number of sites ς in the evaluation of the energy function).

[1] z axis follows the optical axis, while x axis is parallel to the image rows and y axis is parallel to image columns.

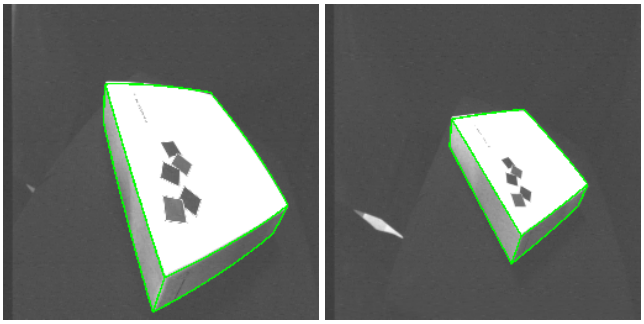

Figure 6. *Box tracking: Distortion is very important due to the use of a 3.5mm lens*

4.3 On-line estimation of radial distortion

We have also tried to estimate on-line the radial lens distortion. We have considered a simple object (a box) and a camera with an important distortion (in that case the focal lens of the camera was 3.5mm) with initial value set to 0 (see Figure 6). We estimate on-line the distortion. It decreases toward the true value when the object projection moves toward the image border (since a better estimation of this parameter is then required). In that case, the tracking is performed at 1 Hz.

4.4 Visual servoing experiments

Image-based visual servoing consists in specifying a task as the regulation in the image of a set of visual features $\mathbf{P}$ that have to match a desired value $\mathbf{P_d}$ [9].

We have conducted experiments dealing with a positioning task wrt. the nut. In that case, visual features are the x and y coordinates of six points of the upper face of the nut. The complete implementation of the visual servoing task, including tracking and control, has been carried out on an experimental testbed involving a CCD camera mounted on the end effector of a six d.o.f cartesian robot. Tracking is performed at 3Hz on a Sun Ultra-Sparc 1 (170Mhz).

Figure 7a contains some of the images delivered by the camera during the positioning task. The current polygo-

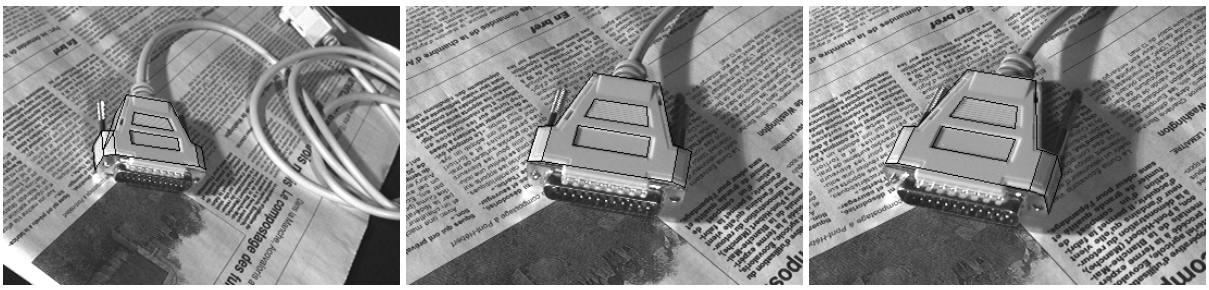

Figure 5. *Tracking a connector on a newspaper background*

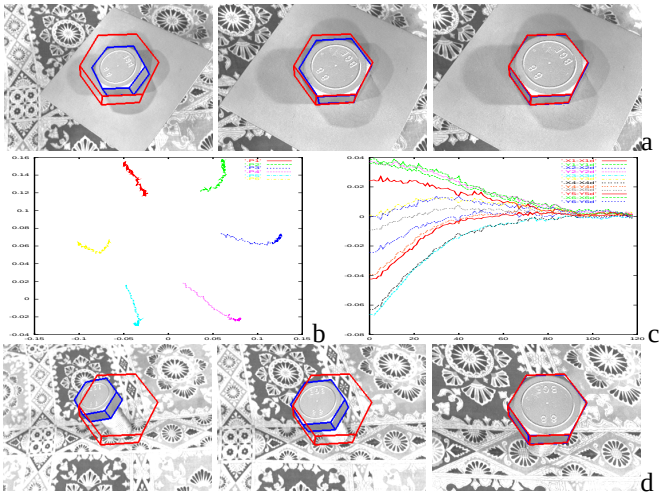

Figure 7. *Positioning on the nut by visual servoing: (a) images 15, 50 and 95, (b) temporal variation of the position of the control points in the image, (c) error between current and desired positions of the control points considered in the specification of the task, (d) another tracking experiment with a highly textured environment*

nal object model contours, depicting the tracking of the nut projection in the image, and the desired final one are drawn over the image. Figure 7b shows the apparent trajectory in the image plane of points $\mathbf{P}$ during the achievement of the task. Figure 7.c presents the temporal evolution of the error $(\mathbf{P} - \mathbf{P_d})$. This demonstrates the stability and the convergence of the control law. The error on each coordinate of the six points specifying the task converges to zero. The noise in the plots are mainly due to the fact that image processing is performed only at 3 Hz.

Robustness. To prove the robustness of our algorithm we put the nut on a highly textured environment (see Figure 7.d). As in the previous experiments, our tracking algorithm coupled with the visual servoing scheme correctly achieved the positioning task wrt. the object. Other experiments have been carried out using a micro-manipulation device as object of interest (see Figure 8). Multiple occlusions by various tools have been imposed during the positioning task.

Accuracy. As the application of this positioning task is grasping, repeatability is very important. The final position of the object in the image must be accurate enough and, in order to achieve the grasping task, the final 3D position of the robot end-effector has also to be consistent. The accuracy (standard deviation) of positioning task wrt. the nut was computed from 40 experiments using the robot odometry (which is very precise). We obtain an accuracy of less that ± 0.7 *mm* in translation and ± 0.17 deg in rotation, while the object is located at 40 cm from the camera. The mean error $\mid \mathbf{P} - \mathbf{P_d} \mid$ is less than 0.2 pixels with a standard deviation of 0.3 pixels.

4.5 Initialization of the tracking in the first image

In the current version of the system described in this paper, initialization of the tracking in the very first image of the sequence is performed partly manually. This means that the user has to click at least four points on both the initial image and the CAD model of the object. A completely automatic object localization procedure could be implemented, but this is outside the tracking issue considered in this paper. Let us finally note that, if the user clicks a minimum of six points, a full camera calibration can be performed. The obtained intrinsic parameters can be used afterwards in the pose computation algorithm.

5 Conclusion

We have presented an original method for tracking complex objects in an image sequence at a high processing rate (but not yet exactly video rate). The tracking is based on the estimation, between two successive images, of a 2D global affine transformation and the computation of the object pose

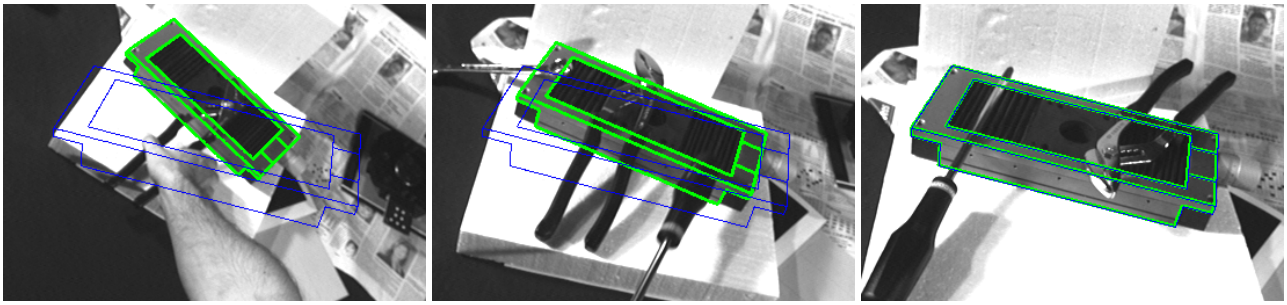

Figure 8. *Positioning wrt. a micro-manipulation device (initial, middle and final position), desired position appears in dark lines.*

formulated as an energy minimization process. To perform this last step, an approximate polyhedral model of the object is sufficient. Appearance and disappearance of hidden faces of the object can be straightforwardly handled. Both steps of the tracking algorithm are robust to partial occlusions. The direct extension to non polyhedral object can be considered provided a 3D description of the object is available. This tracking algorithm allows us to carry out a visual servoing task of positioning with respect to real objects (without any landmarks) in complex situations. Visual servoing is not the only application of this original and efficient tracking method. Indeed, if we are able to achieve a 2D tracking, we can also recover a precise estimation of the position of the camera wrt. the object if the camera is well calibrated, and then we can also perform a real 3D tracking.

Acknowledgment. This study has been supported by EDF - Pôle Industrie- Division Recherche et Développement. Images on Figure 6 have been provided by the LASMEA, University of Clermont-Ferrand, France.

References

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

[2] S. Boukir, P. Bouthemy, F. Chaumette, and D. Juvin. A local method for contour matching and its parallel implementation. *Machine Vision and Application*, 10(5/6):321–330, April 1998.

[3] P. Bouthemy. A maximum likelihood framework for determining moving edges. *IEEE Trans. on Pattern Analysis and Machine intelligence*, 11(5):499–511, May 1989.

[4] N. Daucher, M. Dhome, J. Lapreste, and G. Rives. Modelled object pose estimation and tracking by monocular vision. In *British Machine Vision Conference, BMVC'93*, pages 249–258, Guildford, UK, September 1993.

[5] D. Dementhon and L. Davis. Model-based object pose in 25 lines of codes. *Int. J. of Computer Vision*, 15:123–141, 1995.

[6] D. Gennery. Visual tracking of known three-dimensional objects. *Int. J. of Computer Vision*, 7(3):243–270, 1992.

[7] N. Giordana, P. Bouthemy, F. Chaumette, F. Spindler, J.-C. Bordas, and V. Just. 2d model-based tracking of complex shapes for visual servoing tasks. In G. Hager and M. Vincze, editors, *IEEE Workshop on Robust Vision for Vision-Based control of Motion*, Leuven, Belgium, May 1998.

[8] 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.

[9] 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.

[10] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. In *European Conference on Computer Vision, ECCV'96, LNCS no. 1064, Springer-Verlag*, pages 343–356, Cambridge, UK, 1996.

[11] J. Jain and A. Jain. Displacement measurement and its application in interframe image coding. *IEEE Trans. on Communications*, 29(12):1799–1808, December 1981.

[12] C. Kervrann and F. Heitz. A hierarchical Markov modeling approach for the segmentation and tracking of deformable shapes. *Graphical Models and Image Processing*, 60(3):173–195, may 1998.

[13] H. Kollnig and H.-H. Nagel. 3d pose estimation by fitting image gradients directly to polyhedral models. In *IEEE Int. Conf. on Computer Vision*, pages 569–574, Boston, MA, May 1995.

[14] D. Lowe. Robust model-based motion tracking through the integration of search and estimation. *Int. J. of Computer Vision*, 8(2):113–122, 1992.

[15] 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.

[16] M. Tonko, K. Schäfer, F. Heimes, and H.-H. Nagel. Towards visually servoed manipulation of car engine parts. In *Proc. IEEE Int. Conf. on Robotics and Automation, ICRA'97*, volume 4, pages 3166–3171, Albuquerque/NM, Avril 1997.