

# Active contours for video object tracking using region, boundary and shape information

Mohand Saïd Allili · Djemel Ziou

Received: 17 October 2006 / Revised: 24 March 2007 / Accepted: 30 March 2007 / Published online: 9 May 2007  
© Springer-Verlag London Limited 2007

**Abstract** In this paper, we propose a robust model for tracking in video sequences with non-static backgrounds. The object boundaries are tracked on each frame of the sequence by minimizing an energy functional that combines region, boundary and shape information. The region information is formulated by minimizing the symmetric Kullback–Leibler (KL) distance between the local and global statistics of the objects versus the background. The boundary information is formulated using a color and texture edge map of the video frames. The shape information is calculated adaptively to the dynamic of the moving objects and permits tracking that is robust to background distractions and occlusions. Minimization of the energy functional is implemented using the level set method. We show the effectiveness of the approach for object tracking in color, infrared (IR), and fused color-infrared sequences.

**Keywords** Tracking · Region/Boundary/Shape information · Mixture models · KL-distance · Level sets · Color and infrared

## 1 Introduction

Object tracking is an important problem in computer vision and it has a variety of applications, such as coding, video surveillance, robotics, etc. It has been the focus of considerable research work in the last decades, centered around two

mainstream approaches. In the *motion-based* approach, 2D or 3D motion parameters are estimated for the moving objects in a video sequence [1,2]. Motion models such as similarity, affine or projective transformations are commonly used [2–4]. This approach works well when the objects undergo small deformations. However, its efficiency may drastically decrease when there are significant object structure changes, e.g., for non-rigid objects. In the *model-based* approach, the tracking of the moving objects on a current frame is performed by using the previous frame tracking result as a starting point. The moving objects are then tracked in each video frame by fitting a priori known parameters about the appearance of the objects [5,8–13].

Among the model-based tracking methods, active contours have gained popularity in the last few years [5–7,14,15,17]. With active contours, non-rigid bodies can be tracked thanks to the deformation capacity of the contours. Furthermore, with the advent of the level set formalism [18], changes in the topology of the objects are automatically handled. Among the tracking methods that used active contours in the past are those that assumed a static background for the video sequence. For example, in [15,16], the authors propose partitioning the difference image between two successive frames of the sequence in order to separate the moving objects from the background. In [7], a fast boundary-based method is proposed for object contour tracking. However, only the edge information is used as a cue for tracking, which makes the method sensitive to noise. Among the methods operating without the static background assumption, we find those described in [14,19,20,22]. In [14], the author proposes tracking an object by finding the correspondence of its pixel intensity between successive frames of a sequence. Although the method does not require calculation of object motion parameters, it may fail to track objects in the presence of texture, illumination changes and/or occlusions.

M. S. Allili (✉) · D. Ziou  
Department of Computer Science,  
Faculty of Science, University of Sherbrooke,  
J1K 2R1, Sherbrooke, QC, Canada  
e-mail: ms.allili@usherbrooke.ca

D. Ziou  
e-mail: d.ziou@usherbrooke.ca

The authors in [19,20,22] propose methods for tracking by matching the object intensity histogram between successive frames of a sequence using active contours. However, the histogram matching efficiency may drastically decrease if the object undergoes intensity variations due to noise or illumination changes. To make active-contour tracking robust to noise, the author in [21] uses an autoregressive model for describing object shape variation in image sequences. A parametric description for the object silhouettes is, however, required for the shape representation. Although this parametrization is realistic for a wide range of objects, it cannot be applied to arbitrary non-rigid objects. In general, the difficulties encountered by the above active-contour methods fall into three main groups. The first of these is related to cluttered backgrounds where the contours may be easily distracted. The second problem is related to texture. Indeed, intensity histograms are generally lacking in texture information, and thus the tracking models may fail if a moving object contains texture or moves over a textured or noisy area. Last, but not least, is the problem of the sensitivity of the tracking models to occlusions: the object may be partially or completely lost if an occlusion occurs. These three problems, which may or may not occur together in a given sequence, can drastically affect the performance of the tracking or even cause it to fail.

The present paper proposes an active-contour method which is capable of tracking multiple moving objects on non-static and cluttered backgrounds and efficiently handling inter-object occlusions. The method combines multiple cues from the image to track the objects. These cues include the color and texture of the objects, the image edge map and the shape of the objects. The tracking is formulated by minimizing an energy functional which combines region, boundary and shape information about the objects to find their boundaries in all the video frames. The region information is formulated by minimizing the distance between the local and global statistics of the objects and the background. The boundary information is formulated using a multi-band edge detector, which has the role of aligning the object contours with pixels having a high discontinuity in the region information. Finally, the shape information is formulated using the properties of level set contours, and has the role of mitigating distraction of the tracking in cluttered backgrounds and when object occlusions are encountered. In the proposed method, the initialization required is only segmentation of the objects in the first frame of the sequence. We have successfully applied the method for tracking multiple objects in color, IR and fused color-IR video sequences.

This paper is organized as follows: In Sect. 2, we present the proposed method for single-object tracking. In Sect. 3, we discuss the extension of the method to multiple-object tracking. In Sect. 4, we show a validation of the approach on examples of single and multiple-objects tracking and with different image modalities. In Sect. 5, we discuss some issues

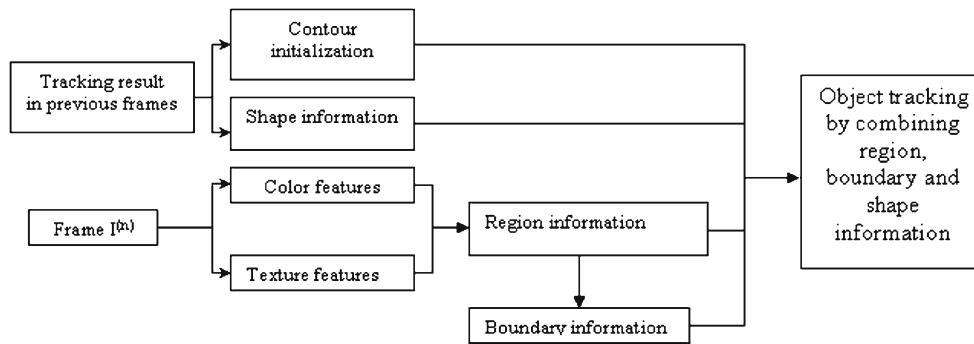
connected with the proposed method and some perspectives for future work.

## 2 Formulation of the model for single-object tracking

In what follows, we denote by  $I^{(n)}$  the  $n$ th frame in the image sequence. The aim of the proposed tracking method is to localize one or more moving object(s) in all the sequence frames  $I^{(n)}$  ( $0 < n < \infty$ ), given only the segmentation of the objects in the first frame  $I^{(0)}$  (reference frame), which is performed manually. Without loss of generality, we assume the scene is composed of a single object moving on a background (extension of the method to multiple objects is described in Sect. 3 of the paper). Let us denote by  $\Omega \subseteq Z^+ \times Z^+$  the domain of frame  $I^{(n)}$ ,  $n \geq 0$ , where  $Z^+$  is the set of positive integers, and let  $R_{\text{obj}}^{(n)}$  and  $R_{\text{bck}}^{(n)}$  represent, respectively, the sets of pixels constituting the object and the background in frame  $I^{(n)}$ . In what follows, we denote the contour of the object by  $\partial R_{\text{obj}}^{(n)}$  and the coordinates of a pixel  $(x, y)$  by  $\mathbf{x}$ . The aim of the tracking algorithm is to locate the object boundaries in frames  $I^{(n)}$ ,  $n > 0$ , of the sequence by using a combination of region, boundary and shape information about the object, given a priori knowledge of  $R_{\text{obj}}^{(0)}$  and  $R_{\text{bck}}^{(0)}$ . Figure 1 gives an outline of how the different information sources are extracted and integrated in the tracking algorithm. In the next sections, we give the details of their calculation and the formulation of the region, boundary and shape tracking modules.

### 2.1 Formulation of the region module

In principle, for color image sequences, a wide range of features could be used for region information, including color, texture, motion, etc. However, having a high-dimensional region vector may slow down the tracking. Thus, we use only color and texture features. For color, we use the CIE- $L^*a^*b^*$  space, which is perceptually uniform. For texture, we use features calculated from the correlogram of the pixel neighborhood [23]. An element of the matrix  $C^{v,\theta}(c_i; c_j)$  gives the probability that, given a pixel  $\mathbf{x}_1$  of color  $c_i$ , a pixel  $\mathbf{x}_2$  at distance  $v$  and orientation  $\theta$  from  $\mathbf{x}_1$  is of color  $c_j$ . From each matrix, we calculate four typical features: *inverse-difference-moment* (IDM), *energy* (EN), *entropy* (ET) and *correlation* ( $C$ ) (see the appendix for details on the calculation of these features). To capture a complete structure of the texture, we use four different orientations,  $\theta \in \left\{0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}\right\}$ , and two displacements,  $v \in \{1, 2\}$ , for the calculation of the correlograms. In what follows, we denote by  $\vec{\mathbf{U}}(\mathbf{x}) = (u_1(\mathbf{x}), \dots, u_d(\mathbf{x}))$  the vector composed of the combination of color and texture features calculated for the pixel  $\mathbf{x}$ . This vector contains seven dimensions (the first three components are for the color features and the remaining components are



**Fig. 1** An outline of the proposed tracking algorithm

for the texture features). Note that the above features are used only for color sequences. In the case where the tracking is performed using IR image sequences, the color is replaced by the image intensity and the correlograms by the co-occurrence matrices [25].

In practice, both object and background may consist of many classes of color and texture. Thus, we propose modeling them using two mixtures of Gaussian pdfs:  $\mathcal{M}_{obj}$  and  $\mathcal{M}_{bck}$ . The probability of observing a value of the region feature vector  $\vec{\mathbf{U}}(\mathbf{x}) = (u_1(\mathbf{x}), \dots, u_d(\mathbf{x}))$  in the reference frame  $I^{(0)}$  is given as follows:

$$f_{in}(\vec{\mathbf{U}}(\mathbf{x})|\Theta_{obj}) = \sum_{i=1}^{M_{obj}} \pi_i p(\vec{\mathbf{U}}(\mathbf{x})|\theta_i) \quad \text{if } (\mathbf{x}) \in \text{Object} \quad (1)$$

$$f_{out}(\vec{\mathbf{U}}(\mathbf{x})|\Theta_{bck}) = \sum_{j=1}^{M_{bck}} \xi_j p(\vec{\mathbf{U}}(\mathbf{x})|\varphi_j) \quad \text{if } (\mathbf{x}) \in \text{Background} \quad (2)$$

where  $f_{in}$  and  $f_{out}$  designate the mixture probabilities of the object and the background, respectively. The terms  $p(\vec{\mathbf{U}}(\mathbf{x})|\theta_i)$  and  $p(\vec{\mathbf{U}}(\mathbf{x})|\varphi_j)$  designate the probabilities of multivariate Gaussians, and  $M_{obj}$  and  $M_{bck}$  designate the numbers of classes contained in the object and the background mixtures. The symbols  $\pi_i$  and  $\xi_j$  designate the mixing parameters for the two mixture models  $\mathcal{M}_{obj}$  and  $\mathcal{M}_{bck}$ . The unknown parameters of the mixtures  $\Theta_{obj} = (\theta_i, \pi_i, i = 1, \dots, M_{obj})$  and  $\Theta_{bck} = (\varphi_j, \xi_j, j = 1, \dots, M_{bck})$  are calculated using maximum likelihood estimation (MLE) [26], which consists of getting the parameters of the two mixtures by maximizing the likelihood functions  $\mathcal{L}_{obj} = \prod_{\mathbf{x} \in R_{obj}^{(0)}} f_{in}(\vec{\mathbf{U}}(\mathbf{x})|\Theta_{obj})$  and  $\mathcal{L}_{bck} = \prod_{\mathbf{x} \in R_{bck}^{(0)}} f_{out}(\vec{\mathbf{U}}(\mathbf{x})|\Theta_{bck})$ . The problem of estimation thus becomes:

$$\hat{\Theta}_{obj} = \operatorname{argmax}_{\Theta_{obj}} [\mathcal{L}_{obj}] \quad (3)$$

$$\hat{\Theta}_{bck} = \operatorname{argmax}_{\Theta_{bck}} [\mathcal{L}_{bck}] \quad (4)$$

Here, we make the assumption that the region feature distribution for the object is stationary from frame to frame. The

background distribution, however, may vary if the camera is not static. Therefore, the mixture  $\mathcal{M}_{bck}$  is recalculated at regular time intervals. Finally, the numbers of classes  $M_{obj}$  and  $M_{bck}$  are calculated automatically by minimizing the MDL information-theoretic criterion [27].

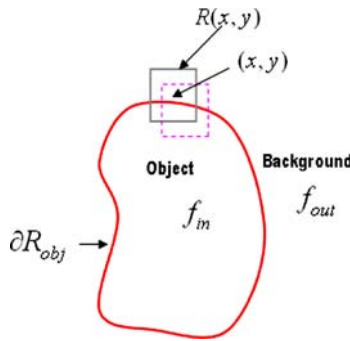
For each pixel  $\mathbf{x}$  in the current frame  $I^{(n)}, n > 0$ , let us model the region features of its neighborhood  $\mathbf{R}(\mathbf{x})$  using a mixture of pdfs  $\tilde{f}(\mathbf{x})$ . The size of the pixel neighborhood is chosen sufficiently high to have enough data (e.g.,  $11 \times 11, 13 \times 13$  pixels, ..., see Fig. 2). The parameters of the mixture are calculated similarly to Eqs. (3) and (4), by maximizing the likelihood of the local mixture on the neighborhood data of the pixel. To move the initial object contour from its position in  $I^{(n-1)}$  to the new object boundaries in  $I^{(n)}$ , we minimize the following energy functional according to the contour  $\partial R_{obj}$ :

$$E_r(R_{obj}) = \iint_{R_{obj}} \mathbf{D}(\tilde{f}(\mathbf{x}), f_{in})d\mathbf{x} + \iint_{R_{obj}^c} \mathbf{D}(\tilde{f}(\mathbf{x}), f_{out})d\mathbf{x} \quad (5)$$

where  $f_{in}$  and  $f_{out}$  are the distributions of the region features in the object and background of frame  $I^{(0)}$ , which are defined respectively by Eqs. (1) and (2). In the above functional, the region energy is expressed using a symmetric KL-distance which is formulated between two distributions  $p_1$  and  $p_2$  by:

$$\mathbf{D}(p_1, p_2) = \frac{1}{2} \left[ \text{KL}(p_1 \| p_2) + \text{KL}(p_2 \| p_1) \right] \quad (6)$$

with  $\text{KL}(p_1 \| p_2) = \int_z p_1(z) \log(p_1(z)/p_2(z)) dz$  and  $\text{KL}(p_2 \| p_1) = \int_z p_2(z) \log(p_2(z)/p_1(z)) dz$ , where the integrals are calculated over the domain of the variable  $z$ . We recall also that the KL-distance defines the Mutual Information (MI) between two random variables  $X$  and  $Y$  by:  $\hat{I}(X, Y) = \text{KL}(p(X, Y) \| p(X)p(Y))$ . The goal of using the energy functional (5) is to track the new boundaries of the object in  $I^{(n)}$  such that the resulting local statistics for each pixel neighborhood  $\mathbf{R}(\mathbf{x})$  within the object (resp. background) are “close” to the global statistics of the object (resp.



**Fig. 2** Illustration of the contour region energy formulated by comparing the local and global statistics of the object and the background

background). Theoretically, minimizing the above energy amounts in some way to minimizing the description length of the image code [27]. A minimal code for a pixel neighborhood  $\mathbf{R}(\mathbf{x})$  will be attained if we assign the pixel  $\mathbf{x}$  to the object when  $\mathbf{D}(\tilde{f}(\mathbf{x}), f_{in}) < \mathbf{D}(\tilde{f}(\mathbf{x}), f_{out})$  and to the background, otherwise. Formulating the region information in this way has several advantages that we can summarize as follows:

- The region vector combines texture and color features, which allows for better separability between the object and the background when they contain different textures but the same color and vice versa. Further, texture features carry information about the local structure of the object (resp. the background) which makes the method more robust to noise.
- Formulating the pixel assignment by using a distance between the local and the global statistics is advantageous in that it also increases the method's robustness to noise and illumination changes. Note that the approaches in [14, 22] can be considered as particular cases of our method since they perform the tracking by using a pixel-wise intensity matching between successive frames of a sequence.

## 2.2 Formulation of the boundary module

In practice, there may be a strong correlation between the object boundaries and the image edge map. Moreover, the presence of illumination changes may cause slight variations in the region information of the object, but the boundary information can still describe the object since it depends only on the image derivatives. Therefore, we propose to exploit the boundary information of the image, together with the region features, to steer the object tracking. To calculate the boundary information of the multi-band image  $\vec{\mathbf{U}}(\mathbf{x})$ , we use the approach proposed in [28] which defines the boundary plausibility as follows. Consider the matrix  $\mathbf{J}$  formed by the

derivatives of the components of the vector  $\vec{\mathbf{U}}(\mathbf{x})$  according to the coordinates  $x$  and  $y$ , respectively:

$$\mathbf{J} = \begin{pmatrix} \partial u_1 / \partial x & \dots & \partial u_d / \partial x \\ \partial u_1 / \partial y & \dots & \partial u_d / \partial y \end{pmatrix} \quad (7)$$

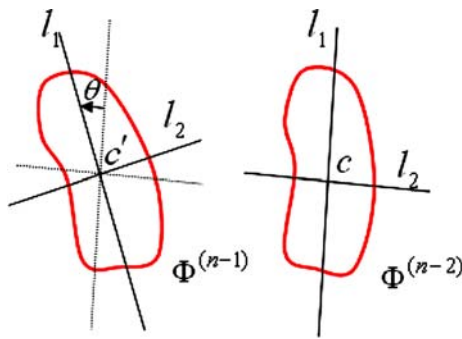
The greatest change in  $\vec{\mathbf{U}}(\mathbf{x})$  coincides with the largest eigenvalue (in absolute value) of the  $2 \times 2$ -matrix  $\mathbf{J}\mathbf{J}^T = \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix}$ , given by  $\lambda_{\max} = \frac{1}{2}(b_{11} + b_{22} + \sqrt{(b_{11} - b_{22})^2 + 4b_{12}^2})$ , and the direction of change coincides with the corresponding eigenvector. The magnitude of this change will constitute the boundary plausibility of the multi-band image made of the components of the vector  $\vec{\mathbf{U}}(\mathbf{x})$  at each pixel  $\mathbf{x}$ . In what follows, we denote the boundary plausibility at pixel  $\mathbf{x}$  by  $\mathbf{P}(\mathbf{x})$ , which is equal to  $|\lambda_{\max}|$ . To align the object contour  $\partial R_{obj}$  with pixels having high boundary plausibility, we propose to minimize the following energy functional:

$$E_b(\partial R_{obj}) = \int_0^{\mathcal{L}(\partial R_{obj})} g(\mathbf{P}(s)) ds \quad (8)$$

where  $s$  and  $\mathcal{L}(\partial R_{obj})$  represent, respectively, the arc-length parameter and the length of the contour  $\partial R_{obj}$ . The symbol  $g$  denotes a strictly decreasing function which is given by  $g(\mathbf{P}(s)) = \frac{1}{\mathbf{P}(s)+1}$ . The boundary energy reaches its local minimum when the contour is aligned with pixels having a high boundary plausibility  $\mathbf{P}(s)$ . The minimization of (8) also allows the contour to be kept smooth during its evolution [29].

## 2.3 Formulation of the shape module

In situations where the moving object and some parts of the background share the same color and texture properties, the object contour may be distracted by the erroneous inclusion of pixels from the background, thereby drifting away from the real object boundaries. Adding prior shape information for the object is valuable in this case to prevent distraction of the contour. To formulate the shape information, we resort to the formalism of level sets to represent the object contour [18]. Let  $\Phi : \mathfrak{R}^2 \rightarrow \mathfrak{R}$  be a level set distance function, i.e.,  $\Phi(\mathbf{x}) = \pm d(\mathbf{x})$ , where  $d(\mathbf{x})$  denotes the Euclidian distance between the pixel  $\mathbf{x}$  and the zero level set. The sign of  $\Phi(\mathbf{x})$  is negative if  $\mathbf{x}$  is inside the zero level set and positive if it is outside. The object contour  $\partial R_{obj}$  is represented by its zero level set, i.e.,  $\partial R_{obj} = \{\mathbf{x} = (x, y) | \Phi(\mathbf{x}) = 0\}$ . In what follows, we denote by  $\Phi^{(n)}$  the level set representing the object contour in the frame  $I^{(n)}$ . Following the work proposed in [30], suppose  $\Phi$  and  $\Phi'$  are two level set functions. To minimize the distance between  $\Phi$  and  $\Phi'$ , the authors propose



**Fig. 3** Illustration of the calculation of the Euclidian transformation parameters for the object contour

minimizing the following distance:

$$E_s(\Phi) = \iint_{\Omega} (\Phi(\mathbf{x}) - \Phi'(\mathbf{x}))^2 \frac{h(\Phi(\mathbf{x})) + h(\Phi'(\mathbf{x}))}{2} d\mathbf{x} \quad (9)$$

where  $h(\Phi(\mathbf{x})) = \frac{H(\Phi(\mathbf{x}))}{\iint_{\Omega} H(\Phi(u,v)) du dv}$  and  $H$  is the Heaviside function ( $H(\Phi(\mathbf{x})) = 1$  if  $\mathbf{x}$  is in the interior of the zero level set, and 0 otherwise). The minimization of the above distance according to  $\Phi$  permits us to align the function  $\Phi$  with  $\Phi'$ . Note, however, that directly using function (9) in its actual form to maintain the similarity of the object contour  $\Phi^{(n)}$  with  $\Phi^{(n-1)}$  is not realistic, since the shape of the object may undergo natural changes due to object motion or camera operations such as zooming, tilting, etc. Thus, when applying the above shape energy to the contour, we should differentiate between object contour modifications due to motion of the object or zooming for example, and modifications due to distractions by the background.

To achieve this goal, we compare the level set  $\Phi^{(n)}$  to a version of  $\Phi^{(n-1)}$  transformed using Euclidian similarity transformation with the parameters  $(r, R_{\theta}, T)$ , where  $r$  represents the scaling factor, and  $R_{\theta}$  and  $T$  represent its rotation matrix and translation vector. These parameters are estimated based on the object contour in frames  $I^{(n-2)}$  and  $I^{(n-1)}$ . The parameter  $T$  is approximated using the difference between the centers of gravity in  $\Phi^{(n-2)}$  and  $\Phi^{(n-1)}$ , that we denote respectively by  $c$  and  $c'$ . The scaling factor is estimated using the fraction  $r = |\bar{R}_{obj}^{(n-1)}|/|\bar{R}_{obj}|$ , where  $||$  denotes the cardinality of a set and  $|\bar{R}_{obj}|$  designates the average object size in the  $L$  frames preceding  $I^{(n-1)}$ . For the rotation matrix  $R_{\theta}$ , we first calculate the principal axis of the objects represented by  $\Phi^{(n-2)}$  and  $\Phi^{(n-1)}$ . The first principal axis  $l_1$  is determined by the line joining the farthest pair of points in the object contour and passing through its center of gravity. The second axis  $l_2$  is defined by the line perpendicular to the first axis that passes through the center of gravity; see Fig. 3 for an illustration. Finally, the angle of rotation between  $\Phi^{(n-2)}$  and  $\Phi^{(n-1)}$  is approximated by the rotation angle of the coordinate system formed by the axes  $l_1$  and  $l_2$ .

### 2.4 Combined energy functional for tracking

By combining the region, boundary and shape modules in one module, we obtain the following energy functional:

$$E_{tot}(\partial R_{obj}^{(n)}) = \alpha E_r(\partial R_{obj}^{(n)}) + \beta E_b(\partial R_{obj}^{(n)}) + \gamma E_s(\partial R_{obj}^{(n)}) \quad (10)$$

where  $\alpha$ ,  $\beta$  and  $\gamma$  are weighting parameters that balance the contribution of the different energy terms. The minimization of the above energy is performed using Euler–Lagrange equations and implemented using level set functions. The motion equation for the object contour  $\Phi^{(n)}$  is given by:

$$\frac{d\Phi^{(n)}(\mathbf{x}, t)}{dt} = \left( \alpha V_b(\Phi^{(n)}) + \beta V_r(\Phi^{(n)}) + \gamma V_s(\Phi^{(n)}) \right) \|\nabla \Phi^{(n)}\| \quad (11)$$

with

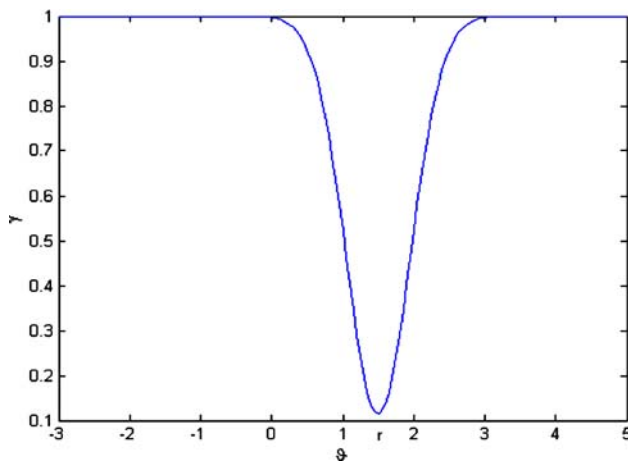
$$V_b(\Phi^{(n)}) = g(\mathbf{P}(\mathbf{x}))\kappa(\mathbf{x}) + \nabla g(\mathbf{P}(\mathbf{x})) \cdot \frac{\nabla \Phi^{(n)}}{\|\nabla \Phi^{(n)}\|} \quad (12)$$

$$V_r(\Phi^{(n)}) = -\mathbf{D}(\tilde{f}(\mathbf{x}), f_{in}) + \mathbf{D}(\tilde{f}(\mathbf{x}), f_{out}) \quad (13)$$

$$V_s(\Phi^{(n)}) = -\Phi_0(h(\Phi^{(n)}) + h(\Phi')) - \frac{\delta(\Phi^{(n)})}{2 \iint_{\Omega} H(\Phi^{(n)}) d\mathbf{x}} \times \left( \Phi_0^2 - \iint_{\Omega} \Phi_0^2 h(\Phi^{(n)}) d\mathbf{x} \right) \quad (14)$$

with  $\Phi_0 = (\Phi^{(n)} - \Phi')$  and  $\delta(\cdot)$  is the delta function. In Eq. (14), we denote by  $\Phi'$  the transformed level set contour  $\Phi^{(n-1)}$  obtained using the transformation parameters calculated in the last section. The symbol  $\kappa$  designates the curvature of the zero level set, which is given by  $\kappa = \text{div}(\nabla \Phi^{(n)} / \|\nabla \Phi^{(n)}\|)$ , where  $\text{div}$  is the vector divergence operator. The numerical implementation of Eq. (11) is performed using the Narrow-Band algorithm [31]. In this algorithm, all the derivatives in Eq. (11) are approximated using finite differences. The level set function is re-initialized when the zero level (i.e., the object contour) reaches the boundary of the band. In our method, we count the period between two re-initializations as one iteration and we recalculate the shape weight  $\gamma$  for each narrow band, as explained in the following paragraph.

As mentioned before, the shape information is added to mitigate distraction of the object contour when the moving object and the background share the same region information. To allow for the shape module to play this role, we set the parameter  $\gamma$  that determines the contribution of the shape information as follows. Let us define the parameter  $\vartheta = |\bar{R}_{obj}^{(n)}|/|\bar{R}_{obj}|$  that determines the area change of the object in the current frame  $I^{(n)}$ , where  $|\bar{R}_{obj}|$  is the average size of the object in the last  $L$  frames ( $I^{(n-L-1)}, \dots, I^{(n-1)}$ ). We should set a high value for  $\gamma$  if we have the following



**Fig. 4** Graph of the function  $\gamma(\vartheta)$  with  $a = 1$ ,  $r = 1.5$  and  $\sigma = 0.45$

situation: ( $\vartheta \ll r$  or  $r \ll \vartheta$ ), where  $r$  is the scaling factor estimated in the last section. This is achieved by using the following lenient function for setting the parameter  $\gamma$ :

$$\gamma(\vartheta) = a \left( 1 - (\sigma\sqrt{2\pi})^{-1} \exp\left(-(\vartheta - r)^2/2\sigma^2\right) \right) \quad (15)$$

where  $\sigma$  controls the sensitivity of  $\gamma$  to the deviation of the object size, defined by  $\vartheta$ , from the tolerated value  $r$ . The parameter  $a$  controls the contribution of the shape module to Eq. (11). In Fig. 4, we show the graph of the function  $\gamma(\vartheta)$ , setting  $a = 1$ ,  $r = 1.5$  and  $\sigma = 0.45$ . Using the function (4) to set the contribution of the shape information in the tracking has the objective of maintaining the consistency of the object shape between the last  $L$  frames and the current frame by penalizing abrupt size changes for the object.

A remark should be made, however, for the shape constraint in scenarios of movements where a rapid shape change occurs (e.g., rapid hand gestures). In fact, the predicted shape  $\Phi'$  with which the current contour is compared [see Eq. (9)] may not perfectly reflect the real shape of the object in  $I^{(n)}$ . Nevertheless, as long as the shape module weight  $\gamma$  remains low, if the change in the area of the object is not large [see Eq. (15)], the region and boundary modules will make a stronger contribution to drive the object contour, allowing the contour to be fitted to the real object boundaries.

### 3 Extension to multiple-objects tracking

Here, we assume that the image sequence contains  $N$  different moving objects  $R_{\text{obj}_1}, \dots, R_{\text{obj}_N}$  lying on the background  $R_{\text{bck}}$ . The number of objects is given initially by the user or can be detected using any background subtraction algorithm, if the camera is stable. Following the formulation of single-object tracking, we define a similar energy functional (10) for

each object. It remains true, however, that the objects may contact, or even occlude, each other.

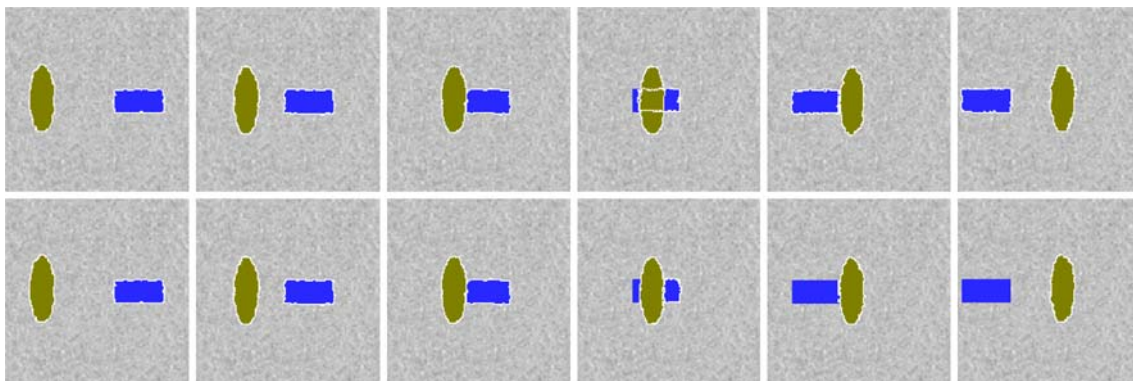
To detect the occurrence of an occlusion, we use the properties of the level set functions representing the contours of the objects. The distance function  $\Phi_k^{(n)}$  of the object  $R_{\text{obj}_k}$  is negative for pixels inside the object and positive for pixels outside. An occlusion occurs between an object  $R_{\text{obj}_k}^{(n)}$  and  $R_{\text{obj}_h}^{(n)}$  if some pixels have negative distance values for both of the functions  $\Phi_k^{(n)}$  and  $\Phi_h^{(n)}$ ,  $k \neq h$ . Let  $N_0$  be the number of these pixels, which could be considered also as the number of occluded pixels of one of the objects. We set the constant  $a$  in the function (15) for the objects  $R_{\text{obj}_k}^{(n)}$  and  $R_{\text{obj}_h}^{(n)}$  to  $a = a_0 + N_0/|R_{\text{obj}_k}^{(n)}|$  and  $a = a_0 + N_0/|R_{\text{obj}_h}^{(n)}|$ , where  $a_0$  is the initial value of the constant  $a$ , which we set to 0.3. If an occlusion occurs, the weight  $\gamma$  of the shape module will be pushed to increase according to the function (15), which increases the contribution of the shape information in order to conserve the shape of the objects momentarily until the end of the occlusion.

To illustrate how this works, Figs. 5 and 6 show two tracking examples in which two kinds of occlusion occur. In Fig. 5, the sequence contains 25 frames and there are two moving objects: the ellipse and the rectangle. The rectangle moves from the right to the left side of the image and passes behind the ellipse. The first row of the figure shows the tracking results obtained for a sample of frames using our method with the shape information. We set  $\sigma = 0.35$  in function (15) and  $L$  to four frames. The occluded object has kept its shape and was not lost after the occlusion. The tracking results shown in the second row of the figure have been obtained for the same frames in the top row, but by suppressing the shape information in the Eq. (11); i.e., we set  $\gamma = 0$ . The object has been lost after the occlusion because of the contour shrinkage.

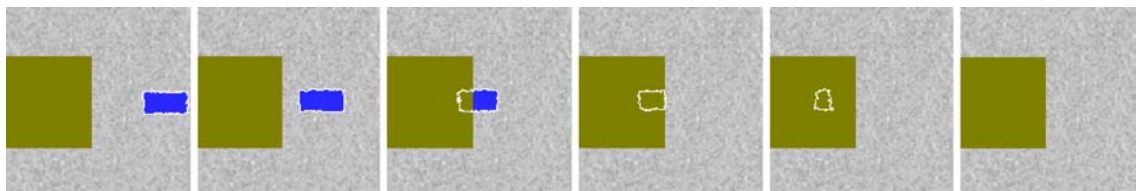
The video sequence in Fig. 6 contains 25 frames and the moving object (the rectangle) has been totally occluded by the background. Here, the constant  $a$  remains unchanged from its initial value  $a_o = 0.3$  since the occlusion is not caused by another moving object. Notice that although the real object has disappeared its contour remained for a few frames during the occlusion because of the shape constraint, and finally disappeared. This permits an understanding of the role of the shape module that constrains the object contour to change smoothly in shape from frame to frame. That is, if a total occlusion occurs and lasts only for a short time, there is a possibility of recovering the occluded object.

### 4 Experiments

In all the experiments that we have conducted, in order to reduce the image histogram dimension and simplify the calculation of texture information, each band of the color space



**Fig. 5** Illustration of the advantage of using shape information in coping with occlusions. The first and second rows represent, respectively, the tracking of two objects with and without the use of shape information. From *left to right* in each row, frames 0, 5, 8, 12, 16 and 21 are shown



**Fig. 6** Illustration of tracking with a total occlusion of the moving object by the background. From *left to right*, frames 0, 4, 8, 13, 18 and 22 are shown

was quantized to 32 bins:  $\{0, \dots, 31\}$ . We set the size of the neighborhood  $\mathbf{R}(\mathbf{x})$  to  $13 \times 13$  pixels. The same neighborhood size was adopted for the calculation of the texture features of each pixel, as shown in Sect. 2.1. To quantitatively evaluate the performance of the proposed method, we used two different criteria. The first criterion measures the accuracy of the spatial location of the tracked objects. Suppose  $R_{\text{obj}}^{(n)}$  is an object location that results from the tracking in frame  $I^{(n)}$ , and  $G_{\text{obj}}^{(n)}$  is the ground-truth obtained by hand segmenting the object in the same frame. The object location error is given by:

$$\epsilon_1 = \frac{|(G_{\text{obj}}^{(n)} - R_{\text{obj}}^{(n)}) \cup (R_{\text{obj}}^{(n)} - G_{\text{obj}}^{(n)})|}{|G_{\text{obj}}^{(n)}| + |R_{\text{obj}}^{(n)}|} \quad (16)$$

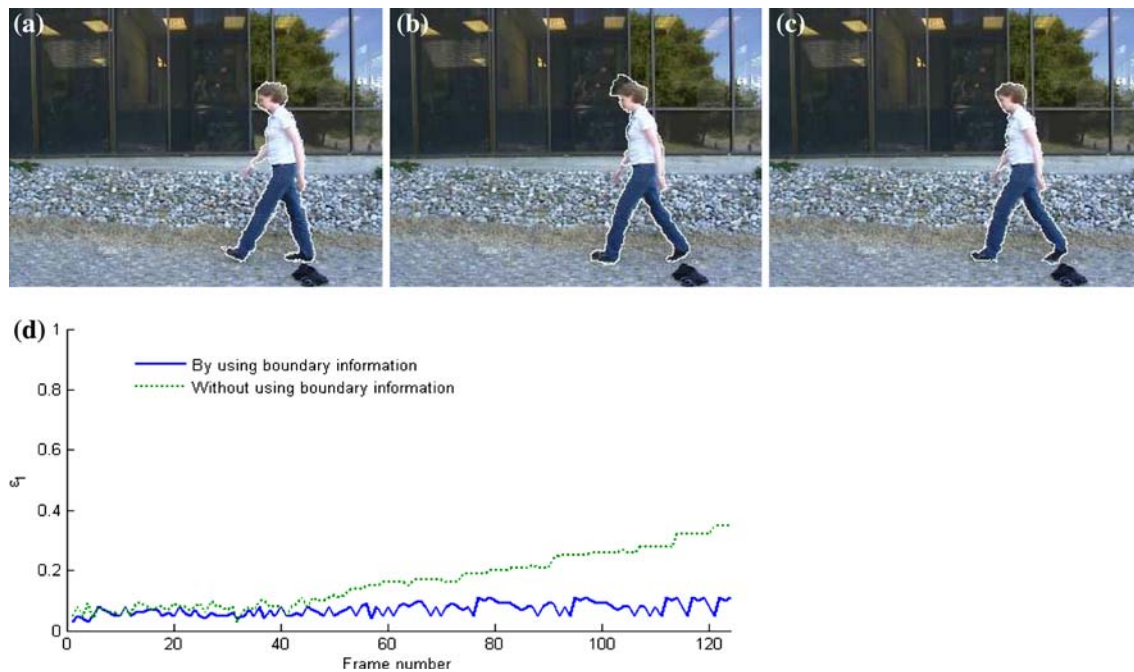
where “ $-$ ” denotes the set difference operator. The error  $\epsilon_1 \in [0, 1]$  measures the union of the two contours minus their intersection, all divided by the sum of their areas. In other words, this error tells us the percentage of misclassified pixels. The second criterion, introduced in [32], measures the discrepancy between an object reference histogram  $H_{\text{ref}}$  (i.e., in frame  $I^{(0)}$ ) and the histogram of the same object  $H^{(n)}$  obtained after tracking in frame  $I^{(n)}$ . The histogram discrepancy is given by:

$$\epsilon_2 = \sqrt{\frac{\sum_c [r_1 H^{(n)}(c) - r_2 H_{\text{ref}}(c)]^2}{NH^{(n)} + NH_{\text{ref}}}} \quad (17)$$

where  $r_1$  and  $r_2$  are scaling parameters used to normalize the data when the total number of elements in the two histograms

is different. They are given by  $r_1 = \sqrt{\frac{NH_{\text{ref}}}{NH^{(n)}}}$ ,  $r_2 = \frac{1}{r_1}$ , where  $NH = \sum_c H^{(n)}(c)$  and  $NH_{\text{ref}} = \sum_c H_{\text{ref}}(c)$  (the summation here is made over the color bins  $c$ ). The range of the error  $\epsilon_2$  is  $[0, 1]$ , where  $\epsilon_2 = 0$  corresponds to identical histograms. Note that when we evaluate the tracking performance in a sequence, the errors  $\epsilon_1$  and  $\epsilon_2$  are averaged for each frame over the number of tracked objects.

In the first experiment, we illustrate the importance of the boundary information for our object tracking method. Figure 7 shows an example where we run our tracking algorithm with and without using the boundary information [setting  $\alpha$  to 0.3 and 0, respectively, in Eq. (11)]. We keep the combination of the region and the shape information. The sequence contains 123 frames. The mixture models of the object and the background are made, respectively, of three and four components. As an illustration, we show the tracking result in frames 47 and 48: the left image of each row shows the initialization from frame 47, the middle and right images, respectively, show the tracking result in frame 48 without and with the boundary information. Notice the leak in the object contour in Fig. 7b, where the tracking result was obtained without using the boundary information. The object contour has erroneously assigned some pixels to the background because the object color in this area is similar to that of the background. Having the boundary information allowed us to correct the contour position by aligning it with the real object boundaries. Notice also the presence of a coarse texture in the background of the images. However, since the region information combines color and texture



**Fig. 7** Example showing the importance of boundary information for tracking: **a** represents the contour initialization from the previous frame, **b**, **c** show tracking results without and with the boundary information. The graph in **d** represents the error  $\epsilon_1$  for the sequence

features, the algorithm has succeeded in tracking the feet of the walking girl. Finally, we hand segmented the sequence; Fig. 7d shows the graph of the object location error  $\epsilon_1$ . Using the boundary information did in fact reduce the error in the object location. We should note, however, that setting very high values for  $\alpha$  would make the tracking very sensitive to spurious edges. After several tests, the values  $\alpha = 0.3$  and  $\beta = 0.3$  gave the best performance and a good compromise between boundary and region information. These values are adopted in the rest of the experiments.

To show the robustness of the proposed method to noise, an example is presented in Fig. 8 where a Gaussian noise with zero mean and standard deviation  $\sigma_n = 3$  was added to the sequence shown in Fig. 7. We ran the tracking on the noisy sequence with the combination of region, boundary and shape information. We performed the tracking on the same sequence using the method proposed in [22], in which the object is tracked by matching the object histogram from frame to frame using the level set method. Figure 9 shows the graph of errors  $\epsilon_1$  and  $\epsilon_2$  for the two methods, calculated for the sequence with added noise with  $\sigma_n = 3$  and  $\sigma_n = 5$ , respectively. Compared to histogram matching, we note that for both the errors  $\epsilon_1$  and  $\epsilon_2$ , our method permits a more accurate spatial location and preservation of the region properties of the object (see the graphs in Fig. 9). Note also that, in the graph of  $\epsilon_1$  when  $\sigma_n = 5$ , the error reached the extreme value of 1 for histogram matching from frame 110, which is reflected in the tracking by the loss of the object. The robustness shown by our method can be explained partly by its use

of pixel classification based on local statistics, which is less sensitive to noise than pixel-wise classification. On the other hand, using the prior shape added a constraint to the contour which prevented it from having large distractions.

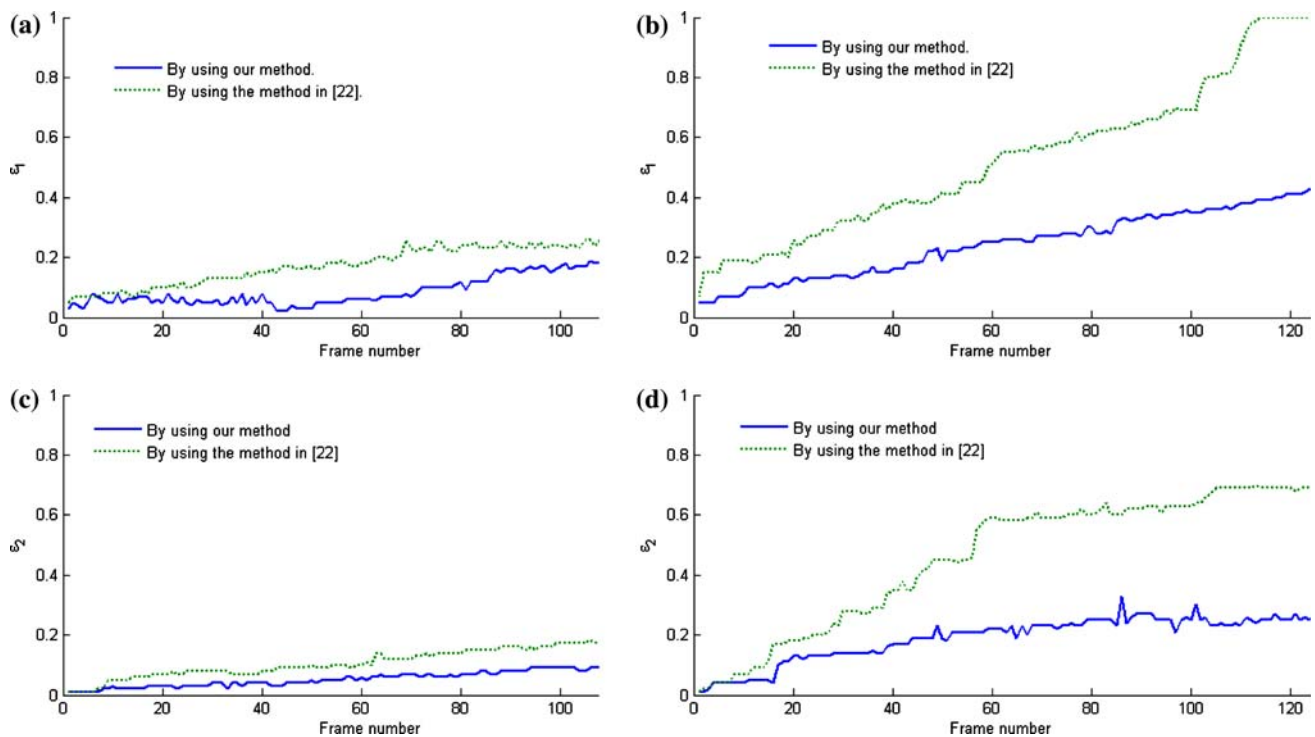
Figure 10 shows another example of tracking performed using our method and the method proposed in [22]. The video sequence is composed of 260 frames, where the moving object (the skater) and the background are modeled using mixtures of two and four components, respectively. Here, since the background changes rapidly, we recalculated the mixture of the background every five frames. We set the parameters  $\sigma = 0.35$  and  $a_0 = 0.3$  in function (15),  $L$  to 4 frames and  $\alpha = \beta = 0.3$  in Eq. (11). In the shown frames, 28 and 118, notice a contact between the object and a part of the background which has similar region properties to the object. Using the method in [22], the contour erroneously included background pixels inside the object and we had to re-initialize the object contour twice to continue the tracking. The distraction was successfully overcome using our method, where the contour has captured the true boundaries of the object thanks to the shape module. The same experiment was performed with the “fish” sequence shown in Fig. 11, keeping the same parameters for the algorithm. The sequence contains 50 frames and there are two moving objects (the two fish), each modeled using a mixture of one component. The background is modeled using a mixture of four components. The two objects were successfully tracked using our method, whereas a severe distraction occurred with the method in [22]. Figure 12 shows the graphs of the errors





**Fig. 8** An example showing robustness of the tracking to noise. In row **a**, we show four original frames of a sequence (40, 46, 52 and 63, from left to right). In row **b**, we show the tracking result obtained using the

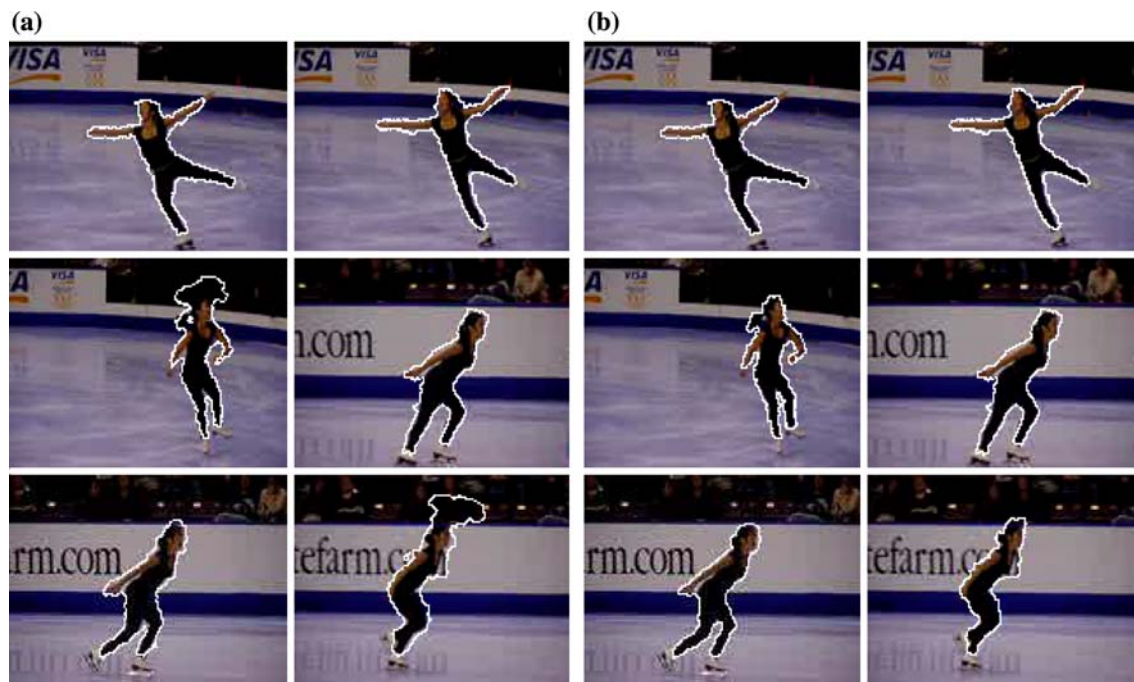
method in [22]. In row **c**, we show the tracking result obtained using our method



**Fig. 9** Graphs of the errors  $\epsilon_1$  and  $\epsilon_2$  for the noisy sequence in Fig. 8: **a**, **b** show the graphs of  $\epsilon_1$  for  $\sigma_n = 3$  and  $\sigma_n = 5$ ; **c**, **d** show the graphs of  $\epsilon_2$  for  $\sigma_n = 3$  and  $\sigma_n = 5$ , respectively

$\epsilon_1$  and  $\epsilon_2$  for the two sequences. Note how the distraction of the contour for [22] increased both errors  $\epsilon_1$  and  $\epsilon_2$  for the two examples, while for our method they remained free of abrupt increases. Note also that in general for both methods

the errors increase slightly with the frame number, which could be explained by the initialization of the object contour in the current frame using the most recent tracking result, which has the effect of accumulating the errors.



**Fig. 10** First example showing the role of the shape module in mitigating distraction of the object contour by the background. The tracking is performed using: **a** the method proposed in [22] and **b** our method.

We show in **a** and **b**, from left to right and top to bottom, the tracking results in frames 19, 22, 28, 112, 115 and 118

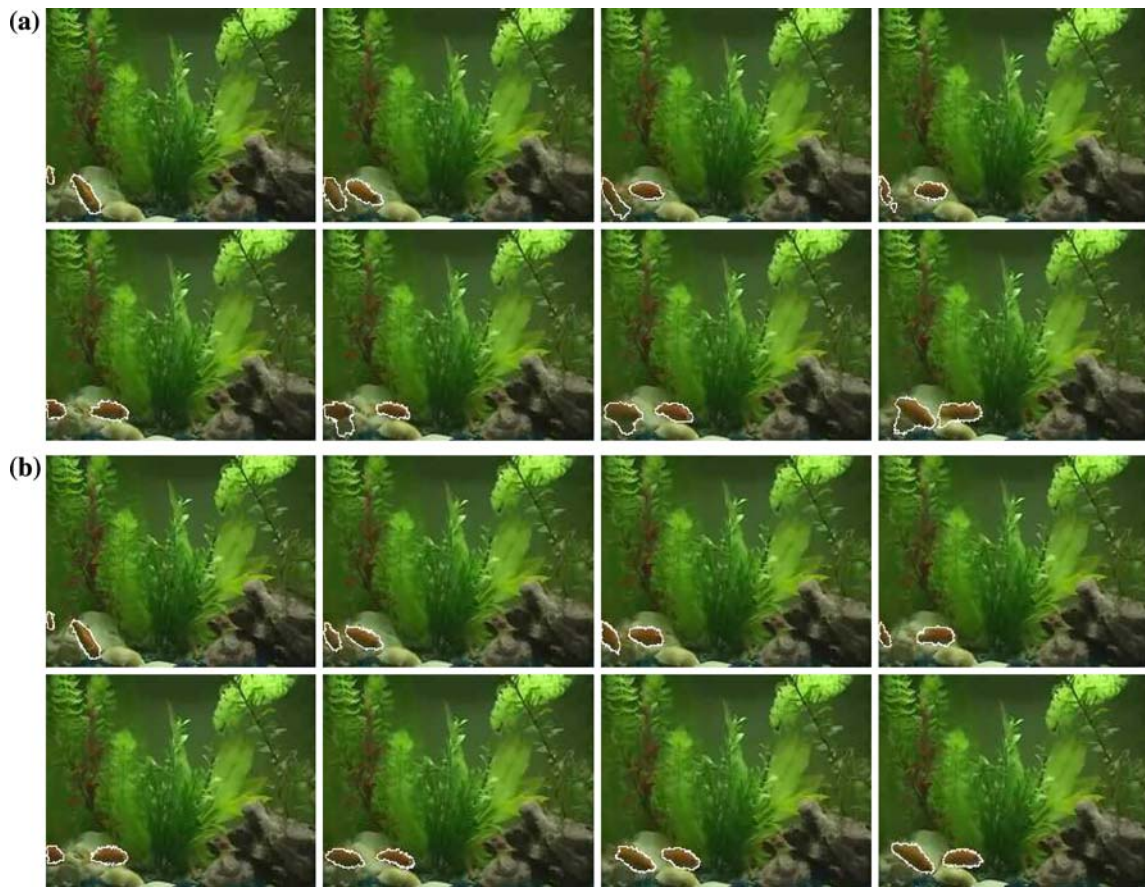
In Fig. 13, we show a tracking example on an IR video sequence containing 650 frames. The sequence is downloaded from the OTCBVS Benchmark Dataset Collection. In the sequence, two objects (pedestrians) composed of one class each are tracked on a background composed of three classes. An occlusion between the two objects occurs between frames 400 and 415 of the sequence. We ran the tracking algorithm with the same parameters as in the previous examples. The tracking of the two objects was successful for all the frames of the sequence. Below each image we show the value of the parameter  $\gamma$ , that we denote by  $\gamma_1$  for the object that moves from left to right and  $\gamma_2$  for the second object. Finally, Fig. 14 shows an example of tracking of two objects where we used a combination of region and boundary features in the color and IR images of two sequences. The sequences were also downloaded from the OTCBVS Benchmark Dataset Collection and contain 1,500 frames each. The mixtures of the two moving objects are made of two components each, while the mixture of the background has five components. Note that performing the tracking using only the color sequence is very challenging since the background has similar color features to some parts of the objects. Adding the IR information enhanced the discrimination power of the region features between the moving objects (pedestrians) and the background. We considered the IR image as one band and calculated the texture features using the co-occurrence matrix. We concatenated in one vector the

color and texture features extracted from the color and IR images. In the sequence, an occlusion has occurred between the two tracked objects from frame 53 to frame 71. As shown in the figure, the objects were successfully tracked using the proposed method. We also show the value of the parameter  $\gamma$ , denoted by  $\gamma_1$  for the object moving from left to right and by  $\gamma_2$  for the object moving from right to left.

Finally, Table 1 gives the different parameters involved in the proposed method, with their ranges and recommended values for tracking. A column has been included in the table to indicate whether or not each parameter has an influence on the computation time of the algorithm.

#### 4.1 Limitations of the proposed method

In the preceding examples, we demonstrated the success of the proposed tracking for several sequences with noise, cluttered backgrounds and inter-object occlusions. The success of the method also results from the satisfaction of our initial assumptions related to the constancy of the object's photometric properties (conservation of the object color) and the swift, short-lived character of the occlusions. In the absence of these assumptions, the tracking may fail, as illustrated in Figs. 15 and 16. In Fig. 15, we show the result of an experiment where the tracking sequence contains 53 frames of size  $166 \times 288$  and there is a single moving object modeled by a mixture of two components, while the background



**Fig. 11** Second example showing the role of the shape module in mitigating distraction of the object contour by the background. The tracking is performed using: **a** the method proposed in [22] and **b** our method.

We show in **a** and **b**, from *left to right* and *top to bottom*, the tracking results in frames 2, 7, 11, 15, 20, 25, 30 and 35

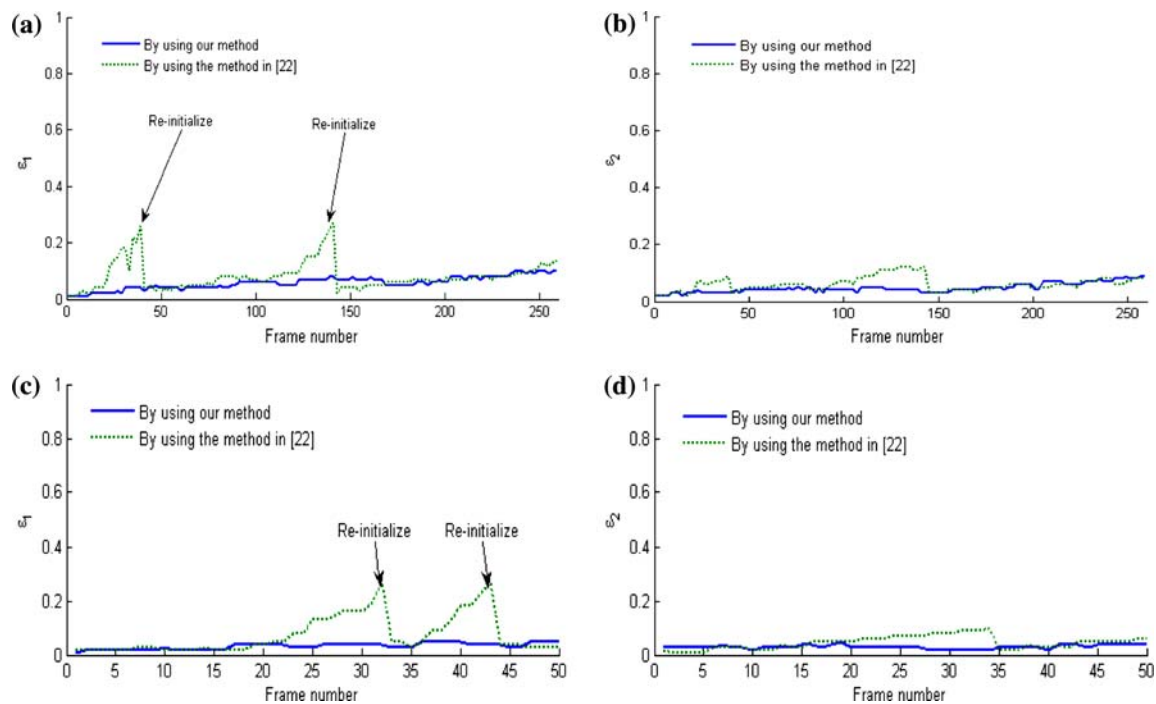
**Table 1** Table giving the different parameters involved in the method, their ranges and the recommended values

Parameters	Range	Recommended value	Influence on computation time
$\alpha, \beta, a_0$	$[0, 1]$	0.3, 0.3, 0.3	Yes
$N$	$\mathbb{Z}^+, \neq 0$	$\{1, \dots, 10\}$	Yes
$ \mathbf{R}(\mathbf{x}) $	$9 \times 9, \dots, 19 \times 19$	$13 \times 13$	Yes (off-line)
$L$	$\mathbb{Z}^+, \neq 0$	4–8	No
$\sigma$	$\mathbb{R}^+, \neq 0$	0.35	No
Video resolution	Any	–	Yes

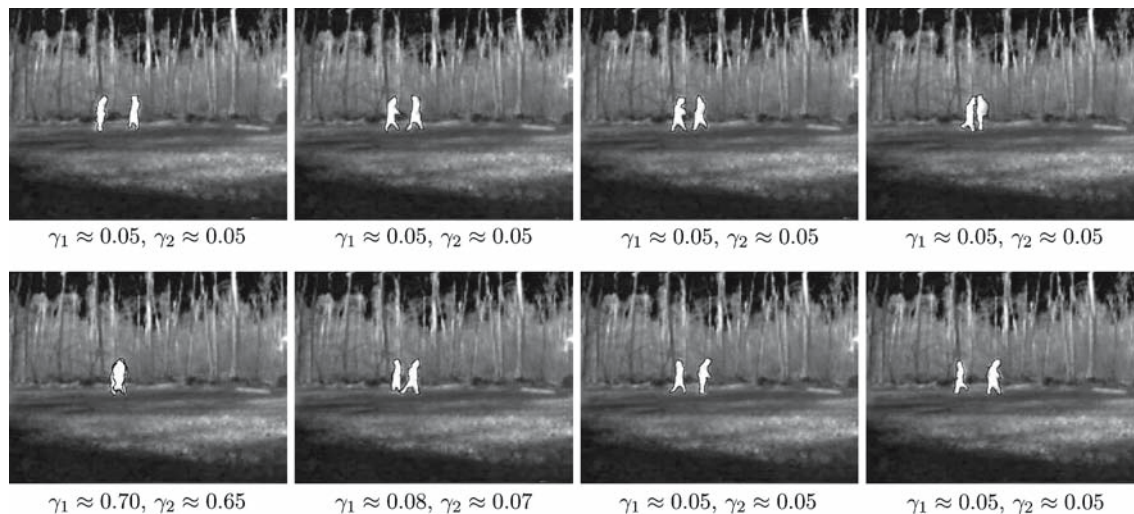
mixture has five components. To show the robustness of our method to illumination changes, we changed the brightness of each frame from column 178–288 using a Gamma correction; that is, we first normalized each color band in the interval  $[0, 1]$  using a linear dynamic change, then we raised the resulting values to different powers  $\nu$ , where  $0 \leq \nu \leq 1$ . Note that the smaller the value of  $\nu$ , the brighter the resultant image. Finally, we performed another linear dynamic change to restore the initial color range  $\{0, 1, \dots, 31\}$ . We observe in Fig. 15 that the tracking is quite robust for higher values

of  $\nu$ . However, accuracy begins to drop when the change of the brightness is severe (see Fig. 15d). One solution that we propose to address this problem is to use adaptive mixture models to represent objects with varying photometric properties. The model in [6] could be adapted, for example, to evolve the object contours adaptively to data, and boundary and shape information could be added to the model to cope with background distractions and occlusions.

Figure 16 shows two examples of tracking for the same sequence, synthetically creating two occlusions with differ-



**Fig. 12** **a, b** show the graphs of the errors  $\epsilon_1$  and  $\epsilon_2$  for the sequence in Figs. 10c and d show the same graphs for the sequence in Fig. 11

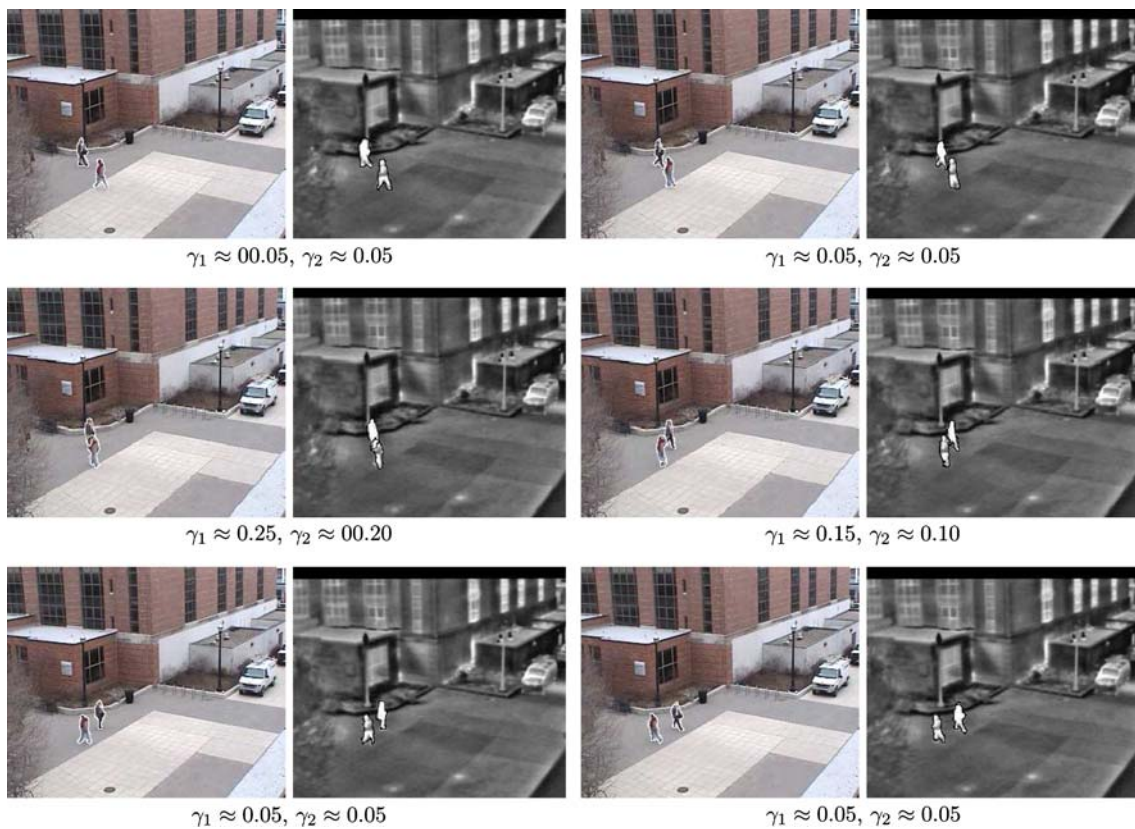


**Fig. 13** Example of multiple object tracking on an IR video sequence using the proposed method. The images represent the tracking results in frames (from left to right and top to bottom) 381, 389, 391, 399, 402, 418, 425 and 427

ent width for the moving object. The first occlusion occurred between frames 24 and 29 and the object was recovered after the occlusion thanks to the shape module. The second occlusion occurred between frames 24 and 37 and the object was lost. This example shows one of the main limitations of the proposed method when the occlusion is not short-lived. One solution that could be advocated for this problem is to com-

bine detection and tracking and automatically re-initialize the contour of the object after the occlusion to recover the tracking.

The last limitation of our method is common to most of active-contour approaches. When performing the tracking in a current frame, a part of the tracked object must initially be contained within the contour; otherwise, the contour may



**Fig. 14** Example of multiple object tracking combining color and IR image features. We show the tracking results in frames 44, 52, 59, 67, 75 and 77 (from left to right and top to bottom) of the color and IR

sequences, respectively. Frames with the same number in the color and IR sequence are placed side by side

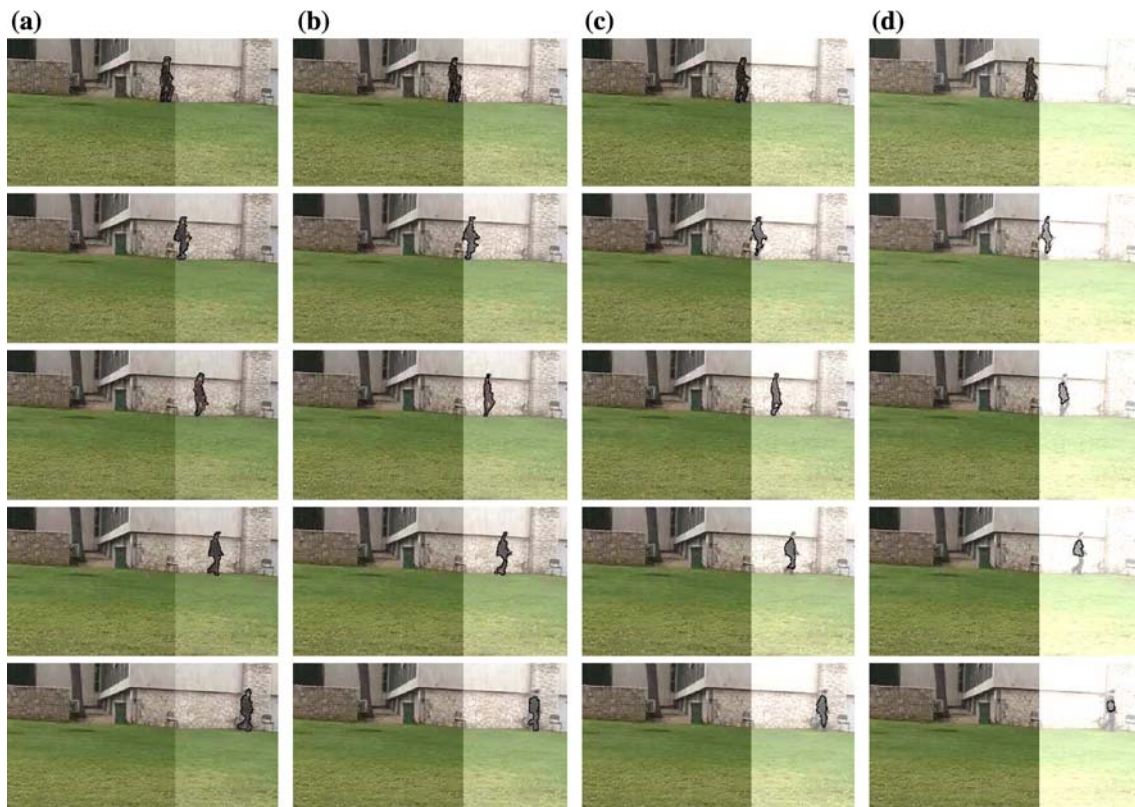
disappear and the object be lost forever. This condition is not always satisfied, e.g. when the movement of the object is very fast. To augment the probability of good contour initialization, we can directly use the level set function  $\Phi'$  which gives the best prediction of the current location of the object. Figure 17 shows the tracking result for the ball in the “tennis” video sequence, using the previous tracking result contour or the predicted one as initialization. The ball (modeled by a mixture of one component) was successfully tracked using the prediction (see row (a)); whereas the object contour disappeared when only the previous tracking result was used as initialization (see row (b)). We should note, however, that using the contour  $\Phi'$  guarantees that the initial contour contains a part of the object only if the latter moves with a relatively constant speed. If the object accelerates suddenly between two frames, the object position may not be efficiently predicted by using the previous motion speed.

## 5 Summary and discussion

This paper presents an effective method for combining region, boundary and shape information to yield a robust

tracking under severe conditions like cluttered backgrounds, noisy images and object occlusions. While the region and boundary are extracted statistically in the images, the shape information is calculated adaptively to the dynamic of each moving object. The experimental results demonstrated good tracking performance despite the presence of occlusions and the possibility of contour distractions. However, there are still issues and challenges that have to be overcome in the future in order to make the method applicable to real-time tracking.

For the tracking examples presented, the average running time is 2 frames/s with an Intel Pentium4 2.5 GHz processor and 512 MB RAM. Of course, this time does not include the computation of the mixture models and the texture and boundary features, which are calculated off-line. For the calculation of the mixture models, for example, we used the EM algorithm [27], where the time to convergence depends on the number of mixture components, the dimensionality of the data and the likelihood function [26,27]. Note also that the computation time increases with the resolution of the video, since more iterations are required for the level set functions to converge to the object boundaries. In future work, we can improve the computation time of the method in two ways. First, using multi-resolution analysis, we can



**Fig. 15** Illustration of tracking under illumination change: columns **a–d** show the tracking result when the Gamma correction parameter  $\nu$  is set to 0.67, 0.40, 0.20 and 0.10, respectively. In each column, from *top to bottom*, we show frames 33, 36, 39, 42 and 48

speed up the evolution of the object contours and process high-resolution videos in less time. Further, using fast numerical schemes for resolving the level set in Eq. (11) (see [7,33] and the references given there) can significantly enhance the speed of the algorithm. Finally, multiple-object tracking can be implemented in parallel programming with a special processing step to handle object occlusions.

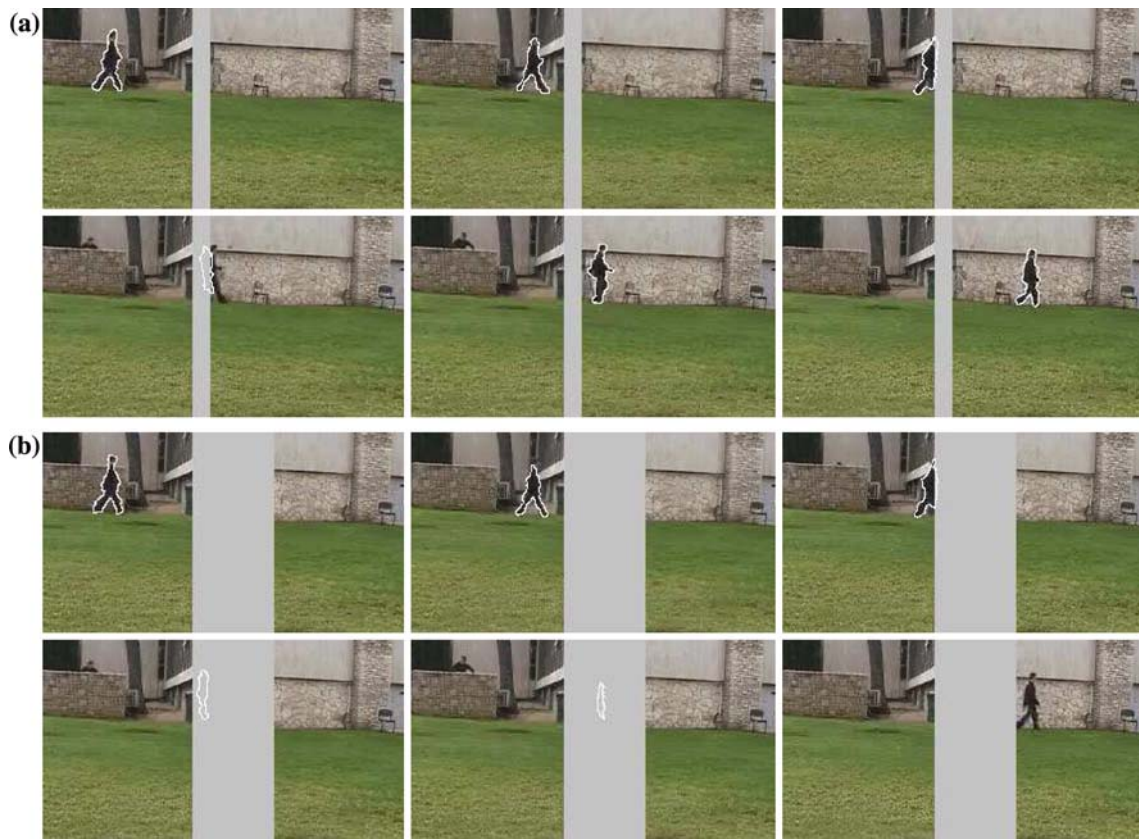
Note also that for all the examples shown in the paper, segmentation of the objects is performed manually in the first frame (frame 0). This segmentation could be performed automatically, by combining detection and tracking, for example. However, that would require that the camera be sufficiently stable for the initial frames of the sequence. Another way to perform automatic initialization is to use motion calculation and segmentation in order to separate different moving objects, then fit a mixture model for their distribution of color and texture. However, different parts of the same object may move differently, causing object over-segmentation. Combining detection and motion would seem to be promising way to perform automatic initialization: for instance, one can estimate the motion of the camera and use a motion-compensated detection (e.g., inter-frame difference) for segmenting the initial objects.

Finally, features derived from texture Gabor filters, edge orientation or motion could be used in addition to the set of features used in this paper in order to enhance the representation of the objects. In our experiments, for example, we showed an example in Fig. 14 for tracking by fusing region and boundary information from color and IR image sequences. We believe that more work is needed in the future to fuse more cues from the two sensors (e.g., wavelets, edge orientation), and apply the method for tracking specific objects such as faces, pedestrians and traffic vehicles.

**Acknowledgments** The completion of this research was made possible thanks to the Natural Sciences and Engineering Council of Canada (NSERC) and Bell Canada's support through its Bell University Laboratories R&D programs. We thank the reviewers for their helpful comments.

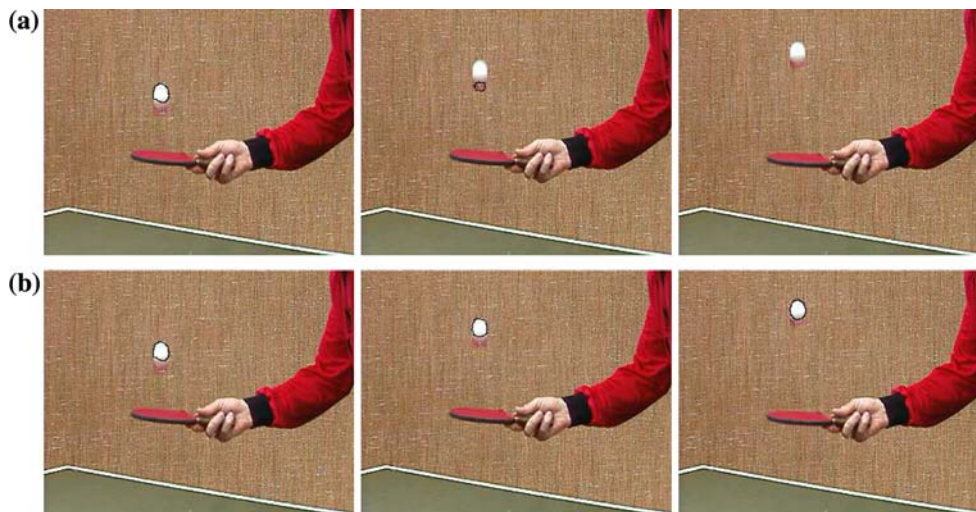
## 6 Appendix

In what follows, we give the formulas used for calculating the correlogram features. Consider a correlogram matrix calculated using a displacement  $\nu$  and orientation  $\phi$ . The *energy* (EN), *entropy* (ET), *inverse-difference-moment* (IDM) and



**Fig. 16** Examples of tracking with two synthetic occlusions: **a** shows an example of tracking success with a small synthetic occlusion, **b** shows an example of tracking failure when the occlusion is larger. The frames

shown in **a** and **b**, from left to right and from top to bottom, are 16, 22, 25, 28, 30 and 38



**Fig. 17** Example of tracking using as initialization: **a** the contour from the last tracking result, **b** the predicted contour

correlation( $C$ ) of the matrix are calculated as follows:

$$EN(v, \phi) = \sum_{c_i, c_j} (C^{v, \phi}(c_i; c_j))^2 \quad (18)$$

$$ET(v, \phi) = \sum_{c_i, c_j} -C^{v, \phi}(c_i; c_j) \log(C^{v, \phi}(c_i; c_j)) \quad (19)$$

$$IDM(v, \phi) = \sum_{c_i, c_j} \frac{1}{(1 + \|c_i - c_j\|^2)} C^{v, \phi}(c_i; c_j) \quad (20)$$

$$C(v, \phi) = \sum_{c_i, c_j} \frac{(c_i - M_x)(c_j - M_y)^T}{|\Sigma_x||\Sigma_y|} C^{v, \phi}(c_i; c_j) \quad (21)$$

where we have

$$M_x(v, \phi) = \sum_{c_i} c_i \sum_{c_j} C^{v, \phi}(c_i; c_j)$$

$$M_y(v, \phi) = \sum_{c_j} c_j \sum_{c_i} C^{v, \phi}(c_i; c_j)$$

$$\Sigma_x(v, \phi) = \sum_{c_i} (c_i - M_x)^T (c_i - M_x) \sum_{c_j} C^{v, \phi}(c_i; c_j)$$

$$\Sigma_y(v, \phi) = \sum_{c_j} (c_j - M_y)^T (c_j - M_y) \sum_{c_i} C^{v, \phi}(c_i; c_j)$$

and  $|\Sigma_x|$  (resp.  $|\Sigma_y|$ ) denotes the determinant of the matrix  $\Sigma_x$  (resp.  $\Sigma_y$ ). It is important to understand the meaning of each of the above texture features. First, EN describes the uniformity of a texture. In a homogeneous image, there are very few dominant color-tone transitions. Hence, the correlogram of such image will have fewer entries of large magnitude and, therefore, a high energy value. The second feature, ET, measures the randomness of the elements of the correlogram. A homogeneous texture will have a low entropy and vice versa. The IDM has a high relatively value when the high values of the correlogram are concentrated near the diagonal (because the term  $\|c_i - c_j\|^2$  is smaller in that part of the matrix). It follows that IDM is high when there are small regions in the texture with the same color, which could be taken as a measure of the coarseness of the texture. Finally,  $C$  measures the correlation between the different elements of the correlogram. When  $C$  is high the texture is more complex than when  $C$  is low.

## References

- Barron, J., Fleet, D., Beauchemin, S.: performance of optical flow techniques. *Int. J. of Comput. Vis.* **12**(1), 43–77 (1994)
- Mansouri, A.R., Konrad, J.: Multiple motion segmentation with level sets. *IEEE Trans. Image Process.* **12**(2), 201–220 (2003)
- Stiller, C., Konrad, J.: Estimating motion in image sequences: a tutorial on modelling and computation of 2d motion. *IEEE Signal Process. Mag.* **16**(4), 70–91 (1999)
- Adiv, G.: Determining 3D motion and structure from optical flow generated by several moving objects. *IEEE Trans. Pattern Anal. Mach. Intell.* **7**(4), 384–401 (1985)
- Blake, A., Isard, M.: *Active Contours*. Springer, Heidelberg (1998)
- Chan, T., Vese, L.A.: Active contours without edges. *IEEE Trans. Image Process.* **10**(2), 266–277 (2001)
- Goldenberg, R., Kimmel, R., Rivlin, E., Rudzsky, M.: Fast geodesic active contours. *IEEE Trans. Image Process.* **10**(10), 1467–1475 (2001)
- Cavallaro, A., Steiger, O., Ebrahimi, T.: Tracking video objects in cluttered background. *IEEE Trans. Circuits Syst. Video Technol.* **15**(4), 575–584 (2005)
- Collins, R.T., Liu, Y., Leordeanu, M.: Online selection of discriminative tracking features. *IEEE Trans. Pattern Anal. Mach. Intell.* **27**(10), 1631–1643 (2005)
- Comaniciu, D., Ramesh, V., Meer, P.: Kernel-based object tracking. *IEEE Trans. Pattern Anal. Mach. Intell.* **25**(5), 564–577 (2003)
- Wren, C.R., Azarbayejani, A., Darrell, T., Pentland, A.P.: Pfunder: real-time tracking of the human body. *IEEE Trans. Pattern Anal. Mach. Intell.* **19**(7), 780–785 (1997)
- Delamarre, Q., Faugeras, O.: 3D articulated models and multi-view tracking with physical forces. *Comput. Vis. Image Understand.* **81**(3), 328–357 (2001)
- Pänkers, R., Fua, P.: Tracking and modeling people in video sequences. *Comput. Vis. Image Understand.* **81**(3), 285–302 (2001)
- Mansouri, A.R.: Region tracking via level set pdes without motion computation. *IEEE Trans. Pattern Anal. Mach. Intell.* **24**(7), 947–961 (2002)
- Paragios, N., Deriche, R.: Geodesic active contours and level sets for the detection and tracking of moving objects. *IEEE Trans. Pattern Anal. Mach. Intell.* **22**(3), 266–280 (2000)
- Paragios, N., Deriche, R.: Geodesic active regions and level set methods for motion estimation and tracking. *Comput. Vis. Image Understand.* **97**(3), 259–282 (2004)
- Unal, G., Krim, H., Yezzi, A.: Fast incorporation of optical flow into active polygons. *IEEE Trans. Image Process.* **14**(6), 745–759 (2005)
- Osher, S., Sethian, J.: Fronts propagating with curvature-dependant speed: algorithms based on Hamilton-Jacobi formulations. *J. Comput. Phys.* **79**(1), 12–49 (1988)
- Jehan-Besson, S., Barlaud, M., Aubert, G.: Detection and tracking of moving objects using a new level set based method. *Proceedings of IEEE International Conference on Pattern Recognition, Barcelona, Spain, 3–8 September, 2000*, pp. 7112–7117
- Jehan-Besson, S., Barlaud, M., Aubert, G.: DREAM2S: deformable regions driven by an eulerian accurate minimization method for image and video segmentation. *Int. J. Comput. Vis.* **53**(1), 45–70 (2003)
- Cremers, D.: Dynamical statistical shape priors for level set-based tracking. *IEEE Trans. Pattern Anal. Mach. Intell.* **28**(8), 1262–1273 (2006)
- Zhang, T., Freedman, D.: Improving performance of distribution tracking through background mismatch. *IEEE Trans. Pattern Anal. Mach. Intell.* **27**(2), 282–287 (2005)
- Huang, J., Kumar, S.R., Mitra, M., Zhu, W.-J., Zabih, R.: Spatial color indexing and applications. *Int. J. Comput. Vis.* **35**(3), 245–268 (1999)
- Allili, M.S., Ziou, D.: Automatic color-texture image segmentation by using active contours. In: *Proceedings of 1st IEEE International Workshop on Intelligent Computing in Pattern Analysis/Synthesis, Xi'an, China, 26–27, August 2006*, LNCS 4153, pp. 495–504
- Tuceryan, M., Jain, A.: Texture analysis. In: Chen, C.H., Pau, L.F., Wang, P.S.P. (eds) *The Handbook of Pattern Recognition and Computer Vision*, 2nd edn. chap. 2.1, 207–248. World Scientific Publishing, Signapore (1998)
- Duda, R.O., Hart, P.E., Stork, D.G.: *Pattern Classification*. Wiley, New York (2001)
- McLachlan, G., Peel, D.: *Finite Mixture Models*, Wiley Series in Probability and Statistics (2000)
- Drewniok, C.: Multispectral edge detection: some experiments on data from landsat-TM. *Int. J. Remote Sens.* **15**(18), 3743–3765 (1994)
- Caselles, V., Kimmel, R., Sapiro, G.: Geodesic active contours. *Int. J. Comput. Vis.* **22**(1), 61–79 (1997)
- Cremers, D., Soatto, S.: A pseudo-distance for shape priors in level set segmentation. In: *Proceedings of 2nd IEEE Workshop on Variational, Geometric and Level Set Methods in Computer Vision, Nice, France, 13–16 October 2003*, 169–176



31. Adalsteinsson, D., Sethian, J.: A fast level set method for propagating surfaces. *J. Comput. Phys.* **118**(2), 269–277 (1995)
32. Erdem, Ç.E., Sankur, B., Tekalp, A.M.: Performance measures for video object segmentation and tracking. *IEEE Trans. Image Process.* **13**(7), 937–951 (2004)
33. Weickert, J., Kühne, G.: Fast methods for implicit active contour models. In: Osher, S., Paragios, N.: (eds) *Geometric Level Set Methods in Imaging, Vision and Graphics*, chap. 3, pp. 44–57. Springer, Heidelberg