Available online at www.sciencedirect.com
ScienceDirect
Neurocomputing 71 (2008) 776–786
NEUROCOMPUTING
www.elsevier.com/locate/neucom

Parametric improvement of lateral interaction in accumulative computation in motion-based segmentation

Javier Martínez-Cantos[a,*], Enrique Carmona[a], Antonio Fernández-Caballero[b], María T. López[b]

[a]Departamento de Inteligencia Artificial, E.T.S.I. Informática, UNED, 28040-Madrid, Spain
[b]Departamento de Sistemas Informáticos, Escuela Politécnica Superior de Albacete & Instituto de Investigación en Informática de Albacete (I3A), Universidad de Castilla-La Mancha, 02071-Albacete, Spain

Available online 19 November 2007

Abstract

Segmentation of moving objects is an essential component of any vision system. However, its accomplishment is hard due to some challenges such as the occlusion treatment or the detection of objects with deformable appearance. In this paper an artificial neuronal network approach for moving object segmentation, called lateral interaction in accumulative computation (LIAC), which uses accumulative computation and recurrent lateral interaction is revisited. Although the results reported for this approach so far may be considered relevant, the problems faced each time (environment, objects of interest, etc.) make that the system outcome varies. Hence, our aim is to improve segmentation provided by LIAC in a double sense: by removing the detected objects not matching some size or compactness constraints, and by learning suitable parameters that improve the segmentation behavior through a genetic algorithm.

Keywords: Parameter optimization; Genetic algorithm; Neural network; Motion segmentation

1. Introduction

Segmentation is a common low-level component in every visual surveillance system [21,22] which tries to establish the image regions in which the belonging pixels share certain characteristics (generally related with motion [11,23]). As a result, the foreground and the background are separated in each frame, i.e. the objects of interest are highlighted over the background.

The main problems that the segmentation suffers are the occurrence of false positives or foreground noise (this is, classification of background elements or defects of capture into objects of interest), and false negatives or background noise (misclassification of an object of interest as background). In addition, when one moving object overlaps another—partially or totally—a motion ambiguity can be produced, confusing so the regions involved. This problem is called occlusion. Nevertheless, this issue affects more the tracking than the segmentation. The tracking is a higher level module [28] responsible for storing the trajectory of the objects of interest detected during the sequence. We must keep in mind that motion segmentation is different from object motion-based segmentation [15]. In this work, we face the latter problem but not the former.

The resulting regions of the segmentation process are called blobs. For the sake of simplicity, we suppose that each blob obtained corresponds to only one moving object, though we are aware that this is not always true [41]. Indeed, a single blob can correspond to an entire object, to a part of a single object, to multiple objects, or even to a false positive (i.e. a shadow or a reflection), depending on the success in segmentation. In order to increase the reliability of segmentation, some systems have been proposed so far, as, for instance, in Ref. [3], where blobs are labeled as real moving object, shadow, ghost, reflection, fluctuation, or background noise.

The segmentation outcome can be bad if the objects of interest: (a) are too close or too distant to the camera, (b) are too many, (c) occlude themselves frequently, or,

*Corresponding author. Tel.: +34 967599200; fax: +34 967599224.
E-mail address: javier.mcantos@uclm.es (J. Martínez-Cantos).

0925-2312/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2007.10.007

(d) change its motion direction and/or velocity constantly. In the literature, segmentation methods are mainly grouped into three approaches, based on: (a) frame difference [20,30], (b) background subtraction [19,38], and, (c) optical flow [37,40]. In some cases, there are developments based on a combination of the previous ones, resulting in hybrid methods [6,7]. In this paper, our study is focused on a segmentation method based on lateral interaction in accumulative calculation (LIAC) [8–10] that can be generically included in the image difference methods. LIAC is a multi-layer artificial neuronal network (ANN) inspired in two models: local accumulative computation [12] and recurrent lateral interaction [27].

Segmentation algorithms usually contain a set of configuration parameters that have to be adapted to each concrete situation. Traditionally, an expert was in charge of configuring these parameters; but this practice is neither effective nor efficient, and sometimes it turns even impossible. Thus, several algorithms have been developed so far to adjust automatically these parameters. For instance, a parameter estimator based on the relationship among image features has been introduced [5]. Also, a set of subsystems has been proposed for auto-critical evaluation, for auto-regulation of the parameters, and for error recovery to provide a reduction in the sensitivity to environmental changes [16]. Moreover, another example for marks-based motion capture has been provided [33].

Since we can assess the fitness of each set of parameters—at least, in an intuitive manner—we recognize an optimization problem. A genetic algorithm (GA) fits perfectly in this context due to its capacity to achieve good results in wide solution spaces. During the last few years, the GA has been used in many studies on problems related to computer vision [2,4]. For example, a GA in which the chromosomes (codification of parameters of interest) are initialized using the results of the previous frame in a video sequence, instead of using random values is presented in Ref. [17]. This way, the unstable chromosomes correspond to motion objects that are evolved by crossover (genetic exchange) and mutation (alteration at random). The application of GA is not trivial and that is why there are specific designs depending on the problem. In Ref. [1], a distributed GA (DGA) is used to optimize a large set of 30 parameters, while in Ref. [18] another DGA is used to extract objects and to track them, without using any a priori knowledge. In Ref. [13], a new model of GA is presented, namely the distributed hierarchical GA: a hybrid technique that partitions the search space into sub-spaces. Recently, evolutionary algorithms are combined with multi-agent systems, generating a group of "segmentation agents" and a single "coordination agent" [26]. In this approach, the whole systems acts as a GA population. In close relation to our approach, other evolutionary algorithms have been used to obtain optimal parameters, like in Ref. [35] where the authors assure that an ant colony optimization algorithm improves the search performance and requires significantly reduced computations, compared with the GA.

The relation between segmentation algorithms and ANNs also occupy an important place. In this case, the problem of parameter optimization is again present, now in form of weight setting, state configuration, and so on. In Ref. [32] several methods for automated parameter estimation are compared. The paper concludes that the stochastic optimization methods usually overcome the deterministic ones. In addition, hybrid methods are the most promising ones. In this sense, many GAs have been applied to ANNs. For instance, a work is presented to both optimize parameters and to assist the network design [14]. Another example introduces the combination of a GA with a neuronal network, to form a classifier decision tree with classification rules for similarity, where the GA is used to tune the parameters of the neuronal network [39]. However, this association sometimes is inverted, using the neuronal network to improve the GA performance. This is the case, for example, of a work where GA carries out object tracking, while the neuronal network increases its effectiveness for multiple target tracking [36].

In fact, LIAC may be affected deeply by the background conditions, the high variety of objects of interest, etc. The adaptation to all these circumstances depends on correct system parameters tuning (Fig. 1 shows some very different system outputs depending on the parametric configuration). Such a job is not automatic and requires an expert to perform the adjustment, interpreting the scene a priori and handling the system to detect exactly the objects of interest.

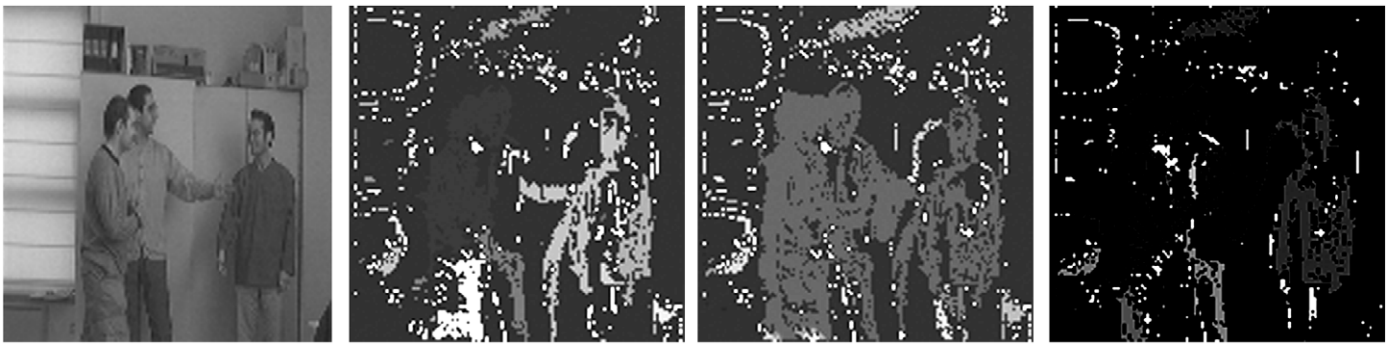

Fig. 1. LIAC outcome for a frame using three different parametric configurations.

In this paper, an approach that tries to achieve the auto-configuration of LIAC is presented. We introduce two modules: the first one aiming to improve the overall output by using new parameters that filter the system outcome and the second one seeking to provide a feedback to the LIAC system to learn more suitable parameters by means of a GA.

Since the tracking systems are usually robust enough to support the casual loss of an object in the trace, we will try to avoid the maximum amount of noise, preserving the blobs corresponding to objects of interest, even though an object could be lost in some frames.

The success degree achieved will be measured in a quantitative manner, mainly by means of an error function. We assume the hypothesis of obtaining one blob per moving object for each frame, but it is difficult to say when an object is stopped completely (total absence of motion). Therefore, we pursue to satisfy the user that uses the surveillance system.

2. LIAC in image segmentation from motion

The complete LIAC process is distributed along four layers. In this section, the role of each of these four layers is explained.

2.1. Layer 0: Segmentation by grey level bands

This layer covers the need to segment the image at a predefined group of n grey level bands. Each element (x, y) is capable of processing motion from input grey level value $IN(x, y, t)$ and its proper charge value. Let $GLS_k(x, y, t)$ be the presence or absence of grey level k at element (x, y) at time t.

$$GLS_k(x,y,t) = \begin{cases} -1, & \text{if } IN(x,y,t) \neq k, \forall k \in [0,255] \\ 1, & \text{otherwise} \end{cases},$$

where k is a particular grey level band. In other words, we have to determine in what grey level band a certain pixel falls. So, we are not evaluating, at this level, if there is motion in a grey level band for a given pixel. This task is left to the following layer.

It must be clear that one, and only one, of the outputs of all the detecting modules of the grey level bands can be activated at a given instant. This fact, although obvious, is of great interest at the higher layers of the architecture, since it will avoid possible conflicts among the values offered by the different grey level bands. Indeed, only one grey level band will contain valid values.

2.2. Layer 1: Lateral interaction for accumulative computation

This layer has been designed to obtain the permanence value $PM_k(x, y, t)$ on a decomposition on a grey level band basis. We will have n sub-layers and each one of them will memorize the value of the accumulative computation present at global time scale t for each element. Lateral interaction in this layer is thought to reactivate the permanence charge of those elements partially loaded and that are directly or indirectly connected to maximally charged elements. The permanence charge of each element will be offered to the following layer as output.

Firstly, at global time scale t, permanence memory charge or discharge due to motion detection is performed. This information, given as input from layer 0, is associated to sub-layer k of layer 1 (grey level band k). The accumulative computation equation may be formulated as

$$PM_k(x,y,t) = \begin{cases} l_{\text{dis}}, & \text{if } GLS_k(x,y,t) = -1 \\ l_{\text{sat}}, & \text{if } (GLS_k(x,y,t) = 1) \\ & \quad \wedge (GLS_k(x,y,t-\Delta t) = -1) \\ \max[PM_k(x,y,t-\Delta t) & \text{if} (GLS_k(x,y,t) = 1) \\ \quad -dv, l_{\text{dis}}], & \quad \wedge (GLS_k(x,y,t-\Delta t) = 1) \end{cases}$$

where l_{dis} is the discharge or minimum permanence value, l_{sat} is the saturation or maximum permanence value, and dv is the discharge value due to motion detection.

Notice that Δt determines the sequence frame rate and is given by the capacity of the model's implementation to process one input image. At each element (x, y) we are in front of three possibilities: (1) the sub-layer does not correspond to the grey level band of the image pixel. The permanence value is discharged down to value l_{dis}; (2) the sub-layer corresponds to the grey level band of the image pixel at time instant t, and it did not correspond to the grey level band at the previous instant $t-\Delta t$. The permanence value is loaded to the maximum of saturation l_{sat}; (3) the sub-layer corresponds to the grey level band of the image pixel at time instant t, and it also corresponded to the grey level band at the instant $t-\Delta t$. The permanence value is discharged by a value dv (discharge value due to motion detection); of course, the permanence value cannot get off a minimum value l_{dis}.

The discharge of a pixel by a quantity of dv is the way to stop maintaining attention to a pixel of the image which had captured our interest in the past. As it will be seen later on, if a pixel is not directly or indirectly bound by means of lateral interaction mechanism to a maximally charged pixel (l_{sat}), it goes down to the total discharge with time.

Secondly, an extra charge rv—recharge value due to neighboring—is added to the permanence memory in those image pixels that receive a stimulus from maximally charged element almost l_1 pixels far away in any of four directions. This recharge can only happen one time, and provided that none neighbor element up to the maximally charged element is discharged. l_1 is called number of neighbors in accumulative computation. This recharge mechanism allows maintaining attention on those pixels directly or indirectly connected to maximally charged

pixels. This mechanism is even able to reinforce the permanence memory value if $rv > dv$.

$$PM_k(x,y,T) = \min[PM_k(x,y,T-\Delta T) + \varepsilon rv, l_{\text{sat}}]$$

where

$$\varepsilon = \begin{cases} 1, & \text{if } \exists(1 \leqslant i \leqslant l_1)\forall|(1 \leqslant j \leqslant i) \\ & ((PM_k(x+i,y,T-\Delta T) = l_{\text{sat}}) \\ & \wedge(PM_k(x+j,y,T-\Delta T) \neq l_{\text{dis}}))\vee \\ & ((PM_k(x-i,y,T-\Delta T) = l_{\text{sat}}) \\ & \wedge(PM_k(x-j,y,T-\Delta T) \neq l_{\text{dis}}))\vee \\ & ((PM_k(x,y,+i,T-\Delta T) = l_{\text{sat}}) \\ & \wedge(PM_k(x,y+j,T-\Delta T) \neq l_{\text{dis}}))\vee \\ & ((PM_k(x,y,-i,T-\Delta T) = l_{\text{sat}}) \\ & \wedge(PM_k(x,y,-j,T-\Delta T) \neq l_{\text{dis}})) \\ 0, & \text{otherwise} \end{cases}$$

Finally, back at global time scale t, the permanence value at each pixel (x, y) is threshold by means of a permanence value threshold (θ_1) and sent to the next layer.

In order to explain the central idea of this layer 1, we will say that the activation toward the lateral modular structures (up, down, right, and left) is based on the following basic ideas: (1) all modular structures with maximum permanence value l_{sat} (saturated) output the charge toward the neighbors; (2) all modular structures with a no saturated charge value, and that have been activated from some neighbor, allow to pass this information through them (they behave as transparent structures to the charge passing); and (3) the modular structures with minimum permanence value l_{dis} (discharged) stop the passing of the charge information toward the neighbor (they behave as opaque structures). Therefore, we are in front of an explosion of lateral activation beginning at the structures with permanence memory set at l_{sat}, and that spreads lineally toward all directions, until a structure appears in the pathway with a discharged permanence memory.

2.3. Layer 2: Lateral interaction for charge redistribution by grey level bands

Layer 2 is also made up of n sub-layers, where, by means of lateral interaction, charge redistribution among all the connected neighbors in a surrounding window of $l_2 \times l_2$ pixels that hold a minimum charge is performed. Besides, distributing the charge $C_k(x, y, t)$ in grey level bands at this level the charge due to the motion of the background is also diluted. The new charge obtained in this layer is offered as an output toward layer 3.

Starting from the values of the permanence memory in each pixel on a grey level band basis, we will see how it is possible to obtain all the parts of an object in movement. A part of an object concretely means the union of pixels that are together and in the same grey level band. The discrimination of each one of the parts that compose the objects is equally obtained by a lateral co-operation mechanism. In case of layer 2, the charge will be homogenized among all the pixels that pertain to the same grey level band and that are directly or indirectly united to each other. This way, a double objective will be obtained. (1) To dilute the charge due to the false image background motion along the other pixels of the background. This way, there should be no presence of motion characteristic of the background, but we will rather keep motion of the objects present in the scene. (2) To obtain a parameter common to all the pixels in the part of the object in a surrounding window of $l_2 \times l_2$ pixels with a same grey level band.

Initially, at global time scale t, the charge value at every pixel (x, y) and at every sub-layer k is given by the value of the permanence from the previous layer.

Afterwards, at local time scale T, provided that the neighbor input charge values are high enough, the center element (x, y) calculates the mean of its value and the partially charged neighbors in a surrounding window of $l_2 \times l_2$ pixels. l_2 is denominated number of neighbors in charge redistribution

$$C_k(x,y,T) = \frac{C_k(x,y,T-\Delta T) + \sum_{i=-l_2}^{l_2}\sum_{j=-l_2}^{l_2}\delta_{x+i,y+j}C_k(x+i,y+j,T-\Delta T)}{1 + \sum_{i=-l_2}^{l_2}\sum_{j=-l_2}^{l_2}\delta_{x+i,y+j}}, \forall(i,j) \neq (0,0)$$

where

$$\delta_{\alpha,\beta} = \begin{cases} 1, & \text{if } C_k(\alpha,\beta,T-\Delta T) > l_{\text{dis}} \\ 0, & \text{otherwise} \end{cases}$$

Again at global time scale t, the charge value at each pixel (x, y) is threshold by means of a charge value threshold (θ_2) and sent to the next layer.

2.4. Layer 3: Lateral interaction for spot fusion

In every element of layer 3, we have an input from each corresponding element of the n sub-layers of layer 2. This layer has as purpose the fusion into uniform spots of the objects in a surrounding window of $l_3 \times l_3$ pixels. That is why it takes the input charges of each one of the grey level bands and performs a fusion of these values, obtaining uniform parts of all the moving objects of the original image. Its output is a set of spots $S(x, y, t)$.

Up to now attention has been captured on any moving objects in the scene by means of co-operative calculation

mechanisms in all the grey level bands. Motion due to background has also been eliminated. It is now necessary to fix as a new objective to distinguish clearly the motion of the different objects. This discrimination is also obtained by lateral co-operation mechanisms. Again we will connect the modular structures of this layer in a mesh form in layer 3. Nevertheless, from now on we will no longer work with sub-layers, but rather with a single layer in which all the information of the n sub-layers of layer 2 ends up. In layer 3, we will homogenize the charge values among all the pixels that contain some charge value superior to a minimum threshold and that are physically connected to each other in a radius of l_3 pixels.

Firstly, the spot charge value at each pixel (x, y) is given by the charge value of the maximally charged sub-layer k from the previous layer.

$$S(x, y, T) = \max[C_k(x, y, T)],\ \forall k \in [0, 255]$$

At local time scale, provided that the neighbor input charge values are high enough, the center element (x, y) calculates the mean of its value and the partially charged neighbors in a surrounding window of $l_3 \times l_3$ pixels. l_3 is denominated number of neighbors in object fusion.

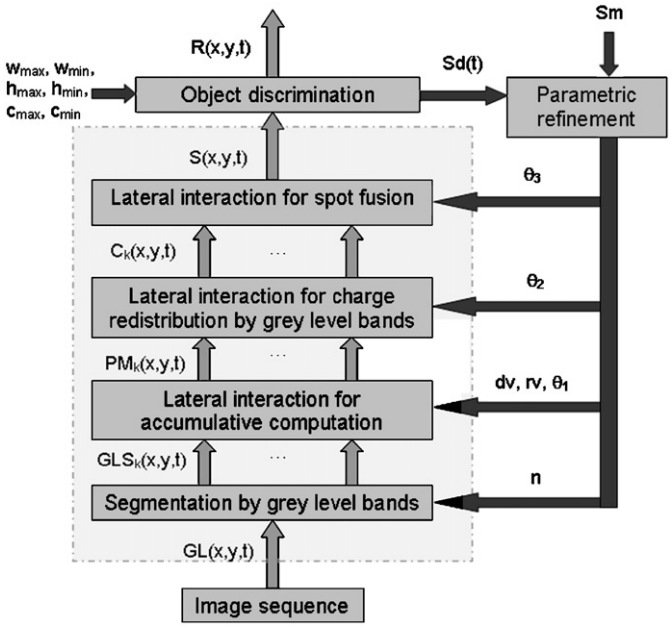

Fig. 2. Framework proposed including by the LIAC system and two new modules for the filtering and parameter learning outcome.

$$S(x, y, T) = \frac{S(x, y, T - \Delta T) + \sum_{i=-l_3}^{l_3} \sum_{j=-l_3}^{l_3} \delta_{x+i,y+j} S(x + i, y + j, T - \Delta T)}{1 + \sum_{i=-l_3}^{l_3} \sum_{j=-l_3}^{l_3} \delta_{x+i,y+j}}, \forall (i, j) \neq (0, 0)$$

where

$$\delta_{\alpha,\beta} = \begin{cases} 1, & \text{if } S(\alpha, \beta, T - \Delta T) > l_{\text{dis}} \\ 0, & \text{otherwise} \end{cases}$$

Back to the global time scale t, the spot charge value at each pixel (x, y) is threshold by means of a shape value threshold (θ_3).

3. LIAC improvements

In this section, we present a framework that includes the previous LIAC approach and two new modules to improve the former parameters and outcome. Fig. 2 shows the layout of the new proposed segmentation system.

As shown in Fig. 1, LIAC in image segmentation from motion produces one set of blobs for each frame. The system enhancement is based on the management of those blobs; on one hand by avoiding the generation of noisy blobs, and on the other hand by removing those noisy blobs already generated. Briefly, the module called "object discrimination" filters those blobs according to the user criterion to obtain only the objects of interest for each image, while the module called "parametric refinement" handles the LIAC parameters depending on the number of detected objects in relation to the user's indications.

3.1. Object discrimination

The resulting blob set from segmentation is filtered in this module by means of size and compactness criteria, in terms of certain characteristics of the bounding box—of minimum size—that envelopes the object. Each object is accepted or rejected according to the adjustment to some spatial restrictions. On one hand, by describing the maximum width ($w_{\max}$), minimum width ($w_{\min}$), maximum height ($h_{\max}$), and minimum height ($h_{\min}$) allowed. On the other hand, by considering the area actually occupied by the object inside its bounding box; that is, the compactness of the object. This ratio is also defined between two values, namely, maximum compactness ($c_{\max}$) and minimum compactness ($c_{\min}$).

This module results in an outcome improvement due to the removing of noise and unwanted objects. Facing the next module, for each frame the number of detected objects ($Sd(t)$) is stored.

3.2. Parametric refinement

The LIAC structure is built as an ANN and hence it inhabits a learning system by itself. However, the learning capability has not been exploited yet. For that reason, we

intend to enhance the LIAC behavior through the modification of its parameters. It is clear that we are facing an optimization problem, and hence we will use a GA in order to obtain sets of parameters for a suitable segmentation. The segmentation success is measured considering the difference between the number of objects detected in each frame and the user's ideal number (indication of how many objects are moving during the scene). This rough measurement is enough and avoids having to specify more accurate knowledge.

In our genetic approach, the individuals involved have a representation based on the LIAC parameters. The chromosome configuration is performed by concatenating those parameters. Therefore, we consider the parameters presented in Section 2; however, we do not want to be limited by parameters related to the number of neighbors in charge redistribution. So, the parameters explained previously, i.e. number of neighbors in accumulative computation (l_1), number of neighbors in charge redistribution (l_2), and number of neighbors in object fusion (l_3) have been set to such a high fixed value that lateral interaction is not affected by them. Since each child is composed from chromosome pieces of his/her parents by means of the crossover operator, it is recommended to locate related parameters into consecutives genes. Following this advice, we put the charge value (dv) together with the discharge value (rv), and the thresholds (θ_1, θ_2, θ_3) aligned. Therefore, (n, dv, rv, θ_1, θ_2, θ_3) is a suitable chromosome where each gene is coded as an 8-bits unsigned integer.

Our GA uses a stack where every new individual of the population is put on. The selection of parents is performed by taking pairs from the bottom to the top (the population must always have even size), and every single couple is crossovered generating a single child whose genes can be affected by mutation. Then, the worst adapted parent of each couple (i.e. the parent with worst fitness) is removed from the population (like in a steady-state model), and hence it is extracted from the stack.

Since we are interested in evaluating the feasibility of the genetic approach proposed, we will run several executions trying to achieve: (a) fast but not premature convergence (gaining some quality at low computational cost), (b) enough selection pressure (forcing the improvement), (c) proportionally reward the individual's aptitude regarding the fitness of the whole population (the more suitable the individual, the more offspring it will have), and (d) guarantee of offspring for each individual (at least one chance for everyone). For the moment, the exploration should prevail over the exploitation, because further works will be able to study in depth the promising results presented in this paper.

The evaluation function designed for the GA (E) tries to minimize the error generated by the number of detected objects in each instant ($Sd(t)$) according the number of expected objects indicated by the user (Sm), in a sequence of k frames:

$$E = \sum_{t=0}^{k-1} \sin(\frac{\pi t}{k}) |Sd(t) - Sm|$$

So, this adaptation or fitness function considers in each frame the difference between the number of detected objects and the number of expected objects, providing more importance to the middle frames in the sequence. The reason of this weight is that in the sequences we use all the objects are not present or they are hardly detectable at the beginning and at the end of the sequence. Moreover, the first frames processed are useless, until the accumulative computation is stable, i.e. the convergence state is achieved.

4. Data and results

In order to show the performance enhancement due to our proposal, we carry out an analysis about the produced results, depending on how the modules "object discrimination" and "parametric refinement" are handled. We use an image sequence from the own LIAC proposal [10], composed of 100 frames with size 128×128 pixels in 256 grey levels. Figs. 3(a) and 4(a) show some frames. Initially, a human standing in the center of a room receives other two persons, each one entering through opposite sides. During the sequence, the people greet and interact with each other, while some occlusions happen and sporadically someone stops moving and then starts again. Finally, all the people leave the scene one to one to the left.

Indeed, LIAC applied to image segmentation offers acceptable results when the parameters are well configured, even though it cannot be assessed visually. A typical set of parameters that have shown good behavior in general [10] is: 8 as grey level bands (n), 63 as discharge value due to motion detection (dv), 31 as recharge value due to neighboring (rv), and 150 for each threshold, namely, permanence value threshold (θ_1), charge value threshold (θ_2), and shape value threshold (θ_3), respectively. According to the description offered in the previous section, the chromosome or candidate solution that summarizes the typical configuration of LIAC is (8, 63, 31, 150, 150, 150), called reference parameters henceforth.

However, for challenging sequences such as those with low resolution, containing various moving objects, some occlusions and so on—as it can be observed in Ref. [10] and in Fig. 3(b)—the obtained blobs are badly defined and mixed with a lot of noise. The selection of non-effective parameters produces more noise, perhaps invalidating the whole segmentation. Looking qualitatively into Fig. 3(b), a human watcher cannot notice that the sequence offers an average number of 312 objects detected per frame. This makes the information treatment harder and more expensive for higher level software layers using the LIAC as its segmentation engine.

In this scenario, we set $Sm = 3$ (three objects of interest in the scene, i.e. the three humans beings) in order to check

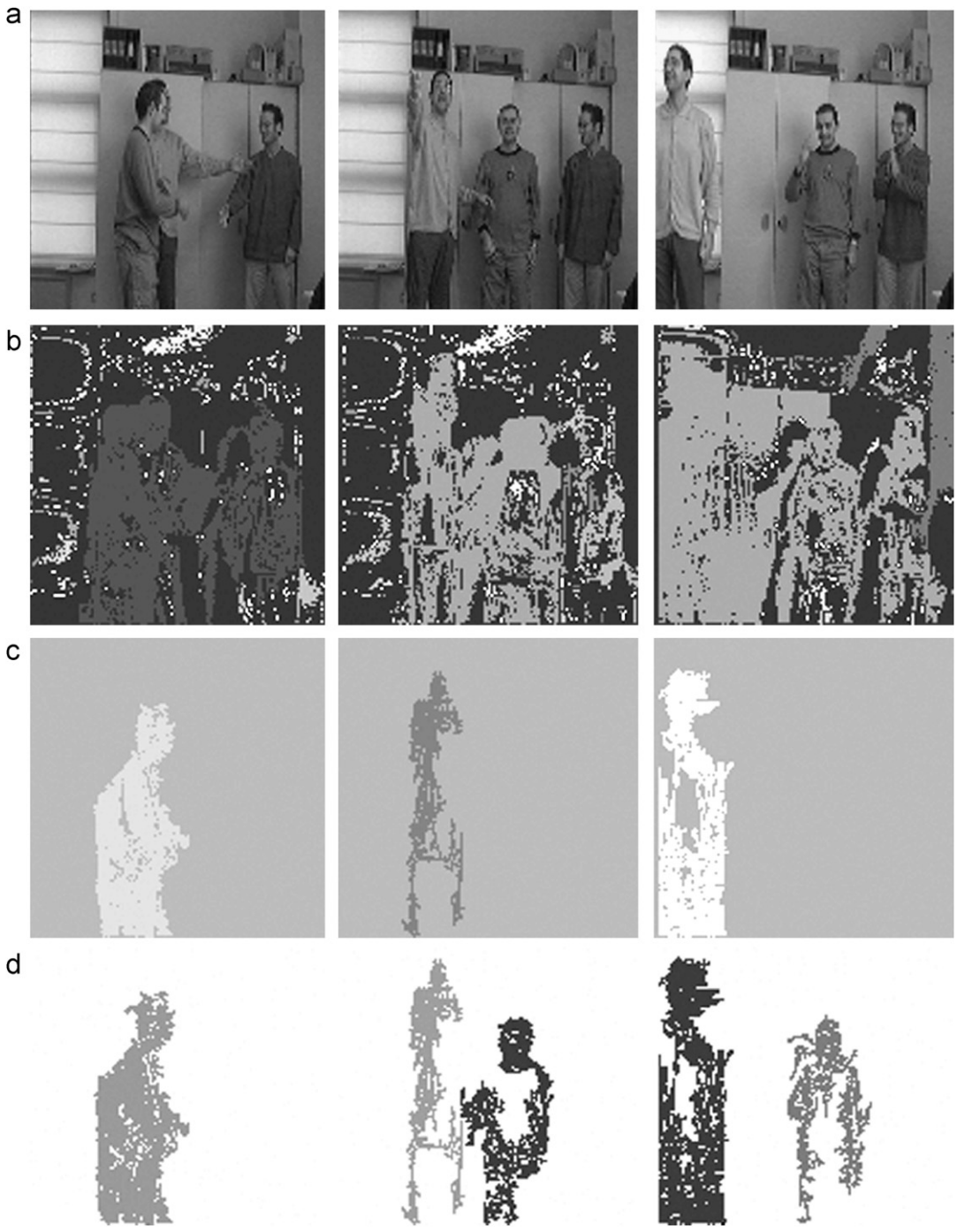

Fig. 3. Comparative: (a) original sequence, (b) segmentation outcome using reference parameters without object discrimination (fitness = 22.4×10^3), (c) (16, 96, 6, 121, 183, 50) with object discrimination (fitness = 156.62) and (d) (16, 235, 157, 76, 189, 0) with object discrimination (fitness = 153.79).

the fitness of the parameters obtained. It must be highlighted that *Sm* is an ideal value indicated by the user; it is quite imperfect, since $Sm = 3$ communicates to the system that there should be detected exactly three objects in each frame. Nonetheless, in this sequence it does not happen this way, because the objects of interest are not always present and even when they are present they are not moving all the time. Moreover, we must keep in mind that the first frames of the sequence always increment the error due to the convergence effect. To conclude, although the evaluation function is an error measurement, we observe that it is impossible to reach the value 0. Fig. 3(b) shows the typical outcome of LIAC using the reference parameters, obtaining a fitness rate as high as 22.4×10^3.

4.1. Results with object discrimination and without parametric refinement

Since we try to determine the feasibility of the proposed solution, we are interested in obtaining several acceptable

Fig. 4. "Ghost apparition": (a) original sequence and (b) some blobs obtained do not correspond with real objects in motion, but with shadows, reflections, and so on.

solutions instead of producing few very well-refined ones. This way, we choose to use an ample range of discrimination values, both in compactness (20% minimum and 95% maximum), and in size (height between 78 and 116 pixels, and width between 7 and 44 pixels). Thus, we would not force the GA to search for the best solutions, but rather we will try to find "clues of optimum solutions".

Using the reference parameters defined in the previous section and applying the mentioned discrimination values, we obtain a fitness of 182.15 (quite reduced in comparison with the one obtained without such a discrimination). So, the discrimination cleans a lot the segmentation, almost eliminating all the noise. Moreover, this module removes the transitory objects produced by the LIAC convergence effect. Nevertheless, objects of interest are also removed sometimes. At this point, we must warn that the fitness value is guidance towards improvement, that it can confuse us if it classifies "ghosts" as objects of interest (false positives). Frequently, the ghosts are present as shadows and reflections (Fig. 4).

4.2. Results with object discrimination and parametric refinement

In a previous paper [25], the effectiveness of this approach has been demonstrated on simple sequences: for a scene where a single human walks along a room, we get a perfect blob involving the whole object. Now, we face a bigger challenge because the current sequence is quite more complex due to: interactions between humans, discontinuous motion, occlusion presence, low-contrast clothes, and so on.

We decided to run the GA several times, about 20 executions, enough to get an idea about the improvement. The "object discrimination" module remains configured as mentioned in the previous section. Since it was already observed that a big population size is better than a smaller one, it has been chosen that there are 20 individuals whose chromosomes are generated at random. The stop condition has been set to a fixed number of generations, exactly 14, because the fitness improvement corresponds to a descendent exponential function and this number is enough to show the population's trend (achieving the exploration-exploitation frontier).

Moreover, since the executions are not very long, the crossover has been configured to 3 cut points so that there will be a greater heterogeneity (many jumps between different regions of the space of solution). As many authors agree, the mutation probability has been assigned to a low value, exactly 9% per gene. For efficiency reasons the values of number of grey level bands were restricted to 2, 4, 8, and 16, which are those of a better behavior, just as the LIAC authors do guarantee.

In Fig. 3(c) and (d), the outcome of some frames of difficult segmentation are represented using two chromosomes obtained through the GA (16, 96, 6, 121, 183, 50) and (16, 235, 157, 76, 189, 0). The fitness measurements demonstrate a substantial improvement: 156.62 and 153.79, respectively. A thorough analysis of every frame involved in the resulting sequences shows that the detection of "ghosts" is almost null.

This can be justified keeping in mind that the set of parameters is configured to values able to classify human motion, removing the sporadic appearance of reflections and shadows. The optical flow of human motion seems to be more homogeneous than the apparent motion due to ghosts. On the one hand, ghosts are usually morphologically unacceptable to the "object discrimination" module, and on the other hand, the parameters are set to detect blobs with certain evolution during few frames.

The chromosomes obtained after the executions provide an excellent way to understand the parameter configuration. A successful number of bands, n, is usually 8 or 16 (higher values). It always occurs that the discharge value due to motion is greater than the recharge value due to neighboring ($dv >$ or even $dv \gg rv$). The thresholds θ_1 and θ_3 always have values lower than θ_2: it is interesting to observe how the chromosome used to obtain the frames represented on Fig. 3(d), even disables θ_3. The results provided in Ref. [25] agree in this sense.

Fig. 5 compares, for each frame, the amount of: (a) objects of interest present (humans in this case), (b) moving objects (those with enough motion than a human can notice it), and (c) moving objects detected by the average of the 20 chromosomes obtained. It is worth noting that we distinguish among number of objects really present in the scene and number of moving objects. Both data have been gathered manually, and, hence, it is just an artificial approximation. In fact, we have decided in what moment an entering or exiting object starts or stops to count, respectively, and when an object is in motion and when it is not.

Since the average outcome provided by the chromosomes achieves certain similarity with the human observation, it is clear that we have obtained an important improvement. However, there are two adverse events.

On one hand, in Fig. 5 we can observe some peaks in the "moving objects" line. This fact represents a hard situation

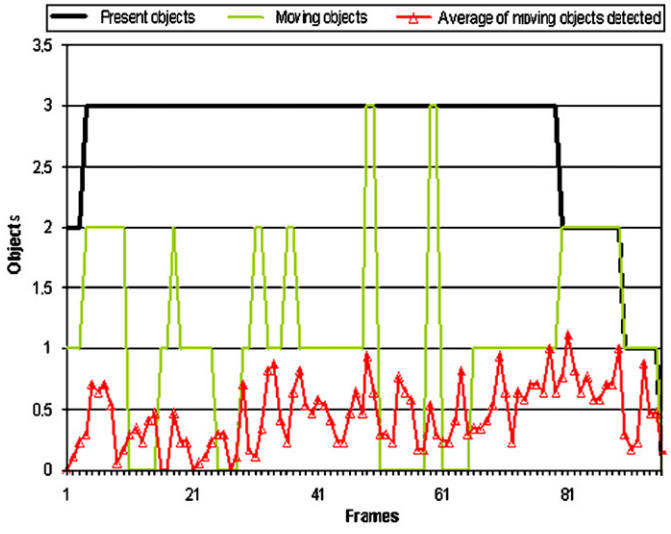

Fig. 5. Average of moving objects detected by the 20 chromosomes obtained, compared with moving objects perceived by a human observer.

in which several objects start and stop motion simultaneously. Peaks require a high discharge value while, generally, human motion requires slow discharges (indeed, it is more or less affected by the whole set of parameters). Therefore, maybe there are controversial cases that make the design of a unique parameter set for a large application difficult.

On other hand, in Fig. 5 we can notice delays between the observation of moving objects and the detection of that motion. This is due to the effect of the permanence memories of the accumulative computation, which are firstly charged and then discharged. Therefore, there is an implicit limitation in the response of the segmentation system that we could not overcome.

5. Conclusions and future work

In this paper the problem of motion segmentation has been presented, focusing on a particular neuronal network called lateral interaction in accumulative computation (LIAC). In comparison to other current approaches, the most significant contribution of the model is that it is capable of detecting all elements moving in an indefinite video image sequence including any kind of motion [9]. Moreover, LIAC does not require image pre-processing, reference images usage, or high-level knowledge injection. LIAC is a 2D approach to motion estimation, but it does not suffer the typical restrictions due to illumination, as it operates on regions instead of individual pixels. Since the method does not depend on the pattern of translation motion, it is not affected by the greatest disadvantage of region-based methods: the translation pattern is valid while the regions are remained quite small. In addition, the method provides charge values that could facilitate the object classification. On the other hand, the most important limitation of the LIAC is the impossibility to differentiate among objects that are seen as a whole due to occlusions [9,28].

Nowadays, the most common approach to detect motion in a sequence of images is background subtracting that is achieved by taking absolute differences between each incoming frame and a background model of the scene. It is supposed to provide the best compromise between performance and reliability, even though this is very sensitivity to illumination changes [29,34]. The development of adaptive background models is the answer for a suitable illumination treatment. However, this introduces new problems such as appearance of holes inside moving blobs [20]. In addition, there is an important trend in improving segmentation through: (a) reducing non-desired noise produced [24], (b) including some domain knowledge [2], (c) minimizing the errors at system deployment time [31], and so on. The segmentation methods that use color have become a recent focus of intensive research and for sure will to produce robust segmentation results [3].

Anyway, the weakness of segmentation systems is that they need their parameters to be tuned correctly. This task

is not trivial at all. Therefore, we proposed a software wrapper that automates the parameter optimization, composed by two modules. In other words, the improvement add-on performs a noise cleaning and a learning of suitable parameters.

The human supervision has been reduced successfully, because the user only needs to vaguely establish some morphological features of the objects of interest (size and compactness), as well as to indicate approximately how many moving objects are present in the sequence, instead of manually setting parameters and comparing their outcomes during many computationally expensive tests.

Both modules "object discrimination" and "parametric refinement" contribute to the improvement: the former by cleaning the noise of the LIAC outcome and the latter by learning the more suitable sets of parameters. As a result, the LIAC output has been greatly improved because the resulting images show in most of the cases only the objects of interest. In spite of the fact that the enhancement is limited by the segmentation method itself, as we have highlighted, the charge and discharge of pixels causes a delay in the detection.

Some additional experiments have pointed out that the number of objects of interest reported by the user does not need to be exactly the number of objects visible at the same time during the sequence. It is enough to report a finite number, equal to or greater than the real number of objects of interest. This is true, because it has been demonstrated that a trained LIAC easily classifies the visual objects in motion, removing the false positives (ghost apparition). Thus, false positives are more unlikely to be considered as objects of interest, each time one real object of interest is detected. Setting an implicit high number of objects constitutes an idea for future work.

This is only one step beyond and a lot of work may still be carried out. It is our intention to achieve a learning system with no initial knowledge at all about what is going to be monitorized, by making implicit the overall parameters that the user currently inputs. So, the final goal is that the whole proposed system does not need any input parameter. In further works, we will try to develop more powerful evaluation functions. In this way, we will study to do implicit the functionality of the "object discrimination" module, avoiding the presence of the user.

Lastly, we suggest detecting the moving objects as groups of—one or more—blobs, instead of forcing the relation one-to-one, because it is more difficult to split one blob into two objects than to compose one from its parts. Moreover, the higher layers of a surveillance system handle more easily the scene interpretation if they can retrieve the object parts.

Acknowledgments

This work has been partially supported by the Spanish Ministerio de Educación y Ciencia TIN2004-07661-C02-01, TIN2004-07661-C02-02, TIN2007-67586-C02-01, and TIN2007-67586-C02-02 grants, as well as by the Junta de Comunidades de Castilla-La Mancha PBI06-099 grant.

References

[1] A. Bevilacqua, Optimizing parameters of motion detection system by means of a distributed genetic algorithm, Image Vis. Comput. 23 (2005) 815–829.

[2] E.J. Carmona, J. Martínez-Cantos, J. Mira, Posprocesamiento morfológico adaptativo basado en algoritmos genéticos y orientado a la detección robusta de humanos, in: A. Fernández-Caballero, et al. (Ed.), 50 Años de la Inteligencia Artificial, Spain, 2006, pp. 249–261.

[3] E.J. Carmona, J. Martínez-Cantos, J. Mira, A new video segmentation method of moving objects based on blob-level knowledge, Pattern Recognit. Lett. 29 (2008) 272–285.

[4] P. Chiu, A. Girgensohn, W. Polar, E.G. Rieffel, L. Wilcox, F.H. Bennett, A genetic segmentation algorithm for image data streams and video, in: Proceedings of the Genetic and Evolutionary Computation Conference, 2000, pp. 666–673.

[5] W. Chojnacki, M.J. Brooks, A. van den Hengel, D. Gawley, A new constrained parameter estimator for computer vision applications, Image Vis. Comput. 22 (2004) 85–91.

[6] R. Collins, A. Lipton, T. Kanade, H. Fujiyoshi, D. Duggins, Y. Tsin, D. Tolliver, N. Enomoto, O. Hasegawa, A system for video surveillance and monitoring: VSAM final report, Technical Report CMU-RI-TR-00-12, Robotics Institute, Carnegie Mellon University, 2000.

[7] Y. Dedeoglu, Moving object detection, tracking and classification for smart video surveillance, Ph.D. Thesis, Department of Computer Engineering and The Institute of Engineering and Science of Bilkent University, 2004.

[12] M.A. Fernández, J. Mira, Permanence memory—a system for real time motion analysis in image sequences, in: Proceedings of the IAPR Workshop on Machine Vision Applications, 1992, pp. 249–252.

[8] A. Fernández-Caballero, J. Mira, M.A. Fernández, M.T. López, Segmentation from motion of non-rigid objects by neuronal lateral interaction, Pattern Recognit. Lett. 22 (14) (2001) 1517–1524.

[9] A. Fernández-Caballero, J. Mira, A.E. Delgado, M.A. Fernández, Lateral interaction in accumulative computation-A model for motion detection, Neurocomputing 50 (2003) 341–364.

[10] A. Fernández-Caballero, M.A. Fernández, J. Mira, A.E. Delgado, Spatio-temporal shape building from image sequences using lateral interaction in accumulative computation, Pattern Recognit. 36 (5) (2003) 1131–1142.

[11] A. Fernández-Caballero, M.T. López, M.A. Fernández, J. Mira, A.E. Delgado, J.M. López-Valles, Accumulative computation method for motion features extraction in active selective visual attention, Lect. Notes Comput. Sci. 3368 (2005) 206–215.

[13] G. Garai, B.B. Chaudhuri, A distributed hierarchical genetic algorithm for efficient optimization and pattern matching, Pattern Recognit. 40 (2007) 212–228.

[14] W.C. Gerken, L.K. Purvis, R.J. Butera, Genetic algorithm for optimization and specification of a neuron model, Neurocomputing 69 (2006) 1039–1042.

[15] N. Gheissari, A. Bab-Hadiashar, D. Suter, Parametric model-based motion segmentation using surface selection criterion, Comput. Vis. Image Underst. 102 (2006) 214–226.

[16] D. Hall, Automatic parameter regulation of perceptual systems, Image Vis. Comput. 24 (2006) 870–881.

[17] E.Y. Kim, K. Jung, Genetic algorithms for video segmentation, Pattern Recognit. 38 (2005) 59–73.

[18] E.Y. Kim, S.H. Park, Automatic video segmentation using genetic algorithms, Pattern Recognit. Lett. 27 (2006) 1252–1265.

[19] A. Leone, C. Distante, F. Buccolieri, A shadow elimination approach in video-surveillance context, Pattern Recognit. Lett. 27 (2006) 345–355.

[20] A.J. Lipton, H. Fujiyoshi, R.S. Patil, Moving target classification and tracking from real-time video, in: Proceedings of Workshop Applications of Computer Vision, 1998, pp. 129–136.
[21] M.T. López, A. Fernández-Caballero, J. Mira, A.E. Delgado, M.A. Fernández, Algorithmic lateral inhibition method in dynamic and selective visual attention task: application to moving objects detection and labeling, Expert Syst. Appl. 31 (2006) 570–594.
[22] M.T. López, A. Fernández-Caballero, M.A. Fernández, J. Mira, A.E. Delgado, Visual surveillance by dynamic visual attention method, Pattern Recognit. 39 (2006) 2194–2211.
[23] M.T. López, A. Fernández-Caballero, M.A. Fernández, J. Mira, A.E. Delgado, Motion features to enhance scene segmentation in active visual attention, Pattern Recognit. Lett. 27 (2006) 469–478.
[24] S. Mahmoodi, B.S. Sharif, Nonlinear optimisation method for image segmentation and noise reduction using geometrical intrinsic properties, Image Vis. Comput. 24 (2006) 202–209.
[25] J. Martínez-Cantos, E. Carmona, A. Fernández-Caballero, M.T. López, Mejora paramétrica de la interacción lateral en computación acumulativa, in: A. Fernández-Caballero, et al. (Ed.), 50 Años de la Inteligencia Artificial, Spain, 2006, pp. 262–273.
[26] K.E. Melkemi, M. Batouche, S. Foufou, A multiagent system approach for image segmentation using genetic algorithms and external optimization heuristics, Pattern Recognit. Lett. 27 (2006) 1230–1238.
[27] J. Mira, A.E. Delgado, A. Manjarrés, S. Ros, J.R. Alvarez, Cooperative processes at the symbolic level in cerebral dynamics: reliability and fault tolerance, in: R. Moreno-Diaz, J. Mira (Eds.), Brain Processes Theories and Models, MIT Press, Cambridge, 1996, pp. 244–255.
[28] J. Mira, A.E. Delgado, A. Fernández-Caballero, M.A. Fernández, Knowledge modeling for the motion detection task: the algorithmic lateral inhibition method, Expert Syst. Appl. 27 (2004) 169–185.
[29] M. Oral, U. Deniz, Centre of mass model—a novel approach to background modeling for segmentation of moving objects, Image Vis. Comput. 25 (2007) 1365–1376.
[30] N. Paragios, R. Deriche, Geodesic active contours and level sets for the detection and tracking of moving objects, IEEE Trans. Pattern Anal. Mach. Interface 22 (3) (2000) 266–280.
[31] J.L. Pedreño-Molina, M. Pinzotas, J. Monzó-Cabrera, A new methodology for in situ calibration of a neural network-based software sensor for S-parameter prediction in six-port reflectometers, Neurocomputing 69 (2006) 2451–2455.
[32] A. Pettinen, A. Yli-Harja, M.L. Linne, Comparison of automatic parameter estimation methods for neuronal signaling networks, Neurocomputing 69 (2006) 1371–1374.
[33] M. Ringer, J. Lasenby, A procedure for automatically estimating model parameters in optical motion capture, Image Vis. Comput. 22 (2004) 843–850.
[34] P. Spagnolo, T. D'Orazio, M. Leo, A. Distante, Moving object segmentation by background subtraction and temporal analysis, Image Vis. Comput. 24 (2006) 411–423.
[35] W. Tao, H. Jin, L. Liu, Object segmentation using ant colony optimization algorithm and fuzzy entropy, Pattern Recognit. Lett. 28 (2007) 788–796.
[36] I. Turkmen, K. Guney, D. Karaboga, Genetic tracker with neural network for single and multiple target tracking, Neurocomputing 69 (2006) 2309–2319.
[37] L. Wang, W. Hu, T. Tan, Recent developments in human motion analysis, Pattern Recognit. 36 (3) (2003) 585–601.
[38] D. Xu, X. Li, Z. Liu, Y. Yuan, Cast shadow detection in video segmentation, Pattern Recognit. Lett. 26 (1) (2005) 91–99.
[39] R.S. Youssif, C.N. Purdy, Combining genetic algorithms and neural networks to build a signal pattern classifier, Neurocomputing 61 (2004) 39–56.
[40] W. Zhang, X. Zhong Fang, X. Yang, Moving vehicles segmentation based on Bayesian framework for Gaussian motion model, Pattern Recognit. Lett. (2006) 956–967.
[41] T. Zhao, R. Nevatia, Tracking multiple humans in complex situations, IEEE Trans. Pattern Anal. Mach. Intell. 26 (9) (2004) 1208–1221.

Javier Martínez-Cantos is Scholarship holder of the Spanish Ministerio de Educación y Ciencia. He received his degree in Computer Science from the Universidad de Castilla-La Mancha, Spain, in 2004. He is currently pursuing his Ph.D. from the Department of Artificial Intelligence of the National University for Distance Education, Spain.

Enrique Carmona received his degree in Electronic Engineering from the University of Granada, Spain, in 1996, and received his Ph.D. in Physics from the National University for Distance Education, Spain, in 2003. Since 1996, he is an Assistant Professor with the Department of Artificial Intelligence of the National University for Distance Education, Spain. His research interests are in machine learning, data mining, and evolutionary computing.

Antonio Fernández-Caballero received his degree in Computer Science from the Technical University of Madrid, Spain, in 1993, and received his Ph.D. from the Department of Artificial Intelligence of the National University for Distance Education, Spain, in 2001. Since 1995, he is an Associate Professor with the Department of Computer Science at the University of Castilla-La Mancha, Spain. His research interests are in image processing, computer vision, neural networks, and agent technology. A. Fernández-Caballero is member of the IAPR.

María T. López received her degree in Physics from the University of Valencia, Spain, in 1991, and received her Ph.D. from the Department of Artificial Intelligence of the National University for Distance Education, Spain, in 2004. Since 1991, she is an Assistant Professor with the Department of Computer Science at the University of Castilla-La Mancha, Spain. Her research interests are in image processing and computer vision. María T. López is member of the IAPR.