Pattern Recognition Letters 30 (2009) 606–614

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Optimal recursive clustering of likelihood functions for multiple object tracking

Séverine Dubuisson *, Jonathan Fabrizio

Laboratoire d'Informatique de Paris 6 (LIP6/UPMC), 104 Avenue du Président Kennedy, 75016 Paris, France

A R T I C L E I N F O

Article history:
Received 2 March 2007
Received in revised form 3 October 2008
Available online 9 January 2009

Communicated by B. Kamgar-Parsi

Keywords:
Tracking
Clustering
Likelihood function
Motion characterization
Gabor filtering

A B S T R A C T

In this paper, we propose a method to track multiple deformable objects in sequences (with a static camera) in and beyond the visible spectrum by combining Gabor filtering and clustering. The idea is to sample moving areas between two frames by randomly positioning samples over high magnitude area of a motion likelihood function. These points are then clustered to obtain one class for each moving object. The novelty in our method is in using cluster information from the previous frame to classify new samples in the current frame: we call that a recursive clustering. This makes our method robust to occlusions, objects entering and leaving the field of view, objects stopping and starting, and moving objects getting really close to each other.

1. Introduction

Object tracking has been a topic of great interest in computer vision. The main source of difficulty lies in the ability to reliably determine whether a given moving area in the frame corresponds to the same object over time. In this paper, we describe a system for detecting and tracking multiple moving objects. We are particularly interested in situations where there are multiple moving and deforming objects, entering and leaving the field of view, being occluded, stopping and re-starting. Our approach is divided into two main steps. We first characterize the motion by filtering the difference image between two consecutive frames using a set of Gabor filter banks. This step results in a 2D motion likelihood function, on which we randomly position samples over high magnitude areas. The second step consists in clustering these samples to identify single moving objects: our goal is to associate each moving object with one class. New samples at time (t) are, either associated with clusters recovered at time $(t-1)$ or reclustered using a classical k-means algorithm, depending on their location in the current frame with respect to the previous one. Therefore the behavior of each cluster is tracked throughout the sequence. The main advantage of this technique is it resolves a lot of ambiguities in the classification because clusters from the previous frame are used to generate a new clustering in the new frame. This is especially efficient when different objects get closer to each other.

The remainder of this article is organized as follows. In Section 2 we classify information sources used for object tracking. In Section 3, we explain how to construct the motion likelihood function between two consecutive frames of a video sequence. In Section 4, we reveal how this function is sub sampled using what we call motion samples, and how these samples are then classified to associate one class with each moving area. We particularly outline the initialization and update steps of this algorithm. Our algorithm has been tested on several sequences in and beyond the visible spectrum and some results are given in Section 5. Finally, we give conclusions and perspectives in Section 6.

2. Information sources for object tracking

Tracking algorithms generally use either of two sources of information: a model of the appearance of the moving objects (color distribution, geometric model, etc.), or a model of the dynamic behavior of the objects. Recently, hybrid approaches have also been developed.

There is a large amount of literature on tracking methods using an appearance model of the object. Many of these approaches use a statistical description of the region of interest to perform the tracking. Region-based methods typically align the tracked regions between successive frames by minimizing a cost function (Hager and Belhumeur, 1998; Jurie and Dhome, 2001). Feature-based approaches (Shi and Tomasi, 1994) extract features (such as intensity, colors, edges, contours) and use them to establish correspondence between model images and target images. Mean shift (Comaniciu et al., 2003) is one of the most successful methods used for locating the optimal position of the tracked object, given a likelihood image

* Corresponding author. Fax: +33 1 44 27 74 95.
E-mail addresses: Severine.Dubuisson@lip6.fr (S. Dubuisson), Jonathan.Fabrizio@lip6.fr (J. Fabrizio).

whose peak is the most likely position of the tracked object. The likelihood image is generated by computing the probability that the given pixel belongs to the object, based on the distribution of feature values learned from the previous frames. Model-based tracking (Black and Jepson, 1998; Isard and MacCormick, 2001) can be very efficient for a particular application but often cannot be easily extended to a different class of target, because they incorporate prior information about the tracked objects. When objects are represented with parametric structures (usually curves), tracking is performed by seeking the lowest potential of a cost function of the parametric representation of the objects (shape, length, curvature, etc.): deformable models, snakes and appearance models (Tsechpenakis et al., 2004; Sclaroff and Isidoro, 2003; Chan et al., 1999) are the most popular techniques for tracking with small deformations.

Another source of information used by tracking methods is the dynamic behavior of the object of interest. For these so-called probabilistic trackers, the object is characterized by a state sequence $\{x_k\}_{k=1,\dots,n}$ whose evolution is specified by a dynamic equation $x_k = f_k(x_{k-1}, v_k)$. The goal of tracking is to estimate x_k given a set of observations. The observations $\{y_k\}_{k=1,\dots,m}$, with $m < n$, are related to the states by $y_k = h_k(x_k, n_k)$. Usually, f_k and h_k are vector-valued, nonlinear and time-varying functions, and v_k and n_k are white Gaussian noise sequences, independent and identically distributed. The Kalman filter and its extensions have been successful for tracking (Gao et al., 2005; Stenger et al., 2001), but they do not cope with highly non linear models or non Gaussian noise. Tracking methods based on particle filters (Noyer et al., 2005; Hue et al., 2002; Yang et al., 2005) can be applied under very weak hypotheses and track multiple objects even when there are occlusions by maintaining multiple hypotheses in the state space. Particle filters need observations to predict the states of the moving objects in a video sequence. Between two observations, the particles evolve according to the underlying Markov chain. Given a new observation, each particle is assigned a weight proportional to its likelihood. New particles are randomly sampled to favor particles with higher likelihood. Recently, new approaches combining appearance and dynamic models have been proposed (Okuma et al., 2004; Li et al., 2003). These methods are very robust, but unfortunately, they also have very high computational cost, especially when combining particle filters and learning approaches.

The proposed method does not use the appearance of the objects since it only works on the difference image and it does not incorporate any motion model either. Meanwhile, we use previous information to predict new positions of moving objects: our algorithm can also be classified as motion based.

3. Motion likelihood function

The first step of our approach consists in identifying and emphasizing the motion between two consecutive frames.

3.1. Motion identification

The Gabor wavelet transform (Lee, 1996) is defined as the convolution of a signal with a family of Gabor wavelets (e.g. filters). Gabor filters are modulation products of Gaussian kernels and complex sinusoidal signals. The output $\hat{f}(x,y)$ of a 2D Gabor filter $G(x,y)$ excited by an input signal $f(x,y)$ is given by $\hat{f}(x,y) = f \star G(x,y)$. A 2D Gabor filter bank $G_{f_0,\sigma,N}$ is defined by a group of N filters $G_i (i = 1,\dots,N)$ so that:

$$G_i(x,y) = \frac{1}{2\pi\sigma^2} \exp\left[-\left(\frac{x^2+y^2}{2\sigma^2}\right)\right] \times \exp[j2\pi f_0(x\cos\theta_i + y\sin\theta_i)] \tag{1}$$

where f_0 and $\theta_i = \frac{i\pi}{N}$ are the central frequency and the orientation of the sinusoidal wave, and σ is the scale (e.g. standard deviation) of the Gaussian kernel. The multi-resolution approach consists in filtering the image with a set of r 2D Gabor filter banks $G_{2^{-k}f_0,2^k\sigma,N}$, where $k = 0$ (respectively $k = r - 1$) corresponds to the lowest (respectively highest) resolution. For motion characterization, the larger the k, the larger the recovered motion. Bruno and Pellerin (2002) have worked on combining Gabor filters and optical flow and have shown that Gabor filter banks with parameters $\sigma \in [2,4]$, $f_0 < 0.15$ and $N \in [3,6]$ can be used to characterize motion. The issue with such a filtering technique is its computational time, that is why we have used a parallel algorithm for discrete Gabor transforms (Sudan et al., 2007) that significantly accelerate the filtering process.

In our case, instead of filtering the input sequence, we have chosen to filter the difference images between consecutive frames because they reveal the motion between frames. The main advantage of the multi-resolution multi-orientation filtering (Gabor) of a difference image is that it characterizes motions in all directions and with all magnitudes. In addition, it highlights the entire moving object, whereas the difference image (even after applying a Gaussian filter) only highlights the contours of the moving objects. This property can be seen in Fig. 2a: on the left we see the difference image between the first two frames of "Taxi" video sequence, and on the right its Gabor motion response. It is clear that the difference image is insufficient to well characterize motion because we need to sample detected motion area, and those highlighted by difference image are too thin.

3.2. Motion characterization

Given a sequence of N_f images $I^{(t)}, (t = 1,\dots,N_f)$, the first step consists in applying a CLAHE filter on the entire sequence (Leszczynski and Shalev, 1989) if the contrast is low: this often can happened when working on visual imagery, more rarely in infrared imagery. The filter enhances the contrast in the image by modifying local histograms to align them to uniform distributions. The goal is to remove the influence of contrast during the sequence. More precisely, the CLAHE algorithm partitions an images into contextual regions and applies the histogram equalization to each one. This evens out the distribution of used gray values and thus makes hidden features of the image more visible. Then, the characterization of the motion between the frames $I^{(t)}$ and $I^{(t-1)}$ consists in the steps in Algorithm 1.

Algorithm 1. Generate samples inside moving objects

```
Compute the image difference $D^{(t)} = |I^{(t)} - I^{(t-1)}|$
Compute the Gabor motion response $G^{(t)}$ using multiple Gabor filters
with different resolutions and orientations:
    $G^{(t)} = \sum_{i=1}^{N} \sum_{k=0}^{r-1} \left(D^{(t)} \star G_{2^{-k}f_0,2^k\sigma,i}\right)$    (2)
Filter $G^{(t)}$ with a median filter to eliminate noise
Position a set of samples $P^{(t)}$ randomly on high magnitude areas
```

In Eq. (2), we sum the responses from all Gabor filters to obtain an image where all moving objects are highlighted, regardless of the motion orientation or resolution. In Fig. 2a we show the difference image between the first two frames of the "Taxi" video sequence (see a frame in Fig. 2b) and its motion response (sum of the magnitude of the Gabor responses). It can be seen that the three moving vehicles are much easier to identify in the Gabor motion response than in the difference image itself. Also, the low contrast (for example the vehicle on the left side of the sequence) does not affect the detection of motion, which is a strong advantage in using Gabor filtering.

The last step in Algorithm 1 consists in randomly positioning samples only on all areas of high magnitude in $G^{(t)}$, also called motion regions. The most important aspect is not to position samples over all object area in the frames, but only on areas of motion. This does not depend on the size of the moving objects. The placement of new samples is done according to a Gaussian distribution over the areas of high magnitude (above a given threshold) in the filtered difference image (i.e. where movement is detected). The motion magnitude of the moving objects does not affect the number of samples that will be assigned to it. In fact, we want to characterize the motion of all objects and not only the objects that move a lot. The number N_p of samples for each object is a compromise between speed and localization accuracy. If we assign a lot of samples, the algorithm is slower, but the classes are very well characterized. We have determined empirically (see Fig. 1) that the tracking is successful with 100–300 samples per object.

4. Sample classification and tracking: recursive clustering

The idea behind our algorithm is to cluster the samples that were placed on moving areas and track the motion classes throughout the sequence. However, instead of computing a new clustering of the new samples at each frame, we update the classification from the previous frame with the new samples. For the first two frames of the sequence, we assume that the number of moving objects is known. Indeed, the initial number of classes is very important that is the reason why we use it as prior knowledge: the first image is the only observation we use (we call observation an image of the sequence for which we know specific informations: in our case, the number of moving objects). Everything else in the sequence is unknown. To automatically determine the number of classes is a difficult problem to which the clustering community has not yet provided an efficient solution. In the following frames, the number of moving objects can change and the partition is updated with the new samples.

In the next sections, we adopt the following notations. Let's consider the set $P^{(t)}$ of N_p samples randomly positioned on the Gabor motion response $G^{(t)}$ (see Section 3): $P^{(t)} = \{p_1^{(t)}, p_2^{(t)}, \ldots, p_{N_p}^{(t)}\}$. Our goal, for each frame of the sequence is to classify these samples into $N^{(t)}$ classes $\omega_k^{(t)} (k = 1, \ldots, N^{(t)})$ corresponding to the $N^{(t)}$ objects moving between frames $(t-1)$ and (t). $\Omega^{(t)} = \{\omega_1^{(t)}, \ldots, \omega_{N^{(t)}}^{(t)}\}$ is the set containing the $N^{(t)}$ classes of moving objects. In cases where the frame number is unambiguous, we will omit the superscript $^{(t)}$ to simplify the notation.

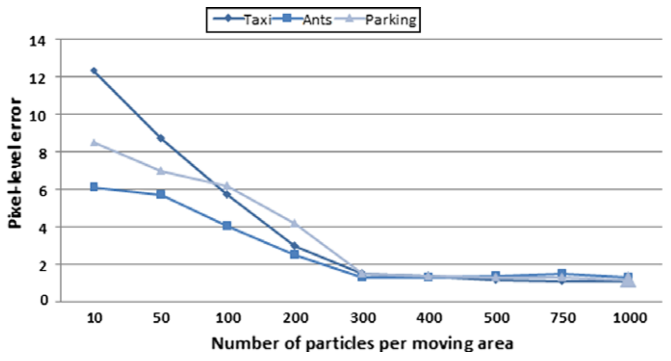

Fig. 1. Test on different sequences to determine the minimal number of samples per moving object needed to obtain satisfactory pixel-level accuracy.

4.1. The first two frames

The first classification occurs in frame (2) using the difference image $D^{(2)} = |I^{(2)} - I^{(1)}|$. In this case, we assume that the number $N^{(2)}$ of moving objects is known. Later in the sequence, this number will be automatically updated since our method can handle objects entering and leaving the field of view. Let $P^{(2)} = \{p_1^{(2)}, p_2^{(2)}, \ldots, p_{N_p}^{(2)}\}$ be the set of the N_p motion samples randomly sampled over high magnitude areas of the Gabor motion response. These samples are clustered using the k-means algorithm which consists of two steps. In the first step, a partition of patterns in k clusters is calculated, while, in the second step, the quality of the partition is evaluated. These two steps are repeated until the quality of the partitions is maximum (see (Jain and Dubes, 1988) for more details). We have decided to use two criteria to evaluate the quality of the partitions for the k-means algorithm: when the centers of the clusters do not change between two iterations, and when the partition minimizes the sum, over all clusters, of the within-cluster sums of points to cluster centroids distances. In our case, the feature vector for each sample consists of its 2D image coordinates. After the clustering algorithm, the samples in frame (2) are split into $N^{(2)}$ classes $\omega_k^{(2)}, k = 1, \ldots, N^{(2)}$. We can then compute statistics for each class.

Let p_{k_i} be the ith point in the kth class, $(k = 1, \ldots, N^{(t)}, i = 1, \ldots, n_k)$, where n_k is the number of samples of class k. For each class ω_k, we estimate the mean $\mu_k = \sum_{i=1}^{n_k} p_{k_i}$ and covariance matrix $\Sigma_k = \frac{1}{n_k - 1} \sum_{i=1}^{n_k} [(p_{k_i} - \mu_k)(p_{ki} - \mu_k)^t]$.

Assuming that p_{k_i} follows a Gaussian distribution $\mathcal{N}(\mu_k, \Sigma_k)$, the class can be represented by an ellipse E_k with center c_k, orientation θ_k, major and minor axes a_k and b_k, so that: $(c_k, \ \theta_k, \ a_k, \ b_k,) = \left(\mu_k, \ tan^{-1}\left(\frac{v_{1_y}}{v_{1_x}}\right), \ 2\sqrt{\lambda_1}, 2\sqrt{\lambda_2},\right)$ where λ_1 and λ_2 are the largest and smallest eigenvalue of Σ_k, and $(v_{1_x}, v_{1_y})^t$ is the eigenvector associated with λ_1. The computation of the class statistics is described by Algorithm 2.

Algorithm 2. Computation of class statistics

```
for $k = 1$ to $N^{(t)}$ do
    Compute statistics $\mu_k$ and $\Sigma_k$ of class $\omega_k$
    Model the class $\omega_k$ as an ellipse $E_k(c_k, \theta_k, a_k, b_k)$
end for
```

The algorithm for the first two frames is summarized in Algorithm 3. Fig. 2b shows the recovered ellipses on the three moving objects using the first two frames of the "Taxi" sequence.

Algorithm 3. First two frames

```
$t \leftarrow 2$
Find the set of samples $P^{(t)}$
Cluster $P^{(t)}$ into $N^{(t)}$ classes using the $k$-means algorithm
Class update: Algorithm 2
$\Omega^{(t-1)} \leftarrow \{\omega_k\}_{k=1,\ldots,N^{(t)}}$
$P^{(t-1)} \leftarrow P^{(t)}$
```

4.2. Update algorithm

Once the moving objects have been extracted using the first two frames of the sequence, the update algorithm is called for all the other frames in the sequence. Given a new frame at time (t), we use Algorithm 1 to extract the samples $P^{(t)} = \{p_1^{(t)}, p_2^{(t)}, \ldots, p_{N_p}^{(t)}\}$ around the moving areas. For each new sample, we can decide whether it belongs to one of the classes $\omega_k^{(t-1)}$ of $\Omega^{(t-1)}$ computed at frame $(t-1)$ by determining whether the sample is inside the corresponding ellipse $E_k^{(t-1)}$. This occurs when a part of the object in frame (t) overlaps with the same object in frame $(t-1)$.

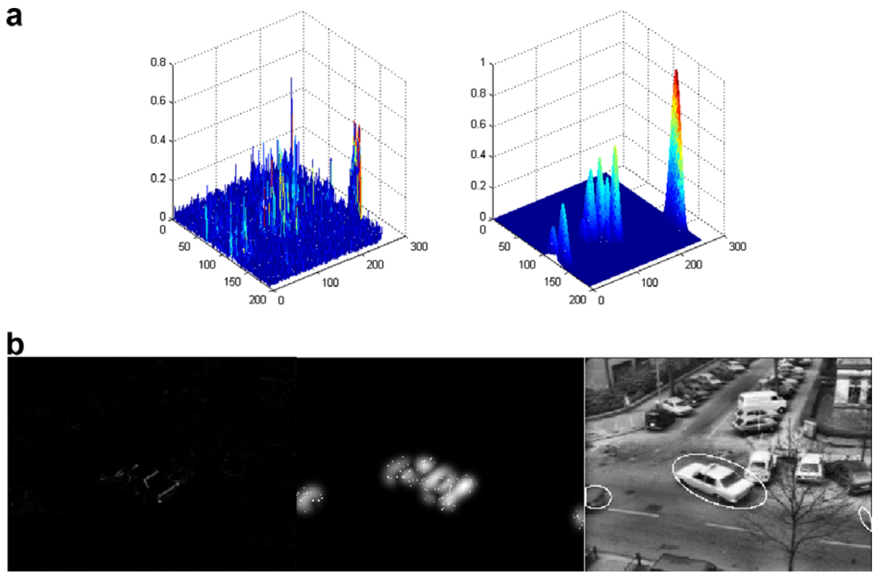

Fig. 2. First two frames of the "Taxi" video sequence. (a) Left: difference image function; Right: Gabor motion response using a set of $r = 3$ Gabor filter banks ($N = 6$ orientations, $\sigma = 2$). (b) Left: difference image. Middle: Gabor motion response with 300 randomly positioned samples (100 per object); Right: recovered ellipses around the three moving objects after applying the k-means algorithm with $k = 3$.

At each time frame, three situations occur (see Fig. 3 for illustration):

- Some samples are associated with some classes from $\Omega^{(t-1)}$, because the corresponding objects are still moving.
- Some samples cannot be classified into any of the $\omega_k^{(t-1)}$ classes, because a new object has appeared in the field of view, or an object already in the field of view has started moving.
- Some classes are such that no new samples were assigned to them, because an object has left the field of view, or an object has stopped moving.

All the samples that satisfy the first case are assigned to their corresponding class and the statistics for all the classes are updated. All the other samples will be revisited at a later stage (see Section 4.3). Algorithm 4 summarizes the steps of the update algorithm.

Algorithm 4. Update algorithm

```
$P_{\text{new}} = \emptyset$
Compute $P^{(t)}$ using Algorithm 1
**for all** $p_i \in P^{(t)}$ **do**
    **if** $\exists \omega_k \in \Omega^{(t-1)} : p_i$ inside $E_k^{(t-1)}$ **then**
        Classify $p_i$ into $\omega_k$
    **else**
        $P_{\text{new}} \leftarrow p_i$
    **end if**
**end for**
Class update: Algorithm 2
$\Omega^{(t)} \leftarrow \{\omega_k\}_{k=1,\dots,N^{(t)}}$
**if** $P_{\text{new}} \neq \emptyset$ **then**
    Algorithm 5
**end if**
Class update: Algorithm 2
$\Omega^{(t)} \leftarrow \{\omega_k\}_{k=1,\dots,N^{(t)}}$
$P^{(t-1)} = P^{(t)}$
$\Omega^{(t-1)} = \Omega^{(t)}$
```

4.3. Classification of new samples

One of the novelty in our method is the handling of new samples in $P^{(t)}$ that do not belong to any of the classes $\omega_k^{(t-1)} \in \Omega^{(t-1)}$ from the previous frame (the samples were placed in the set P_{new}). Two possibilities can happen for each sample:

- it can still be associated with a class from the previous frame because it is close enough to the corresponding ellipse; or
- it is too far from any of the existing classes because it corresponds to a new moving object (that entered the scene or that started moving).

The new sample classification scheme is given in Algorithm 5. In the first step, all new samples are clustered into small classes. Again, we use the k-means algorithm. Since the optimal number of classes is unknown, we use the method described in Chen et al. (2005) based on the gravitational clustering idea (Kundu, 1998) to automatically determine the number of classes. The goal is to apply the k-means algorithm for varying number of classes and find the optimal number k such that the partition minimizes a measure of distortion of the spatial distribution of samples in the clusters. The principle of the gravitational clustering is based on the notion of a force of attraction between each pair of points without using a similarity measure. The clusters are formed by slowly moving each new sample under the effect of those forces. Samples are merged together when they are too close to each other. We have chosen such an approach because it has been shown to be more accurate than k-means, fuzzy k-means or nearest neighbor classifiers (Jain and Dubes, 1988).

Let $C = \{c_1, \dots, c_n\}$ be the set of n classes obtained after classification of the new samples. The distance between a new class c_j and an existing class $\omega_k^{(t)} \in \Omega^{(t)}$ is defined as the average distance

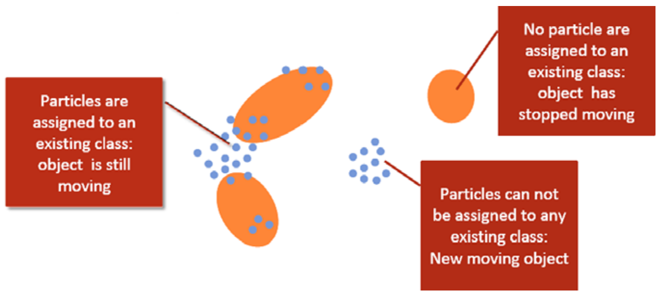
Particles are assigned to an existing class: object is still moving
No particle are assigned to an existing class: object has stopped moving
Particles can not be assigned to any existing class: New moving object

Fig. 3. Illustration of the three cases that can occur for new samples: previous computed classes are represented by ellipses and samples in the new frame by points.

between all points in class c_j and the ellipse $E_k^{(t)}$ corresponding to $w_k^{(t)}$: $D(c_j, \omega_k) = \frac{1}{n_{c_j}} \sum_{i=1}^{n_{c_j}} \text{dist}(p_i, E_k^{(t)})$, where n_{c_j} is the number of samples p_i in c_j and $\text{dist}(., E)$ is the distance from a point to an ellipse E (see (Eberly, 2000) for more details). The class $c_j \in C$ is merged with class $\omega_k^{(t)} \in \Omega^{(t)}$ if $\min_{k=1,\ldots,N^{(t)}} D(c_j, \omega_k^{(t)}) < T$, where T is a threshold depending on the size of the image and on the variance of samples in $\omega_k^{(t)}$: a new sample can be added to a class only if its meaning distance to the samples belonging to this class corresponds to the meaning distances between these samples. Otherwise, c_j is too far from the other classes: this corresponds to a new class (e.g. moving object). After all the samples have been assigned to a class (new or updated), the statistics and ellipses are updated for all classes (Section 4.1). At each frame, some classes $w_k^{(t-1)}$ from the previous frame were not assigned any samples. These classes are not added to the current set of classes $\Omega^{(t)}$, and therefore are abandoned.

Algorithm 5. Classification of new samples

```
$n \leftarrow 1$
**repeat**
    $n \leftarrow n + 1$
    Cluster $P_{\text{new}}$ into $n$ classes ($k$-means algorithm)
**until** all classes $c_j$ have a plausible density according to (Chen et al., 2005)
**for** $j = 1$ to $n$ **do**
    **if** $\exists \omega_k \in \Omega^{(t)}$: $D(c_j, \omega_k) < T$ **then**
        Merge $c_j$ and $\omega_k$
    **else**
        $\Omega^{(t)} \leftarrow c_j$
        $N^{(t)} \leftarrow N^{(t)} + 1$
    **end if**
**end for**
```

Fig. 4 shows comparative results of time computing of k-means algorithm and recursive k-means algorithm for the first 10 frames on the "Ant" sequence (see examples of images on Fig. 5). For this test, $N_p = 1800$ samples have been positioned on moving objects (about 300 per object). We can see the interest of using a recursive scheme for the classification of new samples: for each new frame, the k-means algorithm classifies 1800 samples, while our recursive scheme classifies, in mean, 400 samples. This is an important save of time.

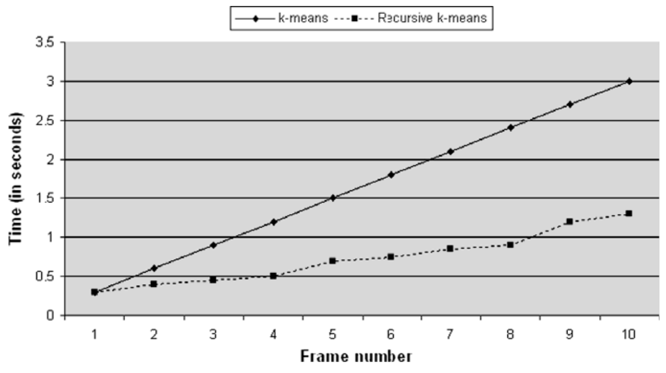

Fig. 4. Computation times for the processing of the "Ant" sequence (see examples of images in Fig. 5). The recursive scheme of k-means algorithm makes decrease computation times, comparing to a classical k-means algorithm.

Table 1 shows the advantage of using our approach of recursive clustering to improve tracking when two objects get close. For example, we can see that when the distance between two objects is smallest than five pixels, we correctly classify 88% of the new samples, when the k-means only classify 66% of these samples.

Considering these two tests, we can affirm that our approach gives a better classification rate of motion samples, giving a more robust object tracking, with a save of time.

5. Results

We have tested our method on several sequences and the algorithm was able to track all the objects in all the sequences. For all these sequences, the difference images between consecutive frames were filtered using a set of Gabor filter banks with parameters: $r = 3, f_0 = 0.14, \sigma = 2$ and $N = 6$. Some motion Gabor response examples are shown in Fig. 2b on the "Taxi" sequence. In this section, we present some interesting results obtained on complex sequences, in and beyond the visible spectrum, to illustrate the robustness of our approach. In each sequence, we have manually pointed moving objects (and fit an ellipse around them) to generate the real trajectory of them. We then can compare our tracking results with these trajectories by calculating the meaning pixel-level error between the real positions of objects and our estimations of them.

5.1. Qualitative results

The "Ant" sequence shows ants moving on a flat surface. Each ant has a complex movement: translation while walking and deformations while moving its head, abdomen, or antennas. In addition, there is a hole in the middle of the flat surface from which ants can disappear and reappear. The sequence contains 74 frames. We have tested our algorithm on this sequence using 1800 samples and all the ants were tracked properly throughout the sequence. Some results are shown in Fig. 5. The right most picture shows the trajectories of all the ants during the sequence. It can be seen that the trajectories are fairly complex and the ants get very close

Table 1
Illustration of the advantage of our recursive clustering approach. New sample classification error rate when two moving objects get close (the distance between objects is given in pixel).

Distance between objects	k-mean clustering (%)	Recursive k-means clustering (%)
30 Pixels	0	0
20 Pixels	1	0
10 Pixels	5	0
5 Pixels	15	5
2 Pixels	33	12

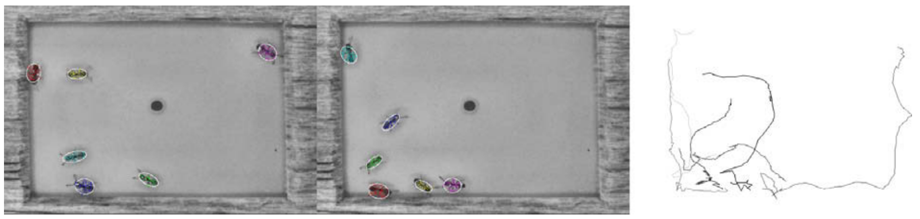

Fig. 5. Results on the "Ant" sequence. Detection of moving ants on frames 5 and 52. The rightmost image contains the trajectories of all the ants during the 74 frame sequence.

to each other during the sequence. Nevertheless, the algorithm was always able to resolve the ambiguities. In our test, the meaning pixel-level error is 5.08.

The "Parking" sequence shows moving cars and pedestrians. All kinds of complicated cases are present in the sequence: some of these objects leave the field of view, others stop and restart their motion, some moving objects occlude other moving objects, the contrast is not always very clear between the background and the moving object. Fig. 6 shows some of the frames in the 2500 frame sequence. Despite all the complicated situations, the algorithm is always able to correctly track all the moving objects.

The OTCBVS benchmark dataset collection contains infrared videos. Fig. 7 shows some of the results on three of the sequences. It can be seen that despite numerous people entering and leaving the field of view, the proposed method was always successful in tracking all the moving people.

We also have worked in a part of the sequence "Tennis" where the player bounces the ping pong ball on his racket (approximatively 25 frames), because this is the only part where the camera does not move. The motion of the ball is not regular (acceleration during descendant phase, deceleration during ascendant phase). The use of a multi resolution and orientation filtering makes our algorithm robust to the speed changes. Moreover, when the ball bounces off of the racket, two classes (one for the ball and one for the racket) are well detected and differentiated. Fig. 8 shows tracking results on several images of this sequence, with $N_p = 600$ samples. Fig. 9 shows the number of misclassified samples along the sequence. We can notice that this number increases in frames 16–20, because the ball and the racket are very close in these frames. However, the meaning error is 23.68 samples among the 600 used in this test, giving a meaning rate of 0.03.

The "Hall" sequence shows people walking alone or in groups in an hallway. A part of the hall is exposed to the sun that induces a changing in illumination. In this test we have used 1000 samples. In Fig. 10, we show the result of our algorithm on the part of the sequence where a man is walking around the sunlit area. Despite the changing in illumination, the man has been correctly tracked (Fig. 10, images 1–3). Moreover, we can see a group of people that split and a new class is well detected (Fig. 10, image 4). Fig. 11 shows the number of misclassified samples along the sequence. During frames 15–30, there is a changing in illumination. We can see the number of misclassified samples does not increase. Between frames 40 and 60, two groups of people are getting closer, then merging. Number of misclassified samples first increases, until a fusion process is performed by our algorithm. Similarly, this group of person is splitting between frames 100 and 120. After few frames, the two classes are detected and the number of misclassified samples decreases. Our algorithm is first disturbed by merging and splitting events, but can quickly change the number of classes to deal with these new situations.

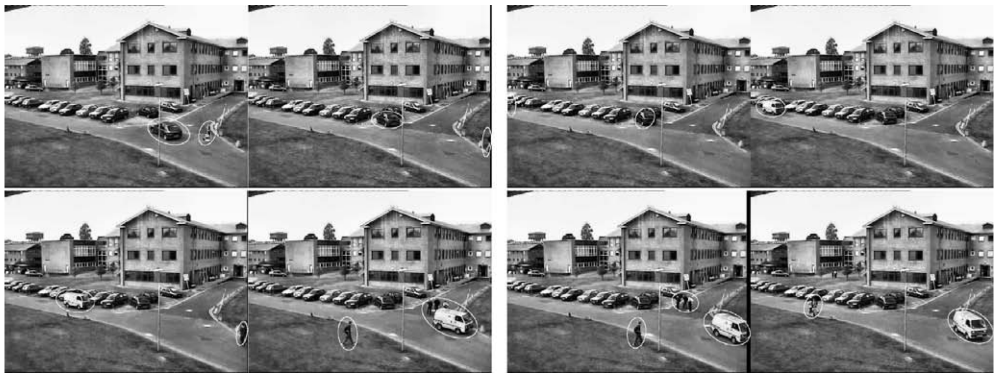

Fig. 6. Results obtained on the "Parking" sequence (from left to right, from top to bottom): 1. initialization; 2. same number of classes; 3. one car entering the field of view (left) and a pedestrian is leaving (right); 4. car stopping (center); 5. people entering the field of view (right); 6. during occlusion (center), classes are merged; 7. after occlusion, classes are separated again; and 8. car backing up and starting in the opposite direction (right).

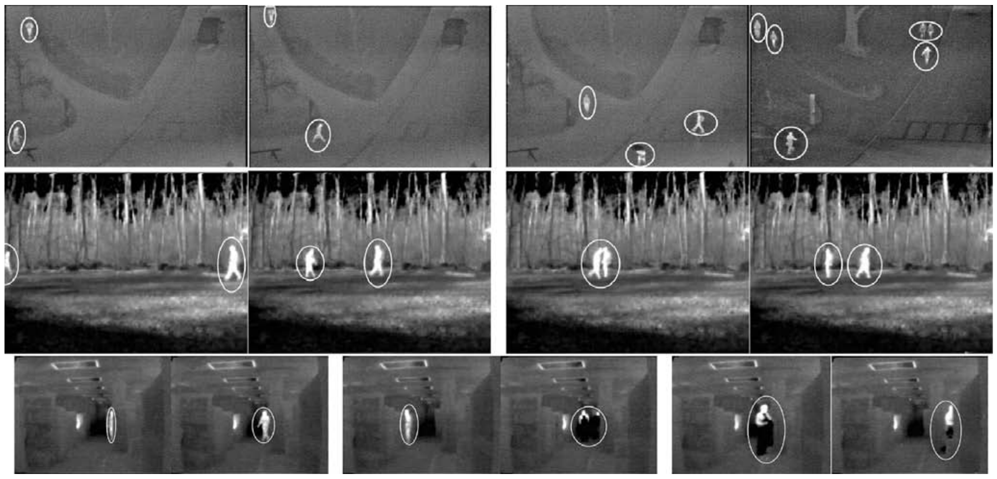

Fig. 7. Results obtained on the OTCBVS benchmark dataset collection for different datasets: 1. first row: Dataset 01: OSU Thermal Pedestrian Database; 2. middle row: Dataset 05: Terravic Motion IR Database (outdoor motion and tracking); and 3. last row: Dataset 05: Terravic Motion IR Database (indoor hallway surveillance).

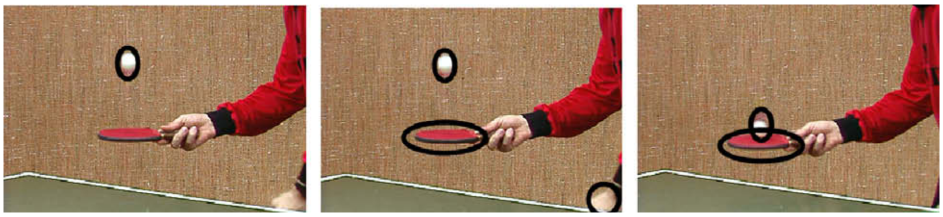
Fig. 8. Results on the "Tennis" sequence: from left to right: 1. detection of the ball with a quasi null motion (top of the parabola of the trajectory); 2. the ball is accelerating: the racket, the ball and the hand are well detected; and 3. contact between the ball and the racket, two classes are still detected.

5.2. Comparisons with particle filter

We have compared our results with those obtained with particle filter based approach on the "Ant 2" sequence (102 frames) on which motion is unknown and highly nonlinear and erratic (see Fig. 5 for examples of images). We use the parameters described at the beginning of this Section for our method and have used 15 frames as observations for the particle filter. We have compared with the Joint Probability Data Association Filter (JPDAF) (Bar-Shalom and Fortmann, 1988), that uses a weighted sum of all measurements near the predicted state, each weight corresponding to the posteriori probability for a measurement to come from a target. JPDAF provides an optimal data solution in the Bayesian framework filter. Fig. 12 shows the comparative results when we use 600 particles (or samples) for JPDAF and our algorithm. If we compute the meaning pixel-level error computation we obtain 7.8 for particle filter, and 5.08 for our method. The error given by JPDAF inscrease when ants change their direction or get closer (see Fig. 13), showing that our algorithm is more robust in such cases. Fig. 13 gives a comparison of meaning pixel-level error between particle filter and our method. We can see this error is more important when using particle filter, especially when ants are getting closer (see for example error given in frames 7–10 and 34–40). Fig. 14 shows the computing time provided by JPDAF and our approach. We can see our method is less time consuming. This is due to the fact that for JPDAF, the number of possible hypothesis increases rapidly with the number of targets, and also the number of particles requiring prohibitive amount of time calculating.

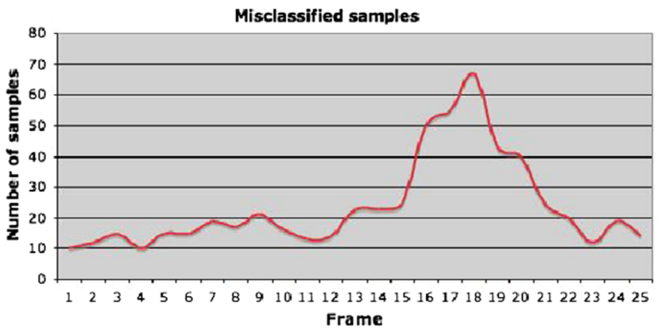

Fig. 9. "Tennis" sequence. Number of misclassified samples along the sequence.

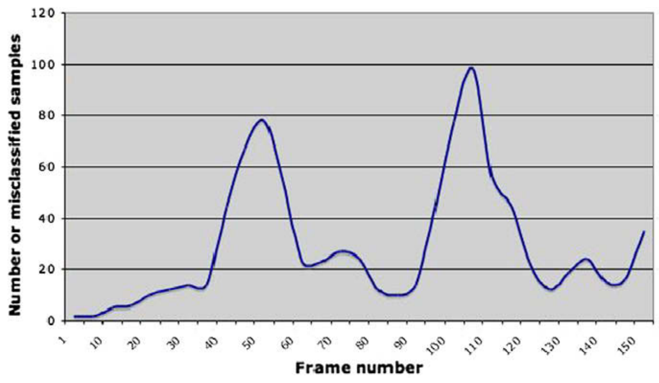

Fig. 11. "Hall" sequence. Misclassified sample along the sequence.

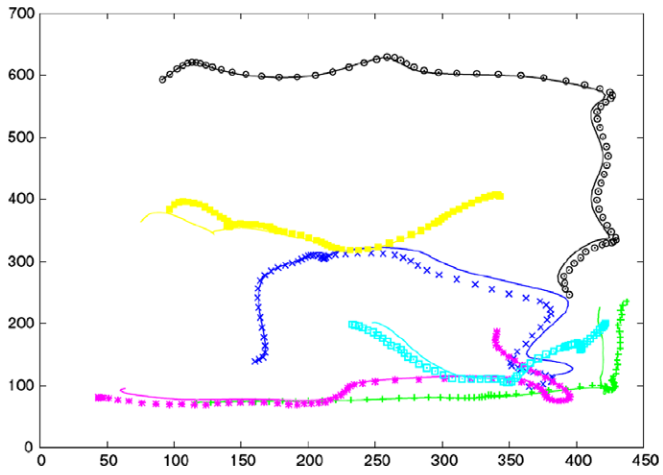

Fig. 12. Comparison tests on the "Ants" sequence: dotted lines show trajectories obtained with particle filter, and solid lines the trajectories obtained with our method. 600 particles (samples) are used.

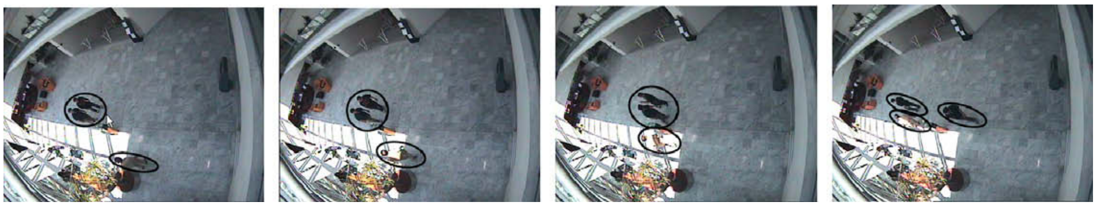
Fig. 10. Results on the "Hall" sequence (from left to right): 1. two classes are detected: one group of two people and one person alone; 2. changing in illumination (from dark to bright) around the moving person; 3. the two classes are close to each other; and 4. changing in illumination (from bright to dark) around the moving person and splitting of a group: three classes are detected.

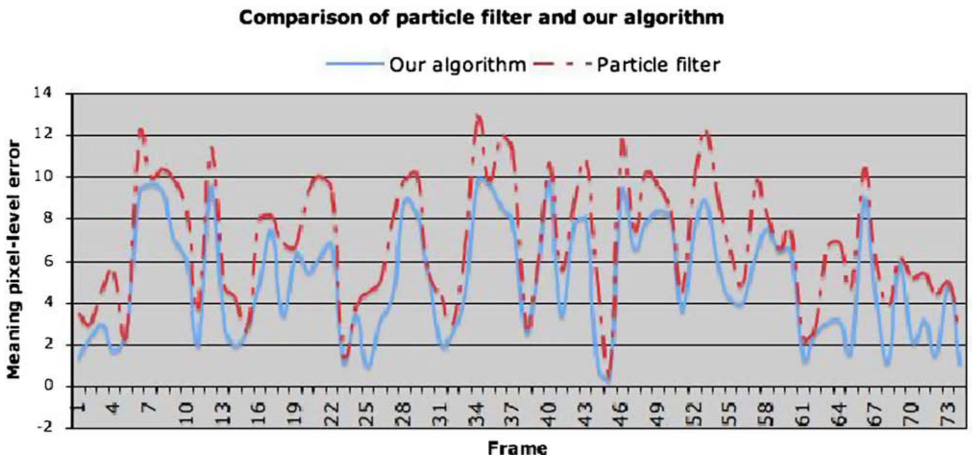

Fig. 13. "Ant" sequence. Comparison of meaning pixel-level error between particle filter and our method for the first 74 frames. 600 particles (samples) are used.

Fig. 15 shows the meaning pixel-level error among the sequence, depending on the number of particles (or samples) used in JPDAF and in our algorithm. For both approaches the error is important when the number of particles (or samples) is small. We can however see our algorithm provides a reasonable error despite a small number of samples (around 500 for the six moving objects) used comparing to JPDAF. JPDAF and our algorithm give similar errors when the number of particles is greater than 2000. In this particular sequence, the error is due to the proximity of ants.

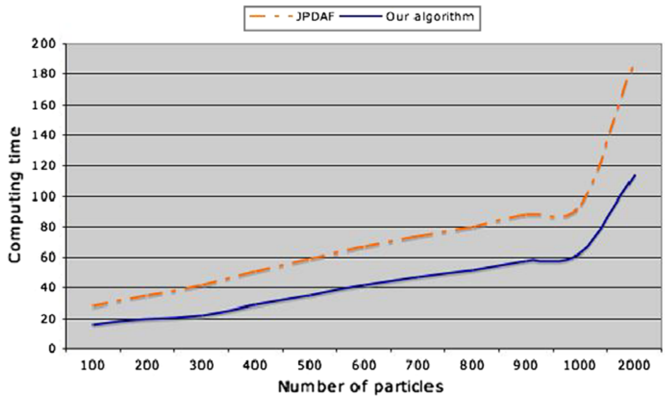

Fig. 14. "Ant" sequence. Comparison of computing time depending on the number of particles (samples in our algorithm), between JPDAF and our approach.

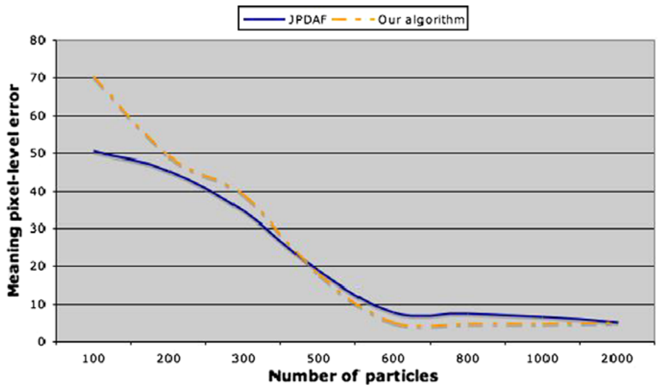

Fig. 15. "Ant" sequence. Comparison of meaning pixel-level error depending on the number of particles (samples in our algorithm), between JPDAF and our approach.

6. Conclusion

In this article, we have proposed a new approach for object tracking in video sequences in and beyond the visible spectrum with a static camera that can deal with multiple targets, occlusions, non-rigid motions, objects entering and leaving the field of view, objects stopping and starting, and moving objects getting really close to each other. The novelty of this method is that motion samples are not only classified in each frame, but classes are also updated in every frame. This is like a recursive clustering. This updating makes the classification step more robust, since in each frame, the approximate locations of the motion classes are known (from the previous frame). In addition, this recursive *k*-means algorithm saves time since we do not have to classify all the samples every time. The scheme is quite simple but highly robust.

Results that have been published in the literature demonstrate that particle filters are the best tools currently available to robustly track multiple objects, with deformations, occlusions, entry and exit of objects in the scene. The algorithm that we propose can also handle these situations. In our case, we do not need to know any motion model (via a transition function) for objects contrary to the particle filter. In fact, without any transition function, the particle filter would fail because the particle would not be propagated in good directions, and the update step would cause the degeneracy of particles. This is not the case of our algorithm: particles are regenerated in each frame on high motion area and then associated to classes (i.e. moving objects). Another advantage of our method over particle filters is that there is no need to learn the appearance (color, shape) of the objects that have to be tracked. Also, the changes in illumination are taken into account in our approach because we are working on the difference image Gabor response, which also means that the tracked objects can change their color over time without affecting the tracking results. In other words, our approach only focuses on movement detection independently of the shape, color, size, etc. of the object. Finally, our algorithm can handle erratic motions (see results on "Ant" sequence). Gabor filtering can be time consuming that is why we have used the recursive scheme that was proposed in Sudan et al. (2007) to speed it up. Our tests on different complicated sequences have shown that this method is very robust. In the future, we attempt to improve the algorithm, that now does not differentiate an object that stops and restarts from an object that appears. In both situations, the object is seen as a new object. We are currently conducting more research to label objects and identify whether an existing object has stopped and restarted or a new object has appeared. This is done using probabilities of appearance in regions of the images. Our method also does not handle occlusions.

Finally, we could determine the motion sample number (before the classification step) in an adaptive way, so that dense motion area could be more characterized. This would give a best compromise between speed and characterization.

References

Bar-Shalom, Y., Fortmann, T.E., 1988. Tracking and data association. Mathematics in Science and Engineering. Academic Press.

Black, M.J., Jepson, A.D., 1998. Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. IEEE Int. J. Comput. Vis. 26 (1), 63–84.

Bruno, E., Pellerin, D., 2002. Robust motion estimation using spatial Gabor-like filters. Signal Process. 82, 297–309.

Chan, S., Ngo, C.W., Lai, K.F., 1999. Motion tracking of human mouth by generalized deformable models. Pattern Recognit. Lett. 20 (9), 879–887.

Chen, C.-Y., Hwang, S.-C., Oyang, Y.-Y., 2005. A statistics-based approach to control the quality of subclusters in incremental gravitational clustering. Pattern Recognit. 38, 2256–2269.

Comaniciu, D., Ramesh, V., Meer, P., 2003. Kernel-based object tracking. IEEE Trans. Pattern Anal. Machine Intell. 25 (5), 564–577.

Eberly, D.H., 2000. 3D Game Engine Design: A Practical Approach to Real-Time Computer Graphics, Morgan Kaufmann, Book and CD Rom Edition.

Gao, J., Kosaka, Akio, Kak, Avinash C., 2005. A multi-Kalman filtering approach for video tracking of human-delineated objects in cluttered environments. Comput. Vis. Image Understanding 99 (1), 1–57.

Hager, G., Belhumeur, P., 1998. Efficient region tracking with parametric models of geometry and illumination. IEEE Trans. Pattern Anal. Machine Intell. 20 (10), 1025–2039.

Hue, C., Le Cadre, J.-P., Prez, P., 2002. Sequential Monte-Carlo methods for multiple target tracking and data fusion. IEEE Trans. Signal Process. 50 (2), 309–325.

Isard, M., MacCormick, J., 2001. BraMBLe: A Bayesian multiple-blob tracker. Int. Conf. Comput. Vis. (ICCV), Vancouver, Canada.

Jain, A.K., Dubes, R.C., 1988. Algorithms for Clustering Data. Prentice Hall.

Jurie, F., Dhome, M., 2001. Real time template matching. IEEE Int. Conf. Comput. Vis. (ICCV), Vancouver, Canada, pp. 544–549.

Kundu, S., 1998. Gravitational clustering: A new approach based on the spatial distribution of the points. Pattern Recognit. 32, 1149–1160.

Lee, T.S., 1996. Image representation using 2D Gabor wavelets. IEEE Trans. Pattern Anal. Machine Intell. 18 (20), 959–971.

Leszczynski, K.W., Shalev, S., 1989. A robust algorithm for contrast enhancement by local histogram modification. Image Vis. Comput. 7 (3), 205–209.

Li, P., Zhang, T., Pece, A.E.C., 2003. Visual contour tracking based on particle filters. Image Vis. Comput. 21, 111–123.

Noyer, J.-C., Lanvin, P., Benjelloun, M., 2005. Non-linear matched filtering for object detection and tracking. Pattern Recognit. Lett. 25 (6), 655–668.

Okuma, K., Taleghani, A., De Freitas, N., Little, J.J., Lowe, D., 2004. A boosted particle filter: Multi target detection and tracking. In: European Conference on Computer Vision (ECCV), Prague, Czech Republic.

Sclaroff, S., Isidoro, J., 2003. Active blobs: Region-based, deformable appearance models. Comput. Vis. Image Understanding 89 (2–3), 197–225.

Shi, J., Tomasi, C., 1994. Good features to track. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, USA.

Stenger, B., Mendona, P.R., Cippola, R., 2001. Model-based hand tracking using an unscented Kalman filter. In: Proceedings of the British Machine Vision Conference, Manchester, UK.

Sudan, K., Saggar, N., De, A., 2007. A parallel algorithm for discrete Gabor transforms. In: Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA), Las Vegas, USA.

Tsechpenakis, G., Rapantzikos, K., Tsapatsoulis, N., Kollais, S., 2004. A snake model for object tracking in natural sequences. Signal Process.: Image Commun. 19 (3), 219–238.

Yang, C., Duraiswami, R., Davis, L., 2005. Fast multiple object tracking via a hierarchical particle filter. IEEE Int. Conf. Comput. Vis. (ICCV), 212–219.