Hindawi Publishing Corporation
EURASIP Journal on Image and Video Processing
Volume 2011, Article ID 163682, 15 pages
doi:10.1155/2011/163682

Research Article

Motion Pattern Extraction and Event Detection for Automatic Visual Surveillance

Yassine Benabbas, Nacim Ihaddadene, and Chaabane Djeraba

LIFL UMR CNRS 8022 - Université Lille1, TELECOM Lille1, 59653 Villeneuve d'Ascq Cedex, France

Correspondence should be addressed to Yassine Benabbas, yassine.benabbas@lifl.fr

Received 1 April 2010; Revised 30 November 2010; Accepted 13 December 2010

Academic Editor: Luigi Di Stefano

Efficient analysis of human behavior in video surveillance scenes is a very challenging problem. Most traditional approaches fail when applied in real conditions and contexts like amounts of persons, appearance ambiguity, and occlusion. In this work, we propose to deal with this problem by modeling the global motion information obtained from optical flow vectors. The obtained direction and magnitude models learn the dominant motion orientations and magnitudes at each spatial location of the scene and are used to detect the major motion patterns. The applied region-based segmentation algorithm groups local blocks that share the same motion direction and speed and allows a subregion of the scene to appear in different patterns. The second part of the approach consists in the detection of events related to groups of people which are merge, split, walk, run, local dispersion, and evacuation by analyzing the instantaneous optical flow vectors and comparing the learned models. The approach is validated and experimented on standard datasets of the computer vision community. The qualitative and quantitative results are discussed.

1. Introduction

In the recent years, there has been an increasing demand for automated visual surveillance systems: more and more surveillance cameras are used in public areas such as airports, malls, and subway stations. However, optimal use is not made of them since the output is observed by a human operator, which is expensive and unreliable. Automated surveillance systems try to integrate real-time and efficient computer vision algorithms in order to assist human operators. This is an ambitious goal which has attracted an increasing amount of researchers over the years. They are used as an active real-time medium which allows security teams to take prompt actions in abnormal situations or simply label the video streams to improve the indexing/retrieval platforms. These kinds of intelligent systems are applicable to many situations, such as event detection, traffic and people-flow estimation, and motion pattern extraction. In this paper we will focus on motion pattern extraction and event detection applications.

Learning typical motion patterns from video scenes is important in automatic visual surveillance. It can be used as a mid-level feature in order to perform a higher-level analysis of the scene under surveillance. It consists of extracting usual or repetitive patterns of motion, and this information is used in many applications such as marketing and surveillance. The extracted patterns are used to estimate consumer demographics in public spaces or to analyze traffic trends in road traffic scenes.

Motion patterns are also used to detect the events that occur in the scene under surveillance by improving the detection, the tracking and behavior modeling, and understanding of the object in the scene. We define an event as the interesting phenomena which captures the user's attention (e.g., running event in crowd, goal event in sports challenges, traffic accidents, etc.) [1]. An event occurs in a high-dimensional spatiotemporal space and is described by its spatial location, its time interval, and its label. We will focus our approach on six crowd-related events which are labeled: walking, running, splitting, merging, local dispersion, and evacuation.

This paper describes a real-time approach for modeling the scenes under surveillance. The approach consists of modeling the motion orientations over a certain number of

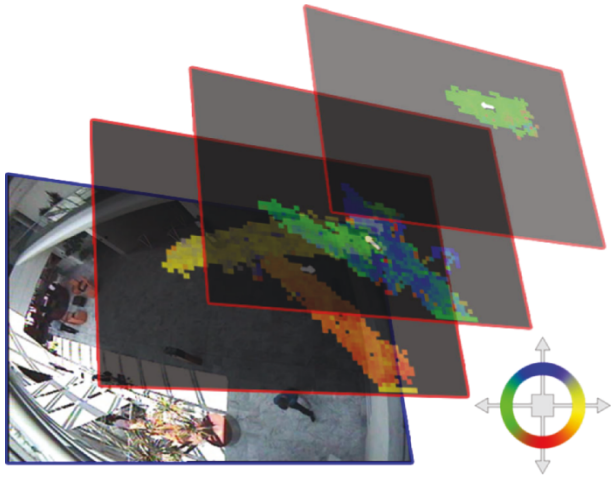


Figure 1: Learned motion patterns on a sequence from the Caviar dataset.

frames in order to estimate a *direction model*. This is done by performing a circular clustering at each spatial location of the scene in order to determine their major orientations. The direction model has various uses depending on the number of frames used for its estimation. In this work, we put forward two applications. The first one consists of detecting typical motion patterns of a given video sequence. This is performed by estimating the direction model by using all the frames of that sequence; the direction model will contain the major motion orientations of the sequence at each spatial location. Then we apply a region-based segmentation algorithm to the direction model. The retrieved clusters are the typical motion patterns, as shown in Figure 1 where three motion patterns are detected. This figure shows the entrance lobby of the INRIA labs. Each motion pattern in the black frame is defined by its main orientation and its area on the scene.

The second application is motion segmentation, which detects groups of objects that have the same motion orientation. We locate groups of persons on a frame by determining the direction model of the immediate past and future of that frame, and then grouping similar locations on the direction model. Then, we use the positions, distances, orientations, and velocities of the groups to detect the events described earlier.

Our work is based on the idea that entities that have the same orientation form a single unit. This is inspired by gestaltism or Gestalt psychology [2], a theory of mind and brain positing which states in the law of common fate that *elements with the same moving direction are perceived as a collective or unit*. In this work, we rely mostly on motion orientation as opposed to a semidirectional model [3] because gestaltism does not consider motion speed. In fact, we can see in real life that moving objects that follow the same patterns do not necessarily move at the same speed. For example, in a one-way road, cars move at different speeds while sharing the same motion pattern. In addition, augmenting the direction model with the motion speed information will increase the computation burden which is not desired in real-time systems.

The remainder of this paper is organized as follows: firstly, in Section 2 we highlight some relevant works on motion pattern recognition and event detection in automatic video surveillance. Section 3 details the estimation of the *Direction Model*. Then Section 4 presents the motion pattern extraction algorithm using the direction model. In Section 5 we detail the event recognition module. We present the experiments and result of our motion pattern extraction and event detection approaches in Section 6. The experiments were performed using datasets retrieved from the web (such as PETS (http://www.cvg.rdg.ac.uk/PETS2009/index.html) and CAVIAR (http://groups.inf.ed.ac.uk/vision/CAVIAR/CAVIARDATA1/) datasets) and annotated by a human expert. Finally, we give our concluding remarks and discuss potential extensions of the work in Section 7.

2. Related Works

The problems of motion pattern extraction and crowd event detection in visual surveillance are not new [4–8]. These problems are related because in general the approaches detect events using motion patterns following these steps: (i) detection and tracking of the moving objects present in the scene, (ii) extraction of motion patterns from the tracks, and eventually (iii) detection of events using motion patterns information.

2.1. Object Detection and Tracking. Many object detection and tracking approaches have been proposed in the literature. A well-known method consists in tracking blobs extracted via background subtraction approaches [9–11] where a blob represents a physical object in the scene such as a car or a person. The blobs are tracked using filters such as the Kalman filter or the particle filter. These approaches have the advantage of directly mapping a blob to a physical object which facilitates object identification. However, they experience poor performance when the lighting conditions change and when the number of objects is very important and occluded.

Another type of approach detects and tracks the points of interest (POI) [12–14]. These points consist in corners, edges, or other features which are relevant for tracking. They are then tracked using optical flow techniques. The detection and tracking of POIs requires less computation resources. However, physical objects are not directly detected because the objects here are the POIs. Thus, physical object identification is more complex using these approaches.

2.2. Motion Pattern Extraction. Once the objects have been detected and extracted, the motion patterns can be extracted using various algorithms that we classify as follows.

Iterative Optimization. These approaches group the trajectories of moving objects using simple classifiers such as

K-means. Hu et al. [15] generate trajectories using fuzzy K-means algorithms for detecting foreground pixels. Trajectories are then clustered hierarchically and each motion pattern is represented with a chain of Gaussian distributions. These approaches have the advantage of being simple yet efficient. However, the number of clusters must be specified manually and the data must be of equal length, which weakens the dynamic aspect.

Online Adaptation. These approaches integrate new tracks on the fly as opposed to *iterative optimization* approaches. This is possible using an additional parameter which controls the rate of updates. Wang et al. [16] propose a trajectory similarity measure to cluster the trajectories and then learn the scene model from trajectory clusters. Basharat et al. [17] learn patterns of motion as well as patterns of object motion and size. This is performed by modeling pixel-level probability density functions of an object's position, speed, and size. The learned models are then used to detect abnormal tracks or objects. These approaches are adapted to real-time applications and time-varying scenes because the number of clusters is not specified and they are updated over time. There is also no need for the maintenance of a training database. However, it is difficult to select a criterion for new cluster initialization that prevents the inclusion of outliers and insures optimality.

Hierarchical Methods. These approaches consider a video sequence as the root node of a tree where the bottom nodes correspond to individual tracks. Hu et al. [18] detect sequence's motion patterns by clustering its motion flow field, in which each motion pattern consists of a group of flow vectors participating in the same process or motion. However, the suggested algorithm is designed only for structured scenes and fails on unstructured ones. It requires that a maximum number of patterns are specified and for that number to be slightly higher than the number of desired clusters. Zhang et al. [19] model pedestrians' and vehicles' trajectories as graph nodes and apply a graph-cut algorithm to group the motion patterns together. These approaches are well suited for graph theory techniques which make binary divisions (such as max-flow and min-cut). In addition, the multiresolution clustering allows a clever choice of the number of clusters. The drawback is the quality of the clusters which is dependent on the decision of how to split (merge) a set that is not generally reflected along the tree.

Spatiotemporal Approaches. These approaches use time as a third dimension and consider the video as a 3d volume (x, y, t). Yu and Medioni [20] learn the patterns of moving vehicles from airborne video sequences. This is achieved using a 4D representation of motion vectors, before applying tensor voting and motion segmentation. Lin et al. [21] transform the video sequence into a vector space using a Lie algebraic representation. Motion patterns are then learned using a statistical model applied to the vector space. Gryn et al. [22] introduce the *direction map* as a representation that captures the spatiotemporal distribution of motion direction across regions of interest in space and time. It is used for recovering direction maps from video, constructing direction map templates to define target patterns of interest, and comparing predefined templates to newly acquired video for pattern detection and localization. However, the direction map is able to capture only a single major orientation or motion modality at each spatial location of the scene.

Cooccurence Methods. These methods take advantage of the advances in document retrieval and natural language processing. The video is considered as a document and a motion pattern as a bag of words. Rodriguez et al. [23] propose to model various crowd behavior (or motion) modalities at different locations of the scene by using a Correlated Topic Model (CTM). The learned model is then used as a priori knowledge in order to improve the tracking results. This model uses motion vector orientation, subsequently quantized into four motion directions, as a low-level feature. However, this work is based on the manual division of the video into short clips and further investigation is needed as to the duration of those clips. Stauffer and Grimson [24] use a real-time tracking algorithm in order to learn patterns of motion (or activity) from the obtained tracks. They then apply a classifier in order to detect unusual events. Thanks to the use of cooccurrence matrix from a finite vocabulary, these approaches are independent from the trajectory length. However, the vocabulary size is limited for effective clustering and time ordering is sometimes neglected.

Evaluation Approaches. The evaluation of motion pattern extraction approaches is difficult and time consuming for a human operator. Although the best evaluation is still performed by a human expert, we find approaches that define metrics and evaluation methodologies for automatic and in-depth evaluation. Morris and Trivedi [25] perform a comparative evaluation on approaches that uses clustering methodologies in order to learn trajectory patterns. Eibl and Brändle [26] find motion patterns by clustering optical flow fields and propose an evaluation approach using clustering methods for finding dominant optical flow fields.

2.3. Event Detection. The majority of the methodologies proposed for this category focus on detecting unusual (or abnormal) behavior. This kind of result is relatively sufficient for a video surveillance system. However, labeling events is more pertinent and challenging. Ma et al. [27] model each of the spatiotemporal patches of the scene using dynamic textures. They then apply a suitable distance metric between patches in order to segment the video into spatiotemporal regions showing similar patterns and recognizing activities without explicitly detecting individuals in the scene. While many approaches rely on motion vectors (or optical flow vectors), this approach relies on that dynamic textures show more possibilities. However, they require a lot of processing power and use gray level images which contain less information than a color image.

Kratz and Nishino [28] learn the behavior of extremely crowded scenes by modeling the motion variation of local

space-time volumes and their spatiotemporal statistical behavior. This statistical framework is then used to detect abnormal behavior. Andrade et al. [29, 30] combine Hidden Markov Models, spectral clustering, and principal component analysis of optical flow vectors for detecting crowd emergency scenarios. However, their experiments were carried out on simulated data. Ali and Shah [31] use Lagrangian particle dynamics for the detection of flow instabilities which is an efficient methodology only for the segmentation of high-density crowd flows (marathons, political events, etc.). Li et al. [32] propose a scene segmentation algorithm based on a static model based on a hierarchical pLSA (probabilistic latent semantic analysis) which divides the scene into semantic regions, where each of them consists of an area that contains a set of correlated atomic events. This approach is able to detect static abnormal behaviors in a global context and does not consider the duration of behaviors. Wang et al. [33] model events by grouping low-level motion features into topics using hierarchical Bayesian models. This method processes simple local motion features and ignores global context. Thus, it is well suited for modeling behavior correlations between stationary and moving objects but cannot model complex behaviors that occur on a big area of the scene.

Ihaddadene and Djeraba [34] detect collapsing situations in a crowd scene based on a measure describing the degree of organization or cluttering of the optical flow vectors in the frame. This approach works on unidirectional areas (e.g., elevators). Mehran et al. [35] use a scene structure-based force model in order to detect abnormal behavior. In this force model, an individual, when moving in a particular scene, is subject to the general and local forces that are functions of the layout of that scene and the motional behavior of other individuals in the scene.

Adam et al. [36] detect unusual events by analyzing specified regions on the video sequence called monitors. Each monitor extracts local low-level observations associated with its region. A monitor uses a cyclic buffer in order to calculate the likelihood of the current observation with respect to previous observations. The results from multiple monitors are then integrated in order to alert the user of an abnormal behavior. Wright and Pless [37] determine persistent motion patterns by a global joint distribution of independent local brightness gradient distributions. This huge, random variable is modeled with a Gaussian mixture model. The last approach assumes that all motions in a frame are coherent (e.g., cars); situations in which pedestrians move independently violate these assumptions.

Our approach contributes to the detection of major orientations in complex scenes by building an online probabilistic model of motion orientation on the scene in real-time conditions. The direction model can be considered an extension of the direction map because it captures more than one motion modality at each of the scene's spatial locations. It also contributes to crowd event detection by tracking groups of people as a whole instead of tracking each person individually, which facilitates the detection of crowd events such as merging or splitting.

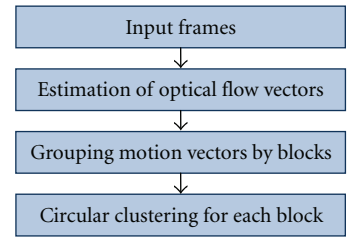


Figure 2: Direction model creation steps.

3. Direction Model

In this section we describe the construction of the direction model. Its purpose is to indicate the tendency of motion direction for each of the scene's spatial locations. We provide an algorithmic overview of the proposed methodology. Its logical blocks are illustrated in Figure 2.

Given a sequence of frames, the main steps involved in the estimation direction model are (i) computation of optical flow between each two successive frames resulting in a set of motion vectors, (ii) grouping of motion vectors in the corresponding block, and (iii) circular clustering of the motion vector orientation in each block. The resulting clusters for each block at the end of the video constitute the direction model. Figure 3 illustrates the three steps.

The direction model creation is an iterative process composed of two stages. The first stage involves the estimation of optical flow vectors. The second one consists of updating the *Direction Model* with the newly obtained data.

3.1. Estimation of the Optical Flow Vectors. In this step, we start by extracting a set of points of interest from each input frame. We consider the Harris corner to be a point of interest [38]. We also consider that, in video surveillance scenes, camera positions and lighting conditions allow a large number of corner features to be captured and tracked easily.

Once we have defined the set of points of interest, we track these points over the next frames using optical flow techniques. For this, we resort to a Kanade-Lucas-Tomasi feature tracker [14, 39] which matches features between two cinsecutive frames. The result is a set of four-dimensional vectors V:

$$V = \{V_1 \cdots V_N \mid V_i = (X_i, Y_i, A_i, M_i)\}, \quad (1)$$

where X_i and Y_i are the image location coordinates of feature i, A_i is the motion direction of feature i, and M_i is the motion magnitude of feature i. It corresponds to the distance between feature i in frame t and its corresponding feature in frame $t + 1$.

This step also allows the removal of static and noise features. Static features move less than a minimum magnitude. By contrast, noise features have magnitudes that exceed the threshold. In our experiments, we set the minimum motion magnitude to 1 pixel per frame and the maximum to 20 pixels per frame.

3.2. Grouping Motion Vectors by Block. The next step consists of grouping motion vectors by blocks. The camera view is

(a) Input frames (b) Optical flow estimation (c) Estimated direction model for the input frames

Figure 3: Representation of the steps involved in the estimation of the direction model for a sequence of frames.

divided into $Bx \times By$ blocks. Each motion vector is attached to the suitable block following its original coordinates. A block will represent the local motion tendency inside that block. Each block is considered to have a square shape and to be of equal size. Smaller block sizes give better results but require a longer processing time.

3.3. Circular Clustering in Each Block. The direction model is an improved direction map [34] that supports multiple orientations at each spatial location. In this section, we present the details of the building of the direction model. For this, we assume for each block the following probabilistic model:

$$p(x \mid \Theta) = \sum_{i=1}^{k} w_i V(x \mid \theta_i), \tag{2}$$

where the parameters are $\Theta = (w_1, \ldots, w_K, \theta_1, \ldots, \theta_K)$ such that $\sum_{i=1}^{K} w_i = 1$. In other words, we assume that we have M mixed *von Mises* densities with K mixing coefficients. We choose $K = 4$ to represent the four cardinal points. $V(x, \theta_i)$ is the von Mises distribution defined by the following probability density function:

$$V(x \mid \theta_i) = \frac{1}{2\pi I_0(m_i)} \exp[m_i \cos(x - \mu_i)]; \quad 0 < x < 2\pi, \quad 0 < \mu_i < 2\pi, \quad m_i > 0, \tag{3}$$

where $\theta_i = (\mu_i, k_i)$, μ_i are the parameters of the ith distribution. μ_i is its mean orientation, m_i is its dispersion parameter, and $I_0(m)$ is the modified Bessel function of the first kind and order 0 defined by

$$I_0(m) = \sum_{r=0}^{\infty} \left(\frac{1}{r!}\right)^2 \left(\frac{1}{2}m\right)^{2r}. \tag{4}$$

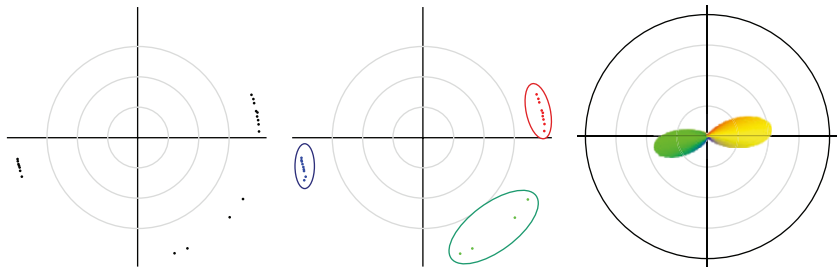
With each new frame, the values of $\Theta = (w_1, \ldots, w_K, \theta_1, \ldots, \theta_K)$ are updated with the new vector set using circular clustering. Instead of using an exact EM algorithm over circular data, we perform an online K-means approximation described in [11] which is originally used for building a mixture of Gaussian distribution. The algorithm is adapted to deal with circular data and considers the inverse of the variance as the dispersion parameter; $m = 1/\sigma^2$. Figure 4 shows the cluster thus obtained and the corresponding distribution's probability density.

The direction model is made up of the whole mixture distribution as estimated for each of the scene's blocks.

4. Detecting Motion Patterns

Given an input video, we compute its direction model which estimates for each block up to K major orientations. In other words, dominant motion orientations are learned at each block (or spatial location). Since motion patterns are the regions of the scene that share the same motion orientation behavior, thus, motion pattern detection can be formulated as a problem of clustering the blocks of the direction model (a motion pattern can be considered as a cluster). We refer to gestaltism in order to find grouping factors such as proximity, similarity, closure, simplicity, and common fate. We then detect the scene's dominant motion patterns by applying a peculiar bottom-up region-based segmentation algorithm to the direction model's blocks. Figure 5 shows the output of our algorithm on a 3×3 direction model with $K = 4$. We can see that neighbor blocks that have similar orientations appear in the same motion pattern. We can also note that traditional clustering algorithms cannot be applied here because a block can be in different motion patterns (cluster) at the same time. This situation happens frequently in real life such as zebra crossing and shop entrances. In addition, since we are processing circular data, the formulas need to be adapted to deal with the equality between 0 and 2π.

We propose a motion patterns extraction algorithm that deals with circular data. Another peculiarity of our algorithm is that it allows a block to be in different motion patterns; more specifically, a block can be in maximum of K clusters. This is done by considering two neighboring blocks in the same cluster if they have at least two similar orientations. In other words, at least one of the K major orientations at the first block has to be similar to at least one of the K major orientations of the second block. This is achieved by storing for each block the corresponding cluster for each dominant orientation. We use a 3D matrix with dimensions $Bx \times By \times K$



(a) Input data (b) Estimated clusters (c) Probability density around the unit circle

Figure 4: Representation of estimated clusters and density of the input data.

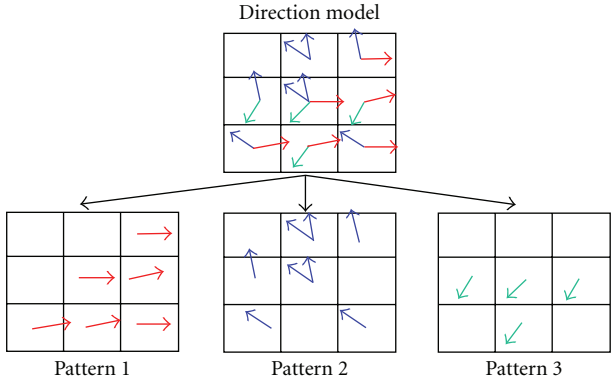


Figure 5: Motion pattern detection from a 3×3 direction model.

and each element of that matrix will be affected by a cluster "id".

The full algorithm is provided for clarification in Algorithm 1 and works as follows: a direction model D that has $Bx \times By$ mixtures of K von Mises distributions and as its input and outputs a set of clusters C. We simplify the notation by introducing a 3D matrix μ with size $Bx \times By \times K$ containing only the mean orientations of the direction model. Thus, an element $\mu(i, j, l)$ contains the mean orientation of the lth von Mises distribution of the direction model block at position (i, j). Next, the algorithm initializes a $Bx \times By \times K$ 3D matrix M used to store the different cluster "id"s associated to the blocks. The next step consists of affecting the blocks to the corresponding regions, which is an iterative procedure. The algorithm uses 1-block neighboring and uses the similarity test explained earlier. The similarity condition between two orientations is satisfied if their difference is less than a threshold α. Experiments have demonstrated that a value of $\alpha = \pi/4$ gives the best balance between the algorithm's efficiency and effectiveness.

5. Event Detection in Crowd Scenes

Our proposed method for event detection is based on the analysis of groups of people rather than individual persons. The targeted events occurring in groups of people are walking, running, splitting, merging, local dispersion, and evacuation.

The proposed algorithm is composed of several steps (Figure 6): it starts by building direction and magnitude models. After that, the block clustering step groups together neighboring blocks that have a similar orientation and magnitude. These groups are tracked over the next frames. Finally, the events are detected by using information from group tracking, the magnitude model, and the direction model.

5.1. Direction and Magnitude Model. In this application, we are interested in real-time detection and group-tracking. Thus, for each frame we build a direction model which is called an instantaneous direction model. The steps involved in the estimation of the direction model are explained in Section 3.

The magnitude model is built using an online mixture of one-dimensional Gaussian distributions over the mean motion magnitude of a frame, given by

$$P(x) = \sum_{k=1}^{4} \omega_k \frac{1}{\sigma_k \sqrt{2\pi}} \exp\left(-\frac{(x-\mu_k)^2}{2\sigma_k^2}\right), \tag{5}$$

where ω_k, μ_k, and σ_k are, respectively, the weight, mean, and variance of the kth Gaussian which are learned from short sequences of walking persons. Hence, this magnitude model learns the walking speed of the crowd.

5.2. Block Clustering. In this step, we gather similar blocks to obtain block clusters. The idea is to represent a group of people moving in the same direction at the same speed by the same block cluster. By "similar", we mean same direction, same speed, and neighboring locations. Each block $B_{x,y}$ is defined by its position $P_{x,y} = (x, y)$; $x = \overline{1 \cdots Bx}, y = \overline{1 \cdots By}$, and orientation $\Omega_{x,y} = \mu_{0,x,y}$ (see Section 5.1).

The merging condition consists of a similarity measure D_Ω between two blocks $B_{x1,y1}$ and $B_{x2,y2}$ defined as

$$D_\Omega\left(\Omega_{x1,y1}, \Omega_{x2,y2}\right) = \min_{k,z} \left| \left(\Omega_{x1,y1} + 2k\pi\right) - \left(\Omega_{x2,y2} + 2z\pi\right) \right|,$$
$$(k, z) \in Z^2, \quad 0 \le D_\Omega\left(\Omega_{x1,y1}, \Omega_{x2,y2}\right) < \pi. \tag{6}$$

input Direction model D that contains $Bx \times By$ mixtures of K vM distributions
return Set of clusters C
Create a $Bx \times By \times K$ 3D matrix M. $M(i, j, l)$ stores the cluster id of the corresponding element
Create a $Bx \times By \times K$ 3D matrix μ and initialize $\mu(i, j, l)$ with the mean orientation of the lth vM distribution of the block at position (i, j)
$C \leftarrow \emptyset$
$n \leftarrow 0$
$M \leftarrow 0$
for $i = 1$ to Bx
  for $j = 1$ to By
    for $l = 1$ to K
      if $M(i, j, l) = 0$
        $n \leftarrow n + 1$
        create new cluster c
        put element (i, j, l) with orientation $\mu_{i,j,l}$ in c and update c
        $C \leftarrow C \cup c$
        $B \leftarrow neighborList(i, j, l, M)$
        $M(i, j, l) = n$
        for each b in B
          if $c.metric - \mu(b \cdot x, b \cdot y, b \cdot k) \leq \alpha$
            $M(i, j, l) = n$
            put element $(b \cdot i, b \cdot \ j, b \cdot l)$ with orientation $\mu_{b \cdot x, b \cdot y, b \cdot k}$ in c and update c
            $B \leftarrow B \cup neighborList(b \cdot x, b \cdot y, b \cdot k, M)$

Algorithm 1: Motion pattern detection.

Considering theses definitions, two neighboring blocks $B_{x1,y1}$ and $B_{x2,y2}$ are in the same cluster if

$$D_\Omega\left(\Omega_{x1,y1}, \Omega_{x2,y2}\right) < \delta_\Omega, \quad 0 \leq \delta_\Omega < \pi, \tag{7}$$

where δ_Ω is a predefined threshold. In our implementation, we choose $\delta_\Omega = \pi/10$ empirically. Figure 7 shows a sample output of the process.

The mean orientation X_j of the cluster C_j is given by the following formula [40]:

$$X_j = \arctan \frac{\sum_{x=1}^{Bx} \sum_{y=1}^{By} \mathbb{1}_{C_j}\left(B_{x,y}\right) \cdot \sin\left(\Omega_{x,y}\right)}{\sum_{x=1}^{Bx} \sum_{y=1}^{By} \mathbb{1}_{C_j}\left(B_{x,y}\right) \cdot \cos\left(\Omega_{x,y}\right)}, \tag{8}$$

where $\mathbb{1}_{C_j}$ is the indicator function.

The centroid $O_j = (ox_j, oy_j)$ of the group C_j is defined by

$$ox_j = \frac{\sum_{x=1}^{Bx} \sum_{y=1}^{Bx} \mathbb{1}_{C_j}\left(B_{x,y}\right) \cdot x_i}{\sum_{x=1}^{Bx} \sum_{y=1}^{By} \mathbb{1}_{C_j}\left(B_{x,y}\right)}, \tag{9}$$

and we obtain oy_j by analogy.

5.3. Group Tracking. When the groups have been built, they are tracked in the next frames. The tracking is done by matching the centroids of the groups in a frame f with the centroids of the frame $f + 1$. Each frame f is defined by its groups $\{C_{1,f}, C_{2,f}, \ldots, C_{n_f,f}\}$ where n_f is the number of groups detected in frame f. Each group $C_{i,f}$ is described by its centroid $O_{i,f}$ and mean orientation $X_{i,f}$. The group $C_{m,f+1}$

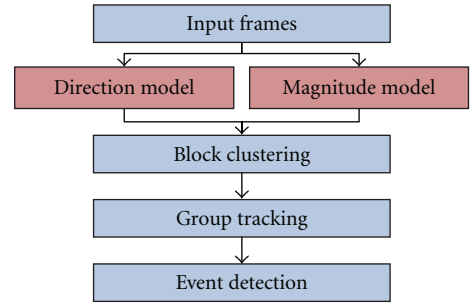


Figure 6: Algorithm steps.

that matches the group $C_{i,f}$ must have the closest centroid to $C_{i,f}$ and has to be in a minimal area around it. In other words, it has to satisfy these two conditions:

$$m = \underset{j}{\operatorname{argmin}}\left(D\left(O_{i,f}, O_{j,f+1}\right)\right),$$
$$D\left(O_{i,f}, O_{m,f+1}\right) < \tau, \tag{10}$$

where τ is the minimal distance between two centroids (we choose $\tau = 5$). If there is no matching (meaning no group $C_{m,f+1}$ meeting these two conditions), then group $C_{i,f}$ disappears and is no longer tracked in the next frames.

5.4. Event Recognition. The targeted events are classified into three categories.

(i) Motion speed-related events: they can be detected by exploiting the motion velocity of the optical flow

(a) Motion detection (b) Estimated direction model (c) Detected groups

Figure 7: Group clustering on a frame.

vectors across frames (e.g., running and walking events).

(ii) Crowd convergence events: they occur when 2 or more groups of people get near to each other and merge into a single group (e.g., crowd merging event).

(iii) Crowd divergence events: they occur when the persons move in opposite directions (e.g., local dispersion, splitting, and evacuation events).

The events from the first category are detected by fitting each frame's mean optical flow magnitude against a model of the scene's motion magnitude. The events from the second and third categories are detected by analyzing crowd's orientation, distance, and position. If two groups of people go to the same area, it is called "convergence". However, if they take different directions, it is called "divergence". In the following, we will give a more detailed explanation of the adopted approaches.

5.4.1. Running and Walking Events. As described earlier, the main idea is to fit the mean motion velocity between two consecutive frames against the magnitude model of the scene. It gives a probability for running P_{run}, walking P_{walk}, and stopping P_{stop} events. As motion flows are processed in this paper, $P_{\text{stop}} = 0$ and $P_{\text{run}} = 1 - P_{\text{walk}}$.

Since a person has more chances of staying in his current state rather than moving suddenly to the other state (e.g., a walking person increases his/her speed gradually until he/she starts running), then the final running or walking probability is a weighted sum of the current and previous probabilities. The result is compared against a threshold to infer a walking or a running event. Formally, a frame f with mean motion magnitude m_f contains a walking (resp., running) event if

$$\sum_{l=f-h}^{f} w_{f-l} \cdot P_{\text{walk}}(m_l) > \vartheta_{\text{walk}}, \tag{11}$$

where ϑ_{walk} (resp., ϑ_{run}) is the walking (resp., running) threshold. h is the number of previous frames to consider. Each previous state has a weight w_l (in our implementation, we choose $h = 1$, $w_0 = 0.8$, and $w_1 = 0.2$). $P_{\text{walk}}(m_l)$ is the probability of observing m_l. It is obtained by fitting m_l against the magnitude model (see Section 5.1) using formula (5). This probability is thresholded to detect a walking (resp., running) event. We choose a threshold of 0.05 for the walking event, and 0.95 for the running event, since there is 95% probability for a value to be comprised between $\mu - 2\sigma$ and $\mu + 2\sigma$ where μ and σ are, respectively, the mean and the standard deviation of the Gaussian distribution.

5.4.2. Crowd Convergence and Divergence Events. Convergence and divergence events are first detected by computing the circular variance $S_{0,f}$ of the groups of each frame f given the following equation [40]:

$$S_{0,f} = 1 - \frac{1}{n_f} \sum_{i=1}^{n_f} \cos\left(X_{i,f} - \overline{X_{0,f}}\right), \tag{12}$$

where $\overline{X_{0,f}}$ is the mean angle of the clusters in frame f defined by

$$\overline{X_{0,f}} = \arctan \frac{\sum_{i=1}^{n_f} \sin\left(X_{i,f}\right)}{\sum_{i=1}^{n_f} \cos\left(X_{i,f}\right)}. \tag{13}$$

$S_{0,f}$ is a value between 0 and 1 inclusive. If the angles are identical, $S_{0,f}$ will be equal to 0. A set of perfectly opposing angles will give a value of 1. If the circular variance exceeds a threshold β (we choose $\beta = 0.3$ in our implementation), we can infer the realization of convergence and/or divergence events. We examine the position and direction of each group in relation with the other groups in order to decide which event happened. If two groups are oriented towards the same direction and are close to each other, then it is a convergence (Figure 8). However, if they are going in opposite directions and are close to each other, then it is a divergence.

More formally, let $\overrightarrow{v_{i,f}}$ be a vector representing a group $C_{i,f}$ at frame f. $\overrightarrow{v_{i,f}}$ is characterized by an origin $O_{i,f}$ which is the centroid of the group $C_{i,f}$, an orientation Ω_i, and a destination $Q_{i,f}$ whose coordinates $qx_{i,f}, qy_{i,f}$ are defined as

$$\begin{aligned} qx_{i,f} &= ox_{i,f} \cdot \cos(\Omega_i), \\ qy_{i,f} &= oy_{i,f} \cdot \sin(\Omega_i). \end{aligned} \tag{14}$$

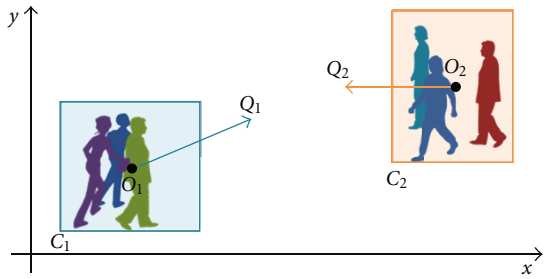


Figure 8: Merging groups.

Two groups are converging (or merging) if the two following conditions are satisfied:

$$D\left(O_i, O_j\right) > D\left(Q_i, Q_j\right),$$
$$D\left(O_i, O_j\right) < \delta, \tag{15}$$

where $D(P, Q)$ is the Euclidean distance between points P and Q, and δ represents the minimal distance required between two groups' centroids (we took $\delta = 10$ in our experiments). Figure 8 shows a representation of two groups participating in a merging event.

Similarly, two groups are diverging if the following conditions are satisfied:

$$D\left(O_i, O_j\right) < D\left(Q_i, Q_j\right),$$
$$D\left(O_i, O_j\right) < \delta. \tag{16}$$

However, in this situation, we distinguish three cases.

(1) The groups do not stay separated for a long time and have a very short motion period; so they are still forming a group. This corresponds to the local dispersion event.

(2) The groups stay separated for a long time and their distance grows over the frames. This corresponds to the crowd splitting event.

(3) If the first situation occurs while the crowd is running, this corresponds to an evacuation event.

To detect the events described above, we add another feature to each group $C_{i,f}$ which corresponds to its "age", represented by the first frame where the group appeared, noted by $F_{i,f}$. There is a local dispersion at frame f between two groups $C_{i,f}$ and $C_{j,f}$ if the conditions in (16) are satisfied. Besides, their motion has to be recent:

$$f - F_{i,f} < \nu,$$
$$f - F_{j,f} < \nu, \tag{17}$$

where ν is a threshold representing the number of frames since the groups have started moving (because group clustering relies on motion). In our implementation, it is equal to 28, which corresponds to 4 seconds in a 7 fps video stream.

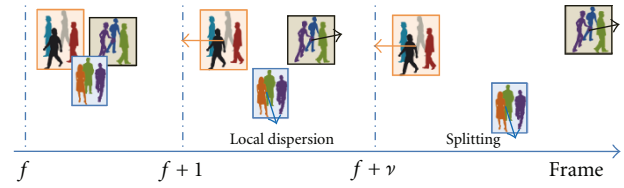


Figure 9: Representation of local dispersion and splitting events.

Figure 10: Representation of an evacuation event.

Two groups $C_{i,f}$ and $C_{j,f}$ are splitting at frame f, if they satisfy the conditions (16). Moreover, at least one of them has a less recent motion:

$$f - F_{i,f} \geq \nu \quad \text{or} \quad f - F_{j,f} \geq \nu. \tag{18}$$

The evolution of the group separation over time from the local dispersion to the splitting event is illustrated in Figure 9.

There is an evacuation event between two groups $C_{i,f}$ and $C_{j,f}$ at frame f if they satisfy the local dispersion conditions (16) and (17) as well as the running conditions (11). Figure 10 shows a representation of two groups participating in an evacuation event.

The probabilities of merging, splitting, local dispersion, and evacuating events noted, respectively, by $P\text{merge}_f$, $P\text{split}_f$, $P\text{disp}_f$, and $P\text{evac}_f$ are null if the circular variance is less than the threshold, since the events are triggered only if the circular variance is greater than the threshold. In that case, merging, splitting, and dispersion probabilities are calculated by dividing the number of times the event occurred in a frame by the total number of times those three events occurred in the same frame. Let $N\text{megre}_f$, $N\text{split}_f$, and $N\text{disp}_f$ be the number of times that merging, splitting, and local dispersion, respectively, occurred between the segments in frame f. Then the merging probability for frame f is given by

$$P\text{merge}_f = \frac{N\text{merge}_f}{N\text{merge}_f + N\text{split}_f + N\text{disp}_f}. \tag{19}$$

We obtain $P\text{split}_f$ and $P\text{disp}_f$ by analogy; for example, $P\text{disp}_f$ is defined by this formula:

$$P\text{disp}_f = \frac{N\text{disp}_f}{N\text{merge}_f + N\text{split}_f + N\text{disp}_f}. \tag{20}$$

Since an event is what catches a user's attention, we consider that the most frequent events in a frame are the ones that characterize it. Thus, we considered a threshold of 1/3 for each event. This approach enables multiple events to occur for each frame but only keeps the most noticeable ones.

Finally, the evacuation event probability at frame f, noted by $P\text{evac}_f$, is a particular case because it is conditioned by the running event in addition to the local dispersion event. Therefore, if there is a running event in frame f (see Section 5.4.1), then $P\text{disp}_f$ is replaced by $P\text{evac}_f$ in formula (20), and $N\text{disp}_f$ is replaced by $N\text{evac}_f$. $P\text{disp}_f$ and $N\text{disp}_f$ are then equal to zero. If there is no running event in frame f, $P\text{evac}_f$ is null. The evacuation event threshold for each frame is also 1/3.

5.5. Event Detection Using a Classifier. We propose a methodology to detect the described events using a classifier. This is performed by using two classifiers, a first one for detecting motion-speed-related events and a second one for detecting crowd convergence and divergence events. Although this double labeling has the drawback of double processing, this is a more natural representation since we permit overlapping between events of different categories. For example, running and merging events can occur at the same frame. Another solution is to use a different classifier for each event. However, this solution is time-consuming and further processing needs to be performed in the case of an overlapping event between the merging and splitting events, for example.

Each classifier is trained by a set of features vectors where each one is estimated at each frame. Thus a classifier can classify an event for a frame given its feature vector. We use the running probability defined in Section 5.4.1 as a feature for the motion speed-related events classifier. The crowd convergence and divergence events classifier uses more features which are the running probability, the number of groups, their mean distance, their mean direction, and their circular variance.

6. Experiments

We show the experiments and the results of our approach in this section. We first focus on the motion pattern extraction experiments using videos from well-known datasets. After that, we experiment the crowd event detection approach using the PETS dataset.

6.1. Motion Pattern Extraction Results. The approach was experimented in various videos retrieved from different fields. The sequences have different complexities. They range from the simple case of structured crowd scenes where the objects behave in the same manner to the complex case of unstructured crowd scenes where different motion patterns can occur at the same location on the image plane. To process a video sequence, we estimate its optical flow vectors in order to build a direction model. The motion pattern extraction is then run on that direction model.

Our approach was first experimented in an urban environment where vehicles and pedestrians use the same road (Figure 11). The sequence was retrieved from the AVSS 2007 dataset (http://www.elec.qmul.ac.uk/staffinfo/andrea/avss2007_d.html); it has a resolution of 720×576 pixels with a sampling rate of 25 Hz. It consists of a two-way road, the traffic flow being on the left side of the road. Vehicles operate on the road and some pedestrians cross it. The proposed approach retrieved the car patterns successfully by retrieving two classes for the traffic flow and a third direction for cars turning left. In addition, it also retrieved the pedestrians' patterns at the bottom of the scene. The advantage of affecting multiple clusters to a single block can be noted in comparison with other approaches where a unique orientation is assumed for each location in the scene.

Figure 12 shows a crowd performing a pilgrimage. In this video, a huge amount of people browse the area in different directions. However, our algorithm detects two major motion patterns despite the complexity of the sequence. This is explained by research in collective intelligence which states that moving organisms generate patterns over time and a certain order is generated instead of chaos.

We compare our approach to [18] which proposed a motion pattern extraction method by clustering the motion field. We show its results to the "Motion Field approach" using the Hadjee sequence in Figure 13, where we see that our approach has better results. In fact, our methodology supports the overlapping of motion patterns as opposed to [18] where the brown and orange patterns did not overlap. We also remark that the "Motion Field approach" detects less motion at the top of the frame because it uses a preprocessing step which may eliminate useful motion information.

Next, we show the results of our approach using a complex scene with both cars and people moving as illustrated in Figure 14. These sequences are retrieved from the Getty-images (http://www.gettyimages.com/) website. It contains three two-way roads on the left, middle, and right parts of the sequence, respectively. In addition, there are two long zebras that cross the roads. We detected most of the motion patterns which are illustrated in Figure 14(b). However, in the areas where the optical flow vectors are not precisely estimated, we could not detect the motion patterns such as the zebra crossing at the back of the scene.

We show more results of our approach using various video sequences in Figures 15 and 16. They are retrieved from video search engines, CAVIAR dataset, and Getty-images website. The sequences are characterized by a high density of moving objects.

Finally, we synthesize the results of our experiments in Table 1 which compares the number of detected motion patterns with the ground truth. We provide the original file names of the sequences. Note that providing only the number of motion patterns is insufficient, and we must also provide an illustration of the detected motion patterns for each sequence. Nevertheless, the evaluation of a motion pattern extraction approach remains subjective and different appreciations may be made for the same video. However, we believe that our approach provides satisfying results given the complexity of the sequences.

(a) (b)

Figure 11: Major motion patterns in an urban scene.

(a) Original frame (b) Detected motion patterns

Figure 12: Detected motion patterns in a pilgrimage sequence.

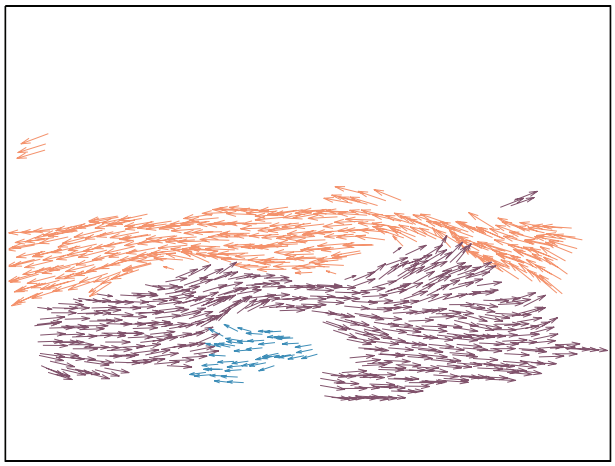


Figure 13: Detected motion patterns in a pilgrimage sequence using the Motion Flow Field [18].

6.2. Event Detection. The approach described in the previous sections has been evaluated in the PETS'2009 datasets. This dataset includes multisensor sequences containing different crowd activities. Several scenarios involving crowd density estimation, crowd counting, single person tracking, flow analysis, and high-level event detection are proposed. Also, a set of training data is available.

Table 1: Comparison results between our approach and the ground truth.

Video	Detected	Ground truth
AVSS_PV_Hard	4	4
pilgrimage1	2	2
pilgrimage2	3	2
Caviar_Browse4	4	4
Caviar_EnterExitCrossingPath	3	3
Getty-341-46_l	3	2
Getty-81059009_q	5	5

For this paper, we processed the event recognition sequences which are organized in the S3 dataset. The algorithm processed five 720×576 video streams at a speed of 4 frames/second on an Intel Celeron 1.8 GHZ. We used a block size of 15 pixels which is the best balance between efficiency and effectiveness.

In our experiments, we collected the 1000 frames of the dataset and we annotated them with two labels. The first one is either running or walking. The second one is split, local dispersion, merge, evacuation, or regular. Figure 18 illustrates each event in a separate image. The local dispersion event is represented in Figure 18(e) by a pink line that links the corresponding groups, merging is represented

(a) Original frame (b) Motion patterns

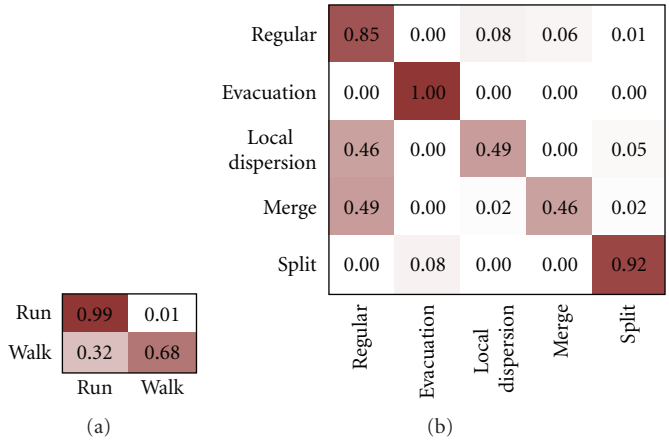
Figure 14: Detected motion patterns in complex sequence with moving cars and people.

(a) Original frame (b) Motion patterns

Figure 15: Detected motion patterns in another pilgrimage sequence.

(a) Original frame (b) Motion patterns

Figure 16: Detected motion patterns at the bottom of an escalator.



(a) (b)

Figure 17: Confusion matrices obtained using random forest classifier: (a) running and walking events, and (b) splitting, merging, evacuation, and local dispersion events.

(a) Walking event (b) Running event (c) Merging event

(d) Splitting event (e) Local dispersion event (f) Evacuation event

Figure 18: Event detection samples. The numbers represent the probabilities of the events. Detected events are colored in blue.

Table 2: Comparison of event detection results. NP means that the result was not provided.

	Approach	Manual parameters	Random forest	Statistical filters [41]	Holistic properties [42]
Walking	Precision	0.97	0.96	—	0.87
	Recall	0.96	0.99	—	NP
Running	Precision	0.75	0.86	0.99	0.75
	Recall	0.81	0.68	0.99	NP
Regular	Precision	—	0.78	0.99	—
	Recall	—	0.85	0.86	—
Evacuation	Precision	0.69	0.83	—	0.94
	Recall	0.82	1	—	NP
Local disp.	Precision	0.67	0.58	—	0.8
	Recall	0.45	0.48	—	NP
Merge	Precision	0.59	0.65	—	0.68
	Recall	0.45	0.46	—	NP
Split	Precision	0.47	0.73	0.65	0.74
	Recall	0.47	0.92	1	NP

in Figure 18(c) by a yellow line, and splitting is represented in Figure 18(d) by a white line.

Since each frame has two labels, two classifiers are necessary. This has the drawback of increasing the computation costs. However, we are able to detect two categories of events such as the frames where the running and merging events occur. The dataset was split into a training set (75%) and a testing set (25%). For both categories of events, the random forest classifier performed best. We show the confusion matrices in Figure 17.

We assemble the results obtained using manual parameters and classifiers as well as the results of other approaches in Table 2. It shows the precision and the recall, if available, for each event. We note that our approach (manual parameters or classifiers) is the only one which is able to detect all of the events. The approach using Statistical filters detects only 3 events and the Holistic Properties approach which does not consider the regular event (which is confused with walking).

The statistical filters approach was designed to detect "abnormal behavior" by using the unusual flow and unusual magnitude features. These features can only detect three categories of events (regular, split, and running). However, the authors claim that their approach is able to detect other events by plugging other features. Unfortunately, no more details are provided on how to plug other features. In addition, we believe that the features modelling all types of

behaviors are better than features modelling only abnormal behaviors. Table 2 shows that our approach has the same results as the approach using statistical filters and has also the advantage of detecting more events "out of the box".

The results of our approach are very close to the Holistic properties approach [42]. However, this approach is slower than ours and does not permit the overlapping of events, which means that we cannot have walking and merging events at the same instant.

7. Conclusion

We have presented an automatic visual surveillance system able to detect major motion patterns and events in crowd scenes. It bypasses time-consuming methods such as background subtraction and person detection and rather resorts to global motion information obtained from optical flow vectors to model the motion magnitude and velocity at each spatial location of the scene. These models use mixture distributions estimated via online algorithms in order to capture multimodal crowd motion over time. Motion patterns are then detected by applying a region-based segmentation algorithm to the direction model of a video stream. Crowd events are detected by analyzing the behavior of the groups in terms of motion direction and velocity.

We demonstrate the performance of our approach using multiple datasets. These experiments show that our approach is applicable to a wide range of scenes which consist of low and high crowd density scenes as well as structured and unstructured (i.e., the motion of crowd at any location is multimodal) scenes. In addition, the system detects groups of people even in the presence of occlusions, which then facilitates the detection of group-related events such as merging or splitting.

In the future, we plan to address some specific problems in order to improve the results like, for instance, performing a finer analysis of the notion of block, adjusting the size of the blocks to the spatiotemporal motion features, or adopting a multiscale approach. Besides, we plan to extend the research domains of our system. More precisely, we will use detected motion patterns as a prerequisite for tracking single persons and detecting abnormal behaviors. Furthermore, we will label the video streams using semantic information retrieved from the event detection module in order to add indexing/retrieval capabilities to our system.

Acknowledgment

This work has been supported by the European Commission within the Information Society Technologies program (FP6-IST-5-0033715), through the project *MIAUCE* (http://www.miauce.org/) and the French ANR project *CAnADA*.

References

[1] M. L. Shyu, Z. Xie, and M. Chen, "Video semantic event/concept detection using a subspace-based multimedia datamining framework," *IEEE Transactions on Multimedia*, vol. 10, pp. 252–259, 2008.

[2] R. Sternberg, *Cognitive Psychology*, Thomson Wadsworth, 3rd edition, 2003.

[3] C. Bahlmann, "Directional features in online handwriting recognition," *Pattern Recognition*, vol. 39, no. 1, pp. 115–125, 2006.

[4] T. C. Robert, A. J. Lipton, and T. Kanade, "Introduction to the special section on video surveillance," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 745–746, 2000.

[5] B. Zhan, D. N. Monekosso, P. Remagnino, S. A. Velastin, and LI. Q. Xu, "Crowd analysis: A survey," *Machine Vision and Applications*, vol. 19, no. 5-6, pp. 345–357, 2008.

[6] A. Baumann, M. Boltz, J. Ebling et al., "A review and comparison of measures for automatic video surveillance systems," *Eurasip Journal on Image and Video Processing*, vol. 2008, Article ID 824726, 30 pages, 2008.

[7] W. Hu, T. Tan, L. Wang, and S. Maybank, "A survey on visual surveillance of object motion and behaviors," *IEEE Transactions on Systems, Man and Cybernetics C*, vol. 34, no. 3, pp. 334–352, 2004.

[8] B. T. Morris and M. M. Trivedi, "A survey of vision-based trajectory learning and analysis for surveillance," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 8, Article ID 4543858, pp. 1114–1127, 2008.

[9] L. Li, W. Huang, I. Y. H. Gu, and Q. Tian, "Foreground object detection from videos containing complex background," in *Proceedings of the eleventh ACM international conference on Multimedia (MULTIMEDIA '03)*, pp. 2–10, ACM, New York, NY, USA, 2003.

[10] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '99)*, vol. 2, pp. 246–252, June 1999.

[11] P. Kaewtrakulpong and R. Bowden, "An improved adaptive background mixture model for realtime tracking with shadow detection," in *Proceedings of the 2nd European Workshop on Advanced Video Based Surveillance Systems*, 2001.

[12] S. Birchfield, "Klt: an implementation of the kanade-lucas-tomasi feature tracker," 2006.

[13] J.-Y. Bouguet, "Pyramidal implementation of the lucas kanade feature tracker description of the algorithm," 2000.

[14] J. Shi and C. Tomasi, "Good features to track," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '94)*, pp. 593–600, June 1994.

[15] W. Hu, X. Xiao, Z. Fu, D. Xie, T. Tan, and S. Maybank, "A system for learning statistical motion patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 9, pp. 1450–1464, 2006.

[16] X. Wang, K. Tieu, and E. Grimson, "Learning semantic scene models by trajectory analysis," in *Proceedings of the 9th European Conference on Computer Vision (ECCV '06)*, vol. 3953 of *Lecture Notes in Computer Science*, 2006.

[17] A. Basharat, A. Gritai, and M. Shah, "Learning object motion patterns for anomaly detection and improved object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '08)*, pp. 1–8, June 2008.

[18] M. Hu, S. Ali, and M. Shah, "Learning motion patterns in crowded scenes using motion flow field," in *Proceedings of the 19th International Conference on Pattern Recognition (ICPR '08)*, December 2008.

[19] T. Zhang, H. Lu, and S. Z. Li, "Learning semantic scene models by object classification and trajectory clustering," in

Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPR '09), pp. 1940–1947, June 2009.

[20] Q. Yu and G. Medioni, "Motion pattern interpretation and detection for tracking moving vehicles in airborne video," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPR '09)*, pp. 2671–2678, June 2009.

[21] D. Lin, E. Grimson, and J. Fisher, "Learning visual flows: a lie algebraic approach," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPR '09)*, pp. 747–754, June 2009.

[22] J. M. Gryn, R. P. Wildes, and J. K. Tsotsos, "Detecting motion patterns via direction maps with application to surveillance," *Computer Vision and Image Understanding*, vol. 113, no. 2, pp. 291–307, 2009.

[23] M. Rodriguez, S. Ali, and T. Kanade, "Tracking in unstructured crowded scenes," in *Proceedings of the 12th International Conference on Computer Vision (ICCV '09)*, pp. 1389–1396, Kyoto, Japan, October 2009.

[24] C. Stauffer and W. E. L. Grimson, "Learning patterns of activity using real-time tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 747–757, 2000.

[25] B. Morris and M. Trivedi, "Learning trajectory patterns by clustering: experimental studies and comparative evaluation," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPR '09)*, pp. 312–319, June 2009.

[26] G. Eibl and N. Brändle, "Evaluation of clustering methods for finding dominant optical flow fields in crowded scenes," in *Proceedings of the 19th International Conference on Pattern Recognition (ICPR '08)*, December 2008.

[27] Y. Ma, P. Cisar, and A. Kembhavi, "Motion segmentation and activity representation in crowds," *International Journal of Imaging Systems and Technology*, vol. 19, no. 2, pp. 80–90, 2009.

[28] L. Kratz and KO. Nishino, "Anomaly detection in extremely crowded scenes using spatio-temporal motion pattern models," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPR '09)*, pp. 1446–1453, June 2009.

[29] E. L. Andrade, S. Blunsden, and R. B. Fisher, "Hidden Markov Models for optical flow analysis in crowds," in *Proceedings of the 18th International Conference on Pattern Recognition (ICPR '06)*, vol. 1, pp. 460–463, August 2006.

[30] E. L. Andrade, S. Blunsden, and R. B. Fisher, "Modelling crowd scenes for event detection," in *Proceedings of the 18th International Conference on Pattern Recognition (ICPR '06)*, pp. 175–178, Washington, DC, USA, August 2006.

[31] S. Ali and M. Shah, "A Lagrangian particle dynamics approach for crowd flow segmentation and stability analysis," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '07)*, pp. 1–6, June 2007.

[32] J. Li, S. Gong, and T. Xiang, "Scene segmentation for behaviour correlation," in *Proceedings of the European Conference on Computer Vision (ECCV '08)*, 2008.

[33] X. Wang, X. Ma, and W. E. Grimson, "Unsupervised activity perception in crowded and complicated scenes using hierarchical bayesian models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 3, pp. 539–555, 2009.

[34] N. Ihaddadene and C. Djeraba, "Real-time crowd motion analysis," in *Proceedings of the 19th International Conference on Pattern Recognition (ICPR '08)*, December 2008.

[35] R. Mehran, A. Oyama, and M. Shah, "Abnormal crowd behavior detection using social force model," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPR '09)*, pp. 935–942, June 2009.

[36] A. Adam, E. Rivlin, I. Shimshoni, and D. Reinitz, "Robust real-time unusual event detection using multiple fixed-location monitors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 3, pp. 555–560, 2008.

[37] J. Wright and R. Pless, "Analysis of persistent motion patterns using the 3D structure tensor," in *Proceedings of the IEEE Workshop on Motion and Video Computing (WMVC '05)*, vol. 2, pp. 14–19, Washington, DC, USA, January 2005.

[38] C. Harris and M. J. Stephens, "A combined corner and edge detector," in *Proceedings of the Alvey Vision Conference*, pp. 147–152, Washington, DC, USA, 1988.

[39] B. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 674–679, Washington, DC, USA, 1981.

[40] L. Gary, L. Gaile, and E. B. James, *Directional Statistics, Concepts and Techniques in Modern Geography*, Geo Abstracts, Norwich, UK, 1980.

[41] A. Kiss, A. Utasi, and T. Sziranyi, "Statistical filters for crowd image analysis," in *Proceedings of the 11th IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS '09)*, 2009.

[42] A. B. Chan, M. Morrow, and N. Vasconcelos, "Analysis of crowded scenes using holistic properties," in *Proceedings of the 11th IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS '09)*, 2009.