

Visual Event Recognition using Decision Trees

Cedric Simon · Jerome Meessen · Christophe
De Vleeschouwer

Received: date / Accepted: date

Abstract This paper presents a classifier-based approach to recognize dynamic events in video surveillance sequences. The goal of this work is to propose a flexible event recognition system that can be used without relying on a long-term explicit tracking procedure. It is composed of three stages. The first one aims at defining and building a set of relevant features describing the shape and movements of the foreground objects in the scene. To this aim, we introduce new motion descriptors based on space-time volumes. Second, an unsupervised learning-based method is used to cluster the objects, thereby defining a set of coarse to fine local patterns of features, representing primitive events in the video sequences. Finally, events are modeled as a spatio-temporal organization of patterns based on an ensemble of randomized trees. In particular, we want this classifier to discover the temporal and causal correlations between the most discriminative patterns. Our system is experimented and validated both on simulated and real-life data.

Keywords Randomized Decision Trees · Automated Visual Surveillance System · Activity Recognition

Cedric Simon
UCL, Communication and Remote Sensing Lab, Louvain-la-Neuve, Belgium
Tel.: +32 10 47 80 72
Fax: +32 10 47 20 89
E-mail: cedric.simon@uclouvain.be

Jerome Meessen
Multitel, Mons, Belgium
Tel.: +32 65 374709
Fax: +32 65 374729
E-mail: jerome.meessen@multitel.be

Christophe De Vleeschouwer
UCL, Communication and Remote Sensing Lab, Louvain-la-Neuve, Belgium
Tel.: +32 10 47 80 72
Fax: +32 10 47 20 89
E-mail: christophe.devleeschouwer@uclouvain.be

1 Introduction

Context The growing number of stored video data, along with the improvement of computer vision techniques, increases the interest of the image processing community in automated video analysis. One of the most thriving application is automated visual surveillance system[7,12,10], which has two main purposes. The first one is to help the control room operators in the tedious work of watching the monitored scenes in real-time. For instance, it might give an alert when suspicious events appear or select the views having the greatest interest for the operator to watch. The second goal is to store, organize, and retrieve video sequences, in an intelligent and efficient way. In that case, detection and recognition of events permits to support automatic or manually assisted annotation tools.

Challenges What makes an event interesting is quite dependent on the context at hand. For instance, surveillance systems may monitor cars in parking lots, on a road intersection, or on highways. They may also observe people in public (airport, train station, mall, etc) or private place (banks, stores, entrance lobby of a company, etc), control checkpoints or railways. All these environments define a broad range of target events, which in turns induces lots of variety in the way visual features are selected and processed to characterize object of interest and to recognize the events associated to their behaviors [2,34,9,31,17,21,25,5].

State-of-the-art A detailed description of the previous works related to the automatic detection of video events is provided in Section 2. We observe that, beyond the way object are represented and described, most previous systems share a very similar framework. They always start with a segmentation algorithm, allowing subtracting the background from each frame. They rely on some a priori information to model the scene or the human body, analyze the objects motions based on tracking and occlusion reasoning, and finally conclude with an event recognition algorithm [7]. As a consequence, the efficiency of the event recognition system is strongly dependent on the video processing algorithms implemented up stream. Long-term tracking typically suffers from a lack of robustness in most realistic video surveillance scenarios, due to illumination changes, cluttered background, occlusions, appearance changes, etc.

Our proposal To circumvent the problem induced by tracking errors, we exploit the regular topology of video data, and decide to represent events based on the spatio-temporal organization of local patterns of motion features. Specifically, we consider a classifier-based approach to learn and recognize classes of possibly sophisticated spatio-temporal organizations of moving objects, detected along the time in the scene at hand. By considering both the absolute and the relative distribution of motion activities, our system becomes capable to understand both individual behaviors and interactions between people. As a specific property, our system does not rely on long-term tracking procedure or intermediate reasoning (e.g. about occlusions). This is interesting, since it does not make its robustness dependent on the tracking reliability. But this might also be constraining in the sense that omitting tracking restricts the class of events that our system can describe and recognize to events for which the matching of objects of interest along the time is not required. To circumvent this drawback, future research should consider detailed appearance-based descriptions of the local patterns detected at each time instant, so as to become able to discriminate events based on the spatio-temporal organization of patterns that are likely describing a unique (set of) object(s) along the time. This would be a direct extension of our framework that could take long term matching of objects into account, without relying on accurate tracking solutions.

In this paper however, we do not consider such extension, and focus on demonstrating the efficiency of our video analysis framework in cases for which events can be recognized without the need for long term object matching. At this point, it is worth mentioning that defining visual events based on supervised learning of a classifier is not a trivial task since:

- Events are semantically complex, and their interpretation is often subjective. Therefore, designing a system that support interaction between the expert and the learning system (e.g. such as expert systems) is recommended.
- Events are often ambiguous, with a large amount of overlap between classes, especially in increasingly complex environments.
- Dynamic events are typically variable in length, which means that they are non stationary over the time dimension. Therefore, the model of each event should be scale invariant.
- One same event can often exhibit a wide range of motion characteristics. One requirement for the event detection algorithm is thus to be robust to the possible variations in an event from the structure of motion in the scene.
- Some events of interest occurs rarely. As a result, there is a broad variation in the occurrence of the events, and using a supervised learning technique can be knotty because of the few available training samples for one or more classes.

To cope with those issues, we follow a divide-and-conquer strategy, and propose to pipeline the recognition process in three main stages that are depicted in Figure 1. First, the foreground objects are extracted from the video sequences, and described by space-time volumes (*STV*) of local features. Formally, each *STV* roughly describes the motion and the shape of the objects in the scene, at a particular instant and location. Second, those *STV* features are aggregated into a limited set of clusters, defining a set of coarse to fine local feature patterns. Finally, a decision tree classifier is used to model the (potentially sophisticated) events of interest in terms of causal and temporal arrangements of feature patterns along the time. In order to reduce the risk of overfitting inherent to the limited size of the training data, while maintaining good accuracy, the classification is based on an ensemble of randomized trees.

Outline Prior works related to visual event recognition are first discussed and compared to our proposed approach in Section 2. Section 3 surveys the general architecture underlying our recognition framework. Then, the preprocessing steps, used to extract the foreground objects features, are detailed in Section 4. Our learning strategy is then presented. It is composed of a clustering and a tree-growing stage. Those two stages are respectively described in Section 5 and 6. Experimental validation is provided in Sections 7 and 8. We first analyze our machine learning strategy in a virtual environment, and then we test the whole framework in a real environment with the CAVIAR database. Finally we conclude and pave the way for future research.

2 Related work

The number of works related to human event recognition from video sequences has recently grown at a tremendous rate.

Sequential process definition and recognition Lots of works related to activity recognition in public space try to model people behavior based on generative models, like Hidden Markov Models (HMM) [17,21] or Bayesian Network (BN)[25], to capture

the discriminatory nature of object trajectories. [5] reviews these methods. An HMM is essentially a quantification of a system’s configuration space into a small number of states, together with probabilities of transitions between the states. Nonetheless, Standard HMMs are often inappropriate for detecting visual events where a large variation of the data as well as discontinuities are present. Hence, more complex formulation of HMMs, such as layered HMMs, mixture HMMs etc., is often used for dynamic activities understanding. In [21], the authors compare two state-based statistical learning architectures (HMMs and Coupled-HMMs) for modeling behaviors and interactions. Given the limited amount of training data available, they propose to train the learner with a synthetic agent training system that allows creating flexible and interpretable prior behavior models. Next to HMM, stochastic context free grammar (SCFG) approaches, first experimented in language modeling and bioinformatics, have also been used for detecting the visual events [30].

More recently, methods dealing with dynamic bayesian networks have attempted to identify human interactions [33] without using any tracking process. Instead of trying to recognize short actions, the framework presented by Xiang proposes to model more elaborated events from the pixel changes distribution in the video sequences. He first detects and classifies object-independent actions at each frame. Then he uses the Bayesian Information Criterion in order to find a specific set of consistent actions. Finally, Dynamic Probabilistic Networks are formulated to model an activity within a video sequence, based on the temporal and causal correlations among discrete action. Our framework shares some characteristics with Xiang’s work, in the sense that we use short spatio-temporal template instead of trajectory features to model possibly complex activities in the scene. However, the two approaches differ significantly in the way they describe the temporal relationships between clusters of activities, respectively through dynamic inference and pure classification mechanisms.

As a conclusion, though generative sequential models are reliable to describe accurately the events for which a common structure can be easily learned from training data, we expect that classifier-based inference approaches are more appropriate when ill-posed learning problems dealing with sophisticated events that are characterized by a large amount of variability [7]. Also, classifier-based solutions are better in handling multi-dimensional and multi-class dataset. For instance, in [23], Perez compared a set of classifiers for human activity recognition. It showed that HMM performs quite well when the number of features is limited, but that, compared to most of the classifier-based approaches, its performance degrades when the number of features increases. Also, some of the interesting activities to detect were often hardly recognizable due to the ill definition of the labels in the ground truth.

Space-time volumes of features Another category of methods defends the direct recognition of the motion itself [2, 34, 31]. These works share the idea that actions can be analyzed by looking at a moving foreground object as a space-time volume. Bobick and Davis [2] were one of the pioneer to experiment this new approach. Using information derived from binary cumulative motion images, they form a *motion history image* (MHI) where each pixel intensity is a function of the temporal history of motion at that point. The resulting MHI are then compared with a matching algorithm. MHI are also used in Orrite paper [22] and projected into a new subspace by means of the Kohonen Self Organizing feature Map (SOM). A new SOM is trained for each action by combining different views and movement belonging to the same action, which allows the action recognition to be independent from the viewpoint. Yilmaz and Shah propose in [34] to match neighboring masks into spatio-temporal volumes in order to represent

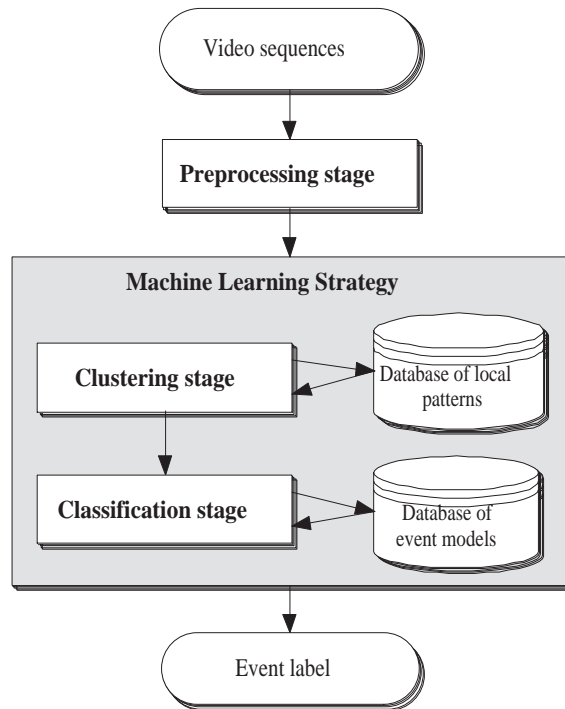


Fig. 1 Event recognition framework. The pre-processing stage considers groups of 6 to 12 consecutive frames, and extracts a set of features that characterize the spatio-temporal volumes (STVs) of activity detected in those frames. The STVs features are clustered, mainly to reduce dimensionality. Events are then represented and defined based on the spatio-temporal organization of the SVTs that have been detected along the video sequence.

the actions. Then they compute the trajectories of specific points on the silhouette boundary, use the Weingarten mapping to determine the relevant action patterns, and finally use those patterns to infer the events. Similar to us, Wang and Suter exploit in [31] a dimensionality reduction approach in order to obtain low-dimensional embedding representations of dynamic silhouettes. They further compute similarity measures between the representations with the median Hausdorff distance and finally classify the actions using the nearest neighbor method.

Most of these works aims at extracting meaningful features from space-time template of the moving objects in the video. Yet, they consider that each template contains the whole information about the human action and is enough by itself to infer the events of interest. Therefore, they restrict themselves to applications where short and individual close up actions are targeted, like human gesture recognition. Instead, we consider the topological organization of a set of local feature patterns, potentially associated to distinct objects, so as to describe activities without any constraint on the temporal and spatial extent of the event to recognize.

3 Overall architecture

Terminology Following Xiang[33] terminology, we adopt the following definitions in the rest of the paper. At the most primitive level, a *primitive event* is a particular motion or combination of motion and shape whose execution is consistent and localized temporally in the video sequence, thus requiring no contextual or sequence knowledge to be recognized. It can represent a simple movement, as well as what’s happening in a whole video frame (e.g. a moving crowd). At the intermediate level, an *action* can be referred as a sequence of movements, where the main required knowledge is the statistic of the sequence. Much of the gesture understanding systems fall within this category of motion perception. Finally, at the top level are the *activities*, which are larger scale events, typically including causal relationship between some actions or primitive events and the interaction with the environment. It is thus used for larger scale scene, possibly involving numerous objects co-existing or interacting in a shared common space.

Scope and approach In this work, a visual surveillance event is seen as a combination of primitive human events occurring at specific times and places in a monitored scene. In order to recognize those events, the proposed framework relies on three main stages (Figure 1).

In a first stage, a set of preprocessing methods is exploited to detect the foreground objects and extract useful information from them. This step is essential in automated visual surveillance system, for which the valuable information is generally composed of the moving objects (people, vehicles...). Once those objects are extracted, the silhouettes of each object are concatenated temporally on a few frames in order to obtain space-time volumes (*STV*). The *STV*, originally used for gesture recognition, are essential to obtain robust shape and motion features of the dynamic objects in the scene. The events of interest can then be recognized by reasoning about how the objects look like, where and how they move, and their possible interaction.

In a second stage, we describe the *STV* with a set of coarse to fine patterns, in order to reduce the dimensionality of the data and be able to infer the events. An unsupervised decision tree method is envisioned to cluster the *STV* according to their features. In this process, each node of the clustering trees identifies a specific local pattern characterizing a group of *STV*. Thus, each node is labeled, and those labels are associated to the appropriate time-steps in the video sequences, according to the type of *STV* existing in each time-step.

In the third stage, the events are classified, based on the temporal and causal relationships between the patterns in the sequences, for example by asking if there is a pattern X 'before' a pattern Y . Those relationships are specified by coarse constraints on the temporal arrangement of the patterns, and do not involve absolute temporal location or scale constraints. The ensemble of randomized trees methodology [8] is used in order to find the discriminatory patterns and their possible relationships.

At the opposite to Xiang work [33], our intermediate features (i.e. the local patterns) are numerous and not especially consistent. But then only a few discriminatory patterns are used during the classification stage to model the events. Instead, they learn statistical models of the temporal sequencing of the primitive events. Therefore they rely completely on the relevance of their primitive events and on the robustness of their primitive event recognition method.

This machine learning framework is inspired on Geman works in [1]. He suggests recognizing specific patterns in images also by using a set of decision trees. In his work,

each node of the classification trees chooses a test that describes a spatial arrangement between local features. Like us, those local features are labels for clusters made at a previous step. Those clusters are also clustering tree nodes, where each node is a successive partition of small sub-images. We propose in this paper to extend this concept for events recognition in video surveillance sequences. In our case, the challenge lies in identifying key objects and actions that are indicative of events of interest. If we use each objects feature independently in a decision tree, the risk is to use features that belong to uninteresting objects or actions and to over-fit the training data. By using higher-level descriptors possibly representing a primitive event of interest in the scene, this risk becomes greatly reduced.

4 Events of interest and features extraction

In this section, we explain how features are extracted from the video sequences to feed the subsequent classification engines. Since we are interested in discriminating the video sequences based on their moving objects, we first consider the extraction of foreground object silhouettes, and then characterize (the movement of) those silhouettes based on a limited set of features.

Foreground object segmentation Prior to features extraction, the silhouette of the objects of interest are first computed based on any suitable background subtraction algorithm [20,16,11,6,19]. The approach we have adopted relies on a mixture of Gaussian to model the luminance of each pixel. It follows the formulation proposed by Stauffer and Grimson [29], augmented based on the improvements proposed by Dar-Shyang Lee in [14]. The main advantage is that it automatically supports backgrounds having multiple states like blinking lights, grass and trees moving in the wind, acquisition noise etc. Furthermore, the background model update is done in an unsupervised way when the scene conditions are changing.

From the images derived from background subtraction, each frame is binarized and the morphological operations are applied to filter the noise. Then, a fast connected-component operator is used to differentiate each foreground object, spatially and temporally. Small regions are then eliminated, and big regions are considered as the relevant objects. In this work, a *spatio-temporal volume* (STV) is thus defined as a group of local pixels belonging to the foreground, and linked spatially and temporally, where the temporal window size is set empirically. In section 8, we build the *STVs* based on 9 frames, with a 4 frames overlap for video sequences recorded at 25 frames per second.

Features The choice of a set of discriminatory features is crucial for any classification algorithm. This is even worse in our case, since the features are expected to capture the essence of the foreground mask appearance changes resulting from distinct behaviors. Typically, different kinds of features are needed to characterize the independent activities of individuals, or the interaction of a group of people. The context might also play an important role. For instance, the absolute position of the moving objects is critical to detect an intrusion; whilst in other cases the disorganized displacement in some open public area might be more representative of a particular stress or anxiety.

In our experiments, we have considered three groups of features to describe each STV. Those features are expected to be sufficient to discriminate the events at hand.

The first group of features is defined directly based on the outputs of the STVs segmentation process. Let us consider one particular STV extracted at time t , within a window of size $(2k + 1)$, ranging from $(t - k)$ to $(t + k)$. Let (x_i, y_i) and (w_i, h_i)

respectively denote the position of the centroid and the bounding box dimensions of the STV slice extracted at time i , with $t - k \leq i \leq t + k$. We then define a set of features that characterize the average size and position of the spatio-temporal volume, i.e.

$$\begin{cases} f_{pos-x} = \sum_{i=t-k}^{t+k} \frac{x_i}{2k+1} \\ f_{pos-y} = \sum_{i=t-k}^{t+k} \frac{y_i}{2k+1} \\ f_{sz-h} = \sum_{i=t-k}^{t+k} \frac{h_i}{2k+1} \\ f_{sz-w} = \sum_{i=t-k}^{t+k} \frac{w_i}{2k+1} \\ f_{sz-S} = f_{sz-h} \cdot f_{sz-w} \end{cases} \quad (1)$$

The second group of features reflects the intrinsic activity of each STVs. Those features are discriminatory for most of the events considered in our experiments. They correspond to the average instantaneous speed and to the average velocity of the object supporting the SVT. Formally, similar to [15], we have

$$\begin{cases} f_{avg-spd} = \sum_{i=t-k}^{t+k} \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2} / (2k+1) \\ f_{avg-vct} = \sqrt{\left(\frac{x_{t+k} - x_{t-k}}{2k+1}\right)^2 + \left(\frac{y_{t+k} - y_{t-k}}{2k+1}\right)^2} \end{cases} \quad (2)$$

From the height h_i and width w_i of each SVT slice, we compute the following metrics, reflecting the silhouette appearance change on the temporal interval,

$$\begin{cases} f_{ch-sz} = \frac{\sum_{i=t-k}^{t+k} \left| \frac{(w_i \cdot h_i - w_{i-1} \cdot h_{i-1})}{w_{i-1} \cdot h_{i-1}} \right|}{2k+1} \\ f_{ch-wd} = \frac{\sum_{i=t-k}^{t+k} \left| \frac{(w_i - w_{i-1})}{w_{i-1}} \right|}{2k+1} \\ f_{ch-ht} = \frac{\sum_{i=t-k}^{t+k} \left| \frac{(h_i - h_{i-1})}{h_{i-1}} \right|}{2k+1} \end{cases} \quad (3)$$

The third and last group of features has been introduced to reflect and quantify the interaction occurring between the STVs. They envision each pair (n, m) of STVs extracted from the $(2k+1)$ frames of the window of interest. The features measure the distance and the difference of speed and velocities between the m^{th} and the n^{th} STVs. They also compute the ratio $f_R^{n,m}$, between the sum of velocities and the sum of the norms of velocities of the two STVs. This ratio gives a good description of the relative direction between the two STVs, since it tends to 1 when they are moving in the same direction, and is close to 0 when they are moving in opposite directions.

$$\begin{cases} f_D^{n,m} = \frac{\sum_{i=t-k}^{t+k} \sqrt{(x_i^m - x_i^n)^2 + (y_i^m - y_i^n)^2}}{2k+1}, \forall m \neq n \\ f_{\Delta avg-spd}^{n,m} = |f_{avg-spd}^m - f_{avg-spd}^n|, \forall m \neq n \\ f_{\Delta avg-vct}^{n,m} = |f_{avg-vct}^m - f_{avg-vct}^n|, \forall m \neq n \\ f_R^{n,m} = \frac{|| (x_{t+k}^n - x_{t-k}^n, y_{t+k}^n - y_{t-k}^n) + (x_{t+k}^m - x_{t-k}^m, y_{t+k}^m - y_{t-k}^m) ||}{|| (x_{t+k}^n - x_{t-k}^n, y_{t+k}^n - y_{t-k}^n) || + || (x_{t+k}^m - x_{t-k}^m, y_{t+k}^m - y_{t-k}^m) ||} \end{cases} \quad (4)$$

In final, the computed features are stored in a table that is used as input by our tree-based clustering and classification algorithms.

5 Clustering the features into local patterns

The previous section computes an ensemble of features for all foreground objects in the sequences. Individually, each single feature carries little information about any elaborated event. Discriminative power is expected to arise from the combination among the features, both spatially and temporally. The idea of the following section is to cluster the STVs according to their features. Those clusters are local patterns characterizing primitive events at specific instants in the video sequences, and are represented by a specific location, shape and motion of the STVs. The main purpose of the clustering stage is to reduce the dimensionality of the subsequent classification problem.

Since the labels associated to the training samples refer to the global event occurring in the video rather than to the occurrence of a specific local observation, we do not have explicit information about whether a particular STVs is discriminatory or not with respect to some particular event of interest. In particular, it is likely that identical local patterns do appear in video samples corresponding to distinct events. This is the case when the events differ in the temporal organization of local patterns rather than in the observation of a specific pattern at some particular instant. For this reason, our clustering procedure will be unsupervised. It will thus omit the labels assigned to the video samples from which STVs have been extracted. Moreover, since we do not have explicit information about the discriminative power of STVs, we can reasonably assume that any suitable clustering method will perform equally well regarding the final recognition rate of the system. For this reason, we have chosen to build our clusters based on decision trees. They offer the advantage (1) to define embedded, coarse to fine, and redundant STVs patterns, (2) to be flexible and intuitive in the way they define the clusters, and (3) to rely on similar and efficient software tools than the subsequent classification stage.

It is worth noting here that the diversity offered by redundant clusters is compatible with the requirements of the subsequent event classifier. The effectiveness of an ensemble of randomized trees indeed mainly depends on the variety and weak correlation of the trees, which in turns depends on the diversity of the features considered to define the attributes in the nodes of the trees. Hence, building diverse (and partly redundant) clusters is well suited to the subsequent classification task envisioned in our framework.

Formally, we cluster similar STVs by using the classical unpruned decision tree methodology. Each node of the tree can be seen as a weak classifier that organizes the STVs of the input sequences into two branches, based on the binary answer of a question. Given a set of STVs in a tree node, the growing process selects the question in a vast database so as to maximize the intra-node distance (i.e. the Euclidean distance between the attribute values of the examples belonging to the same node) compared to the inter-node distance. Hence, as explained above, we do not use the events classes, making this stage completely unsupervised. The family of STVs in entire set of the video sequences is thus recursively partitioned with binary splits. A stop splitting criteria can be applied in order to prevent small and non-representative clusters. In our experiment section, we stopped splitting when the number of STVs in a node reached less than one percent of the total number of STVs in the training video sequences.

The three groups of features are considered separately to cluster the *STVs*. We thus build three pools of clustering trees. An example of clustering tree for the first pool is proposed in Figure 2. It shows the first three levels along with some instances from the CAVIAR video sequences. Once the trees are built, each node represents a

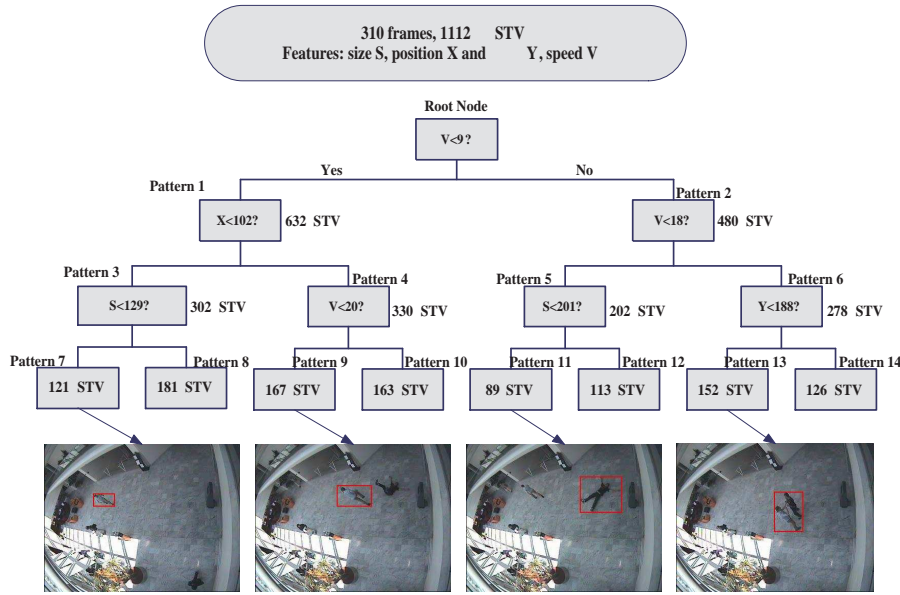


Fig. 2 Example of clustering tree built from the CAVIAR dataset. An attribute is assigned to each node, characterizing the local STVs in the two branches arising from the node. The classifier presented in the next section further analyzes the temporal and causal relationships among the STV patterns.

pattern and is tagged with a number (except for the root node), so that each time-step of the sequences can be labeled with the appropriate patterns. Therefore, those patterns are a data-driven combination of the *STV* features which allows us to reduce the dimensionality of the data while keeping most of the information. The number of clustering trees is set empirically, knowing that more trees means more diversity for the local patterns but also more redundancy and confusion for the subsequent event classifier, as described in the next section.

6 Tree-based events classification

In the previous section, we have explained how to define a limited number of representative local STV patterns to represent each video by a sequence of aggregated features. We now rely on those local patterns to describe the events to recognize in the video sequence. The purpose of the following section is to design a classification system that is able to infer knowledge about the events of interest, based on the observed sequence of STVs. We propose to use an ensemble of randomized trees (*ERT*) to define the events through temporal and causal relationships observed between the local patterns in the training samples.

6.1 Modeling the events based on spatio-temporal organization of local patterns

Decision trees are here considered to classify the video sequences according to the spatio-temporal relations observed among their STVs. In each node of the tree, a binary attribute is selected to partition the input training sequences in two branches, depending on the presence/absence of a specific arrangement of STVs in the video sequences. To make those arrangements time-scale invariant¹, coarse binary relations like "before", "after" and "at the same time" are used. An example of arrangement would then be: pattern X exists "before" pattern Y which exists "simultaneously" to pattern Z .

The example of a tree that has been built to recognize the events in the CAVIAR sequence is considered in Figure 3. The left part of the figure presents the list of local STVs that have been defined by the initial clustering stage, based on the three categories of features presented in Section 5. The right part of the figure presents an example of tree path that is followed by the sequences corresponding to the event 'fight'. We observe that each node encountered along the tree path raises one question related either to the presence of a specific local pattern within the sequence, or to the existence of a temporal relationship between local patterns.

Formally, we adopt the following methodology to build the decision trees in a supervised and recursive way. At each node, one question -or attribute- is chosen within a vast database of candidate questions. As explained below, the question is selected to provide a significant information gain, i.e. so as to reduce the impurity of the output variable within the local learning subset. Two formulations are considered to define the set of candidate questions in node N_j . The first formulation selects one STV pattern P_i in the set of patterns \mathcal{P}_{N_j} extracted from the training sequences in node N_j , and not yet considered by one of the questions encountered along the path linking the root of the tree to node N_j . A number of binary questions are then envisioned with respect to this pattern. The simplest one just asks whether P_i has been extracted from the video sample at hand or not. In addition to the inclusion of P_i in the sample, more restrictive questions considers binary relationships between pattern P_i and one of the patterns used in the previous nodes of the tree path linking the root to node N_j . Those patterns belong to $\mathcal{P}_{N_j}^{used}$, and are common to all samples in node N_j by construction. The second kind of formulations directly envisions binary relationships between a pair of patterns P_i and P_j exploited in at least one previous node of the tree. P_i and P_j are thus selected within the set $\mathcal{P}_{N_j}^{used}$, and the binary relationship is selected within the [before, simultaneous, after] set of relations.

In order to evaluate the entropy gain resulting from the binary partition associated to a particular attribute, we compute a *Score* related to the normalized version of the Shannon information gain, as proposed by Wehenkel [32]

$$Score = \frac{2.I_C^A(S)}{H_A(S) + H_C(S)} \quad (5)$$

where S is the set of training video samples in the node, $H_C(S)$ is the entropy of each class, $H_A(S)$ is the entropy of the chosen attribute and $I_C^A(S)$ its mutual information. The *Score* metric varies between 0 and 1, 1 meaning that the classes are completely separated. In practice however, to mitigate potential overfitting problems, we build our classifier based on an ensemble of diversified trees, and do not systematically select the

¹ Scale invariance is required in numerous visual surveillance contexts.

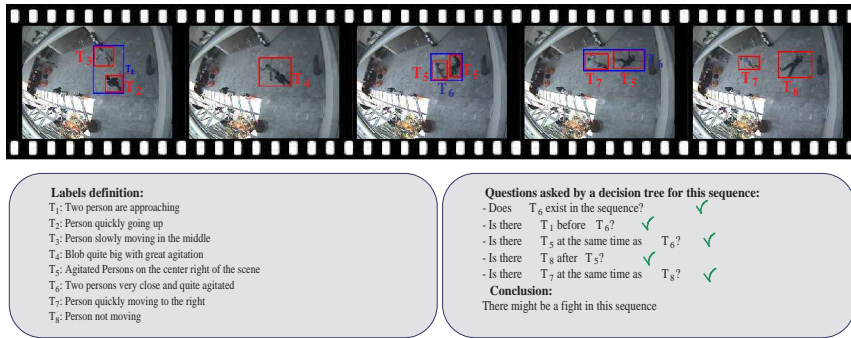


Fig. 3 Modeling an event through temporal relationships between STVs. On the left side of the figure, a set of STVs clusters ($T_1 \dots T_8$) is first created based on the *STV* features introduced in Section 5. Temporal relationships between STVs clusters are then considered to classify the events through a decision tree classifier. The right side of the figure presents an example of tree path associated to the event ‘fight’ in the CAVIAR dataset.

question that maximizes the *Score* value in a node. More details about the pseudo-random attribute selection are provided in Section 6.2 below.

Before digging into the mechanics of the classification engine, to explain how an ensemble of trees can effectively deal with the drawbacks of a conventional decision tree classifier, it is worth making some comments about the advantages and limitations of our STV-based model, when describing visual events. By considering both individual and relative STV features (see Section 5), our system becomes capable to understand both individual behaviors and interactions between people (see the ‘fight’ example in Figure 3). Besides, as a specific property, our system does not rely on long-term tracking procedure or intermediate reasoning (e.g. about occlusions). This is interesting, since it does not make its reliability directly dependent on the tracking effectiveness. But this might also be constraining in the sense that omitting tracking restricts the class of events that our system can handle. A priori, our system can handle all kind of events for which the long term matching of moving objects along the time is not required to discriminate the events of interest. In practice, this is not really true, since long term matching can be exploited by our system in cases for which few objects are present in the scene (in this case matching between objects observed at distinct instants is implicit), or when the size of objects is sufficient to differentiate them. However, our system cannot deal with cases for which explicit associations have to be defined among groups of silhouettes observed at distinct instants. To circumvent this drawback, future research should consider detailed appearance-based descriptions of the STVs, so as to become able to discriminate events based on the spatio-temporal organization of local patterns that are likely describing the same (set of) object(s) along the time. Formally, this could be done by forcing the nodes of the trees to limit the temporal relationships investigation to STVs with similar appearance. Hopefully, such a direct extension of our framework could take long term (probabilistic) matching of objects into account, and should thus allow covering a broad range of event classes without relying on accurate tracking solutions. A detailed analysis of such extension is however beyond the scope of this paper, whose primary objective is to demonstrate the relevance of the two stages video understanding framework depicted in Figure 1.

6.2 Ensemble of randomized trees

Similar to numerous previous works [1,3,8], randomization methods are considered to improve the tree decision accuracy. These methods explicitly randomize the tree-growing algorithm, and perform multiple runs of the growing process, so as to produce an ensemble of more or less diversified tree models, whose predictions are aggregated by a simple majority vote. The purpose of multiple trees is to solve the approximation error and the estimation error problem at the same time.

In our case, we follow the Extra-Tree methodology [8], meaning that we select the attributes of the questions at each node partially at random, i.e. we choose the patterns and the binary relation maximizing the information gain among X (set to 10 in our experiments) randomly chosen patterns/relations. This contrasts with bagging approaches, which introduce randomization based on sub-sampling of the input training data. In our case, we do not want to grow the tree models based on a subset of the training samples because we are often dealing with scenarios for which the training set is already quite small. Further sub-sampling would thus dramatically impact the model accuracy.

For completeness, it is worth telling that we stop splitting a node when all the training sequences in the node belong to the same classes. This criterion is the most appropriate for applications where few training data are available.

7 Experimental evaluation on simulated data

This section simulates a number of events related to the displacement and interaction of people in a closed environment. Simulating the events allows analyzing the proposed recognition framework without being restricted by the number of training samples. Working in a simulated environment also allows controlling the noise, making it possible to discuss the robustness of our system to typical noises occurring in a real environment.

7.1 Simulated data set generation

A Matlab code has been written to control the displacement of one or several blobs (representing objects or individuals) in a closed environment, so as to mimic some particular event of interest [27]. Formally, each blob is represented at each time instant by a vector whose components define (1) its position in a rectangular room, (2) its instantaneous velocity, and (3) its width and height (those two latter are equal for simplicity). The state vector of each blob is updated according to a random process whose distribution depends on the event scenario to emulate. Typically, the size of the blob is chosen according to a uniform random distribution, in an interval that depends on the intrinsic nature of the entity described by the blob (individual or object). Specific targets, e.g. a scene exit point or another individual, which directly depend on the scenario to emulate, generally guide the trajectories of the blobs. However, the instantaneous displacement of each blob is chosen randomly, within an interval of directions and magnitudes that are defined according to the target position and to the scenario to mimic. When two blobs get very close to each other, a larger blob replaces the two smaller blobs. The output of the code gives a table containing most of the blobs features proposed in section 4.

Five events of interest have been considered:

- *Meeting*. Two individuals slowly move closer, and stop or walk together once they get close to each other.
- *Pocket-picking*. An individual catches up another one, touches it, and runs away while the other one run after him or simply turn around.
- *Fighting*. One person catches up another one, touches it and makes random trajectories around it. The other individual then start to act similarly, changing speed and direction pseudo-randomly around the first one.
- *Leaving bag*. One person walks at a random place in the scene, stops a few seconds and leaves the scene in a random direction, abandoning a small motionless object in the scene.
- *Forbidden zone*. Someone enters a specific forbidden zone, and stays more than 2 second inside it. Both the arrival and departure direction and velocities are random.

In addition, one 'non event' class has been considered. The class *passers-by* is characterized by an individual going slowly and pseudo randomly through the scene from one random entry to one random exit.

7.2 Parameters settings, and system components assessment

In this section, we analyze how the performance of our system depends on its key parameters, and how they degrade when some of its components are not implemented. In final, it allows us to draw important conclusions regarding the importance of the clustering stage.

In this section, the performance of the system are measured in terms of accuracy, which is defined to be the quantity of correctly classified test samples over the total number of test samples. We use 100 randomly generated test samples of each class for each trial, while the number of training samples is set by default to 20 per class. Each trial is repeated five times and results are averaged.

Figure 4 analyzes how our system's performance depends on:

- The number of *clustering trees* ($N_{ClustTr}$) envisioned to define the local patterns.
- The number of *classification trees* ($N_{ClassTr}$) used in the learning process.
- The number of training sequences per class used ($N_{TrainSeq}$).

Without surprise, the system's performance monotonically increases with the number of trees and the number of training samples. Nevertheless, we observe a saturation effect, when $N_{ClassTr}$ and $N_{TrainSeq}$ reaches 50 in this particular case. More interestingly, we observe a local maximum on the curve depicting the accuracy as a function of the number of clustering trees. It reveals that the number of clustering trees should be chosen carefully between two extreme cases. On the one hand, too few clusters are unable to capture the main structures observed in the envisioned application. On the other hand, using too many clusters is equivalent to exploiting the entire unclustered feature space.

We now analyze the behavior of our system when some of its components are downgraded or simply by-passed. For this purpose, we use $N_{ClustTr} = 10$, $N_{ClassTr} = 30$ while $N_{TrainSeq}$ is set to 20 per class. Figure 5 presents the average performance obtained in six different cases of alteration :

- **Case 1** : This is the reference set up of the system. The accuracy reaches 92.7%.

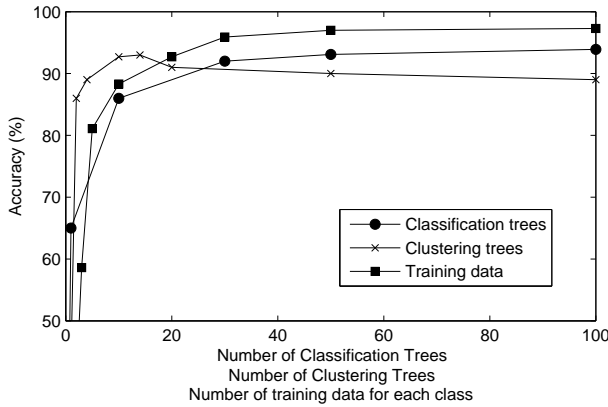


Fig. 4 Behavior of our system with respect to the three main parameters. By default, those parameters are set up at $N_{ClassTr} = 30$, $N_{ClustTr} = 10$ and $N_{TrainSeq} = 20$

- **Case 2** : The features related to the interaction between pairs of blobs (cf equations 4) are not used in the clustering process. The accuracy decreases by more than 7%, which shows that using appropriate features is essential.
- **Case 3** : Clusters are not any more defined based on a hierarchy of features, but rather based on a single independent feature. Thus, the patterns are not a combination of intervals coming from different features, but represent instead an interval of one specific feature. Figure 5 shows that accuracy reduces to (86.3%).
- **Case 4** : In this case, the nodes of the classification trees are not using the temporal relation between the tags. Each node simply chooses a pattern that is enough discriminatory by itself. Accuracy is reduced to 88.1%. Intuitively, the benefit of using the temporal arrangement between the local patterns should increase when the events can be described as an ordered sequence of short actions (or instantaneous observations).
- **Case 5** : Here, the clustering trees are not used. It means that the nodes of the randomized trees choose directly one feature and one threshold, instead of choosing (relationships among) local patterns. The system accuracy falls to 71%, which is the lowest result so far. Nevertheless, this case reaches an upper limit of 84% when adding 50 more training data for each classes, which reveals that clustering -by reducing the dimensionality of the subsequent classification problem- is especially relevant to deal with small training samples.
- **Case 6** : In this case, the ensemble of randomized trees are built without considering the event classes. Hence, the questions at each node are chosen completely at random (as long as it separates one sample from the training samples). The event classes of the training samples are only used to assign the appropriate classes to the leaves of the trees. We observe that the system remains relatively accurate with 88% of correct classification, but with a variance 2 times higher than previous case.

From the results presented in Figure 5, we conclude that **Case 3, 4 and 5** demonstrate the importance of running a clustering stage prior to the classification trees. First, **Case 3** and **Case 5** shows that using multi-dimensional representative patterns

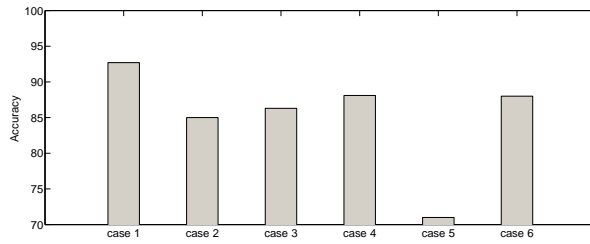


Fig. 5 Accuracy when the features construction or the classification process changes. Each case corresponds to one specific change in our framework.

(as defined by clustering stage) at each node of the classification trees gives better performance than directly using individual features.

Second, the clustering stage allows the classification stage to use temporal arrangement between the created patterns, and **Case 4** shows that these temporal relations are useful for modeling the events of interest.

7.3 Evaluating the impact of noise

In this section, we investigate how parasite activities or error-prone feature extractions may affect the recognition system. By parasite activities, we mean real activities occurring in the scene, but that are not related to the event of interest. This is for example the case of people crossing the scene at hand (i.e. the passers-by). By error-prone feature extraction, we refer to the artifacts affecting the video sequence analysis, typically the foreground silhouettes extraction. Table 7.3 analyzes the impact of those two phenomenons on the recognition accuracy. The first column gives the accuracy when both the learning samples (LS) and the test samples (TS) are disturbed. In the second column, only the testing sequences are disturbed. This second case is relevant in the sense that, in a practical deployment scenario, manual assistance might be considered to correct erroneous features and/or to erase a potential passer-by.

Table 1 Accuracy when the input data are subject to erroneous features or irrelevant passer-by's. Column *TS and LS* presents the accuracy when both the learning sample (LS) and the test samples (TS) are disturbed. Column *TS* corresponds to the accuracy when only the test samples are disturbed.

Input data	Overall Accuracy	
	TS and LS	TS
Only events	92.7 %	92.7 %
1 passer-by added	77.9 %	87.7 %
2 passers-by added	61.7 %	82.5 %
3 passers-by added	56.8 %	78.7 %
Random noise added	90.8 %	91.4 %
People split in multiple blobs	87 %	90.1 %

The first row of the table gives the accuracy of the system when using a clean dataset. The next 3 rows show how the accuracy decreases when up to three passers-by are crossing the scene. Performances are decreasing quickly, especially when the

algorithm is trained with the passers-by. Finally, the last two rows of Table 1 demonstrate the robustness of the system towards two conventional kinds of noise. Random noise is simulated by adding in each frame of the sequence up to 3 blobs having a random position, size and speed. It reflects false blob detection due for example to illumination changes or shadows in real environments. The last row of the table outlines a very common artifact in the blobs segmentation process. It replaces the blob characterizing one person by two or three blobs that are very close to each other.

One of the most important challenges our framework wanted to answer concerns its sensitivity to the input quality of the video. Four common noises generally affect the segmentation result, which tends to affect subsequent classification stages. The first one is randomly distributed, and typically due to the low quality of the camera, the global illumination changes or other contextual elements. The second noise corresponds to fragmentation of the foreground object. Table 1 shows that our architecture is quite robust toward these types of noise. A third common source of error appears when the foreground objects are hardly differentiable from the background, or when it is too far or hidden from the camera. This kind of noise leads to excessive missed detections errors. By being able to pick hierarchically the most discriminative patterns only, our learner is able to skip moment in the sequences where no valuable data are available, making our approach able to recognize events even in presence of partial observations. The fourth common error in the event analysis happens when two nearby people tend to get clustered as a single merged object (for instance the pattern *T4* in Figure 3). We get over this merged detection error thanks to appropriate *STV* features.

Nonetheless, one could question the choice of a tree structure for classification regarding robustness to noise. Although a single decision tree often tends to over-fit the data and thus lacks of robustness, ensemble methods applied to decision trees like *ERT* proved to be efficient and robust in image classification systems [8]. It was also applied with success on multivariate time series [28,13]. Our main motivation in experimenting *ERT* on video event recognition is its computational efficiency which makes the system usable in real-time, and also its flexibility, which allows us to tend towards a greater genericity.

8 Experimental evaluation with the CAVIAR Data Set

This section aims at evaluating the performance of our proposed system in real-life conditions, and to compare them to previous works.

The video clips considered in this section were captured by the CAVIAR project² with a wide angle camera lens in the entrance lobby of the INRIA Labs at Grenoble, France. The resolution is half-resolution PAL standard (384 x 288 pixels, 25 frames per second) and compressed using MPEG2. The CAVIAR team members acted out six basic scenarios. A ground truth XML version of the sequences is also available together with the videos, describing each blob and labeling them with specific events. There are 29 sequences for a total of about 40K frames, with between 300 to 2500 frames each, all hand-labeled. Like in most real visual surveillance contexts, only a few training samples of the activities are available, which makes supervised learning especially challenging.

We use our proposed framework to classify both the short-term actions and long-term activities happening in the video sequences. Since the ground truth data have

² <http://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1/>

been widely used by researchers to test specific stages of their long-term activity recognition framework [15, 26, 24], we first use the ground truth data to train and assess the clustering and classification stages of our framework (Section 8.1). Then, we consider a real-life scenario, and only use the videos (without the feature extraction ground truth) as inputs of the entire pipeline depicted in Figure 1, including the STVs feature extraction tools (Section 8.2).

8.1 Recognition based on the CAVIAR feature extraction ground truth

To compare our system to previous works, independently of the low-level feature extraction module, this section considers that the CAVIAR ground truth features are provided as inputs to our clustering and classification tools.

The ground truth data are XML sequences labeled by humans, frame-by-frame. Individuals and groups are tracked along the sequences, and have a unique identifier in each sequence. For each of them, a bounding box is given, with its position and orientation. They all have a context label, a situation label, a role label and a movement label. The context involves the people in a sequence of situations, where it plays a role. In the rest of the section, our proposed framework is named *Pattern Trees*, because the patterns that are created by the clustering process are used as input to the classification trees.

8.1.1 Recognizing short-term actions

The first experiment below tries to recognize the instantaneous movements of the people in the scene, with the categories *inactive*, *active*, *walking* and *running*. Our goal is to analyze our system performance on cases for which the events can be recognized on short time intervals. In particular, we want to check if the clustering stage does not act as a burden for the subsequent classifier, when events are simple and temporally localized.

STV features are based on 12 consecutive frames belonging to the same class. STV features are extracted every 6 frames, so that there is an overlap of six frames between consecutive STVs. At each time-step, the sample is classified in one of the four categories. It gives us a data set with very imbalanced classes (Table 8.2.1).

Table 2 Number of samples per classes

Inactive	Active	Walking	Running
893	1342	4438	134

From those samples, we run a 10-fold cross validation, where one tenth of each class is kept for the validation. Results are given on table 3, where IN , A , WK and R are the classes *inactive*, *active*, *walking*, *running*. We use the subscript A for the actual classes and the subscript P for the predicted classes. In case (a), the Pattern Trees method runs with all the features proposed in section 4, while in case (b) only the features related to the change of position and the change of size (i.e. equation 2 and 3) are used. In case (c) and (d), we use an ensemble of randomized trees (ERT), without any clustering stage.

Table 3 Confusion matrix for the 4 classes *inactive*, *active*, *walking*, *running*

%	IN_P	AP	WK_P	R_P
IN_A	84.2	6	9.8	0
A_A	4.3	82.1	12.5	1.1
WK_A	2.8	4.1	93.1	0
R_A	2.2	13.3	38	46.5

(a) Pattern Trees, all features

%	IN_P	AP	WK_P	R_P
IN_A	84.2	5.5	10.3	0
A_A	2.7	84.6	12.7	0
WK_A	3	3.2	93.3	0.5
R_A	6.4	9	53.3	32.3

(c) ERT, all features

%	IN_P	AP	WK_P	R_P
IN_A	85.7	9.2	9.1	0
A_A	4.3	82.4	13.3	0
WK_A	1.2	4.8	92.6	1.4
R_A	4.3	8.7	36.8	50.2

(b) Pattern Trees, selected features

%	IN_P	AP	WK_P	R_P
IN_A	86.9	4.8	8.3	0
A_A	2	86	12	0
WK_A	1.1	3.1	95.8	0
R_A	2.6	5.7	39.6	52.1

(d) ERT, selected features

Results from the ERT with selected features (case (d)) slightly outperform the result of the Pattern Trees with the same features. Indeed, the proposed classes can be recognized on one single frame, without taking into account any temporal information. Therefore, one of the advantages of using the pattern trees, which is the possibility to benefit from the temporal arrangement between the patterns, does not help. Nonetheless, when using all the features, the benefit of the ERT method falls down, showing that the Pattern Trees efficiently reduces the dimensionality of the classification problem.

Besides, we also observe that the results are quite poor on the classes having few training data (i.e. mainly the class *running*), which is normal as the unbalanced data issue is not taken into account during training. In Table 4, a balanced training set is used, with 100 training instances from each classes. In contrast to previous results, the class *running* gets the highest recognition rate.

Table 4 Confusion matrix, with a balanced training set and all features.

%	IN_P	A_P	WK_P	R_P
IN_A	77.4	8.1	14.5	0
A_A	8.5	85.9	4.1	1.5
WK_A	10.2	3.6	77	9.2
R_A	2.4	0	5.3	92.3

The results presented in Tables 3 and 4 can be compared with the ones proposed by Perez et al [24]. In their paper, they compare the recognition rate of 7 state of the arts classifiers for the four frame-based activities *inactive*, *active*, *walking*, *running*. Like us, they only use motion features based on the 2D centroid coordinates and the height and width of each person’s blob. Table 5 presents the results that Perez et al. obtain with 4 selected features, on a small subset of the CAVIAR dataset. They use half of their samples for training and half for validating, which leads to a very unbalanced training set, similar to our case in table 3(b). Yet, they are using 4 video clips, and compute the blobs features at each frame, while we are using 28 video clips and our STV features are computed every 6 frames (which leads to less redundancy between the STV). As the

preprocessing conditions are somewhat different, the comparison between the results is not totally fair. Nevertheless, they reach a very poor (null) accuracy for the classes with few training instances, which is not our case.

Table 5 Confusion matrices from the two best methods obtained with similar conditions in Perez et al.’s paper [24]. It is not a percentage anymore but it shows the number of samples in each classes. (a) They used here an HMM with a genetic algorithm to adjust the parameter of the mixture of gaussian that define each state of the HMM. Four selected features were used from 40 motion features.(b) Naives Bayes were used here, also with four selected features.

	IN_P	WK_P	A_P	R_P
IN_A	600	99	0	0
WK_A	190	1497	12	8
A_A	0	170	0	0
R_A	0	168	0	0

(a)

	IN_P	WK_P	AC_P	R_P
IN_A	576	57	0	0
WK_A	202	1377	0	128
AC_A	15	155	0	0
R_A	2	160	0	0

(b)

8.1.2 Recognizing long term behaviors

In this section, we complete the above experiments by considering long-term behavior recognition. The envisioned categories are *Fighter* (F), *Leaving Group* (LG), *Browser* (BR) and *Left Object* (LO), along with the category *Walker* (WK). Those are the categories defined for each individual in the *Role* field of the CAVIAR ground truth data. Our goal is to check the behavior of our system when the events tend to be more complex and less localized temporally. The number of samples for those events are given in Table 8.1.2.

Table 6 Number of samples per classes

	WK	F	LG	BR	LO
Frame level	5723	77	28	607	227
Activity clip	58	10	10	23	11

In the first row, the numbers of STVs are given for each class. The unbalanced issue is even stronger than in the previous experiments, with a huge part of the data being in the class *Walker*. In the second row, the number of *Activity clips* belonging to one of the categories is given, knowing that an *Activity clip* of activity A_k is defined by a time interval $[t_1, t_2]$ with the following rules [15]:

$$\begin{cases} L_{grad}(t_1) \neq L_{grad}(t_1 - 1), \\ L_{grad}(t) = A_k, \forall t \in [t_1, t_2], \\ L_{grad}(t_2) \neq L_{grad}(t_2 + 1) \end{cases} \quad (6)$$

where $L_{grad}(t)$ is the ground-truth activity label of frame t .

The classification performances are given in Table 7, for each class. The figures provided in Table Table 7 correspond to the number of correctly classified samples from one class over the total number of samples from this same class. The four columns of the table correspond to four different experiments. In *frame level* I , each individual is

classified at each time-step. Like in Section 8.1.1, no temporal arrangement between the patterns are possible, since we are only considering one STV at a time. In *frame level II*, people are also classified at each time-step, but their STV features one second before and one second after are taken into account. In this case, a sequence of 9 time-steps corresponding to a 2 seconds interval is thus used for the training and the validation of the Pattern Trees approach.

Eventually, the Pattern Trees are tested at the activity level, meaning that we make a decision for a complete *Activity clip*. Specifically, in *Activity level I*, the decisions made at each time-step in one *Activity clip* are used to infer a decision over the whole clip. Because we are using an ensemble of trees, the classification at each time-step defines an histogram over the classes, which is simply obtained by aggregating the class decisions of the trees. Since one histogram is created at each time step, the method *Activity level I* further aggregates them within an *Activity clip* to define the majority class at the clip level. In *Activity level II*, finally, we don't take the labels of each frame into account anymore. Instead, each *Activity clip* constitutes one single sample that is used for training and validating the Pattern Trees. Therefore, the number of instances dramatically decreases (cfr. Table 8.1.2). Like in Table 4, we use a balanced training set, which gives a better idea about the performance of our system. For all those experiments, 10 independent trials are done with about 8 *Activity clips* chosen at random for training, while the remaining clips are used for validation.

Table 7 Accuracy related to the 5 activities *Walker*, *Fighter*, *Leaving Group*, *Browser* and *Left Object*, and four different procedures. See the text for detailed definitions of procedures.

%	Frame level I	Frame level II	Activity level I	Activity level II
WK_A	64.1	65.6	95	84.3
F_A	74.7	79.4	100	90.4
M_A	68.6	75.5	100	87
BR_A	53.6	61.7	90.1	76.2
LB_A	59.2	64.2	85.8	75.1

When analyzing the figures provided in Table 7, we first observe an increase of the performance when switching from *Frame level I* to *Frame level II*, which tends to confirm that exploiting temporal relationships with the Pattern Trees is beneficial. Then, we note that the accuracy significantly increases when switching to *Activity level I*. This is not surprising, but it provides a useful reference to compare our method with the one of Lin et al.'s in [15] (see the next paragraph). Finally, results on the *Activity level II* are especially promising. They show that very few training samples are needed for the Pattern Trees to be efficient. Moreover, the scenario considered in this last case is closer to a real life automatic visual surveillance system, where the expert might only approximately label the video sequences to be used for as training instances.

Our results can be compared to Lin et al.'s work in [15]. There, the authors propose to represent an activity by a combination of category components based on gaussian mixture model. In order to recognize the activities, an algorithm to find the "confident" frames in an activity clip is also developed. Like us, they test their framework on the CAVIAR activities, both at the frame level and at the activity level. If their recognition performances are especially good for simpler events like *Inactive*, *Active* and *Walking*, they have a harder time recognizing activities *Fighting* and *Leaving bag*.

The unbalanced data issue can explain part of these difficulties. However, the classes *Running* and *Fighting* have almost the same number of instances, while the *Fighting* class error rate is twice as much as the *Running* class error rate. This difference might be due to the complexity of the *Fighting* class that needs a longer period to be recognized. Hence, we conclude that using a framework that is able to take into account the temporal information between the *STV* seems to be valuable for more complex activities.

8.2 Recognition of events from the CAVIAR video sequences

8.2.1 Performance on real life video clips

This section processes the CAVIAR video sequences, and not the ground truth dataset. The objective is to assess the performance of our system in real-life conditions, for which *STV*s features have to be extracted automatically. The events to be recognized are listed in Table 8. They correspond to the 5 scenarios proposed by the CAVIAR project. We added the event *Sign* which corresponds to the person showing in body sign language the scene number in each original video clip. Those clips include several events. We removed some parts of the data where none of those events was happening, and we annotated the other parts with the appropriate classes. More detailed information about this experimental procedure is available in [27].

Table 8 Events information

Events	Number of clips	Length
<i>Walk</i>	10	5s
<i>Browse</i>	10	7s to 15s
<i>Left Bags</i>	6	5s to 15s
<i>Meet</i>	6	5s to 10s
<i>Fight</i>	4	5s to 8s
<i>Sign</i>	15	8s

Each video clip is primarily segmented and filtered according to the proposed methods in section 4. Like previously, *STV* features are computed over 12 frames with a 6 frames overlap, which gives us approximately 4 time-steps per second to represent each sequence. Only *STV* larger than 50 pixels are considered. The perspective given by the camera field of view influences size, appearance and speed of the people in the scene. However, we keep the initial image measurements, without trying to achieve viewpoint invariance.

We used a 10-fold cross validation to validate our results. We have 6 classes, and thus classifying them at random would give for each sequence 16.7 % probability to obtain the right class. The performance of the system are presented in Figure 6, when the number of training samples are increasing. It shows that our framework works reasonably well with a great variety of events, and the performance tends to increase rapidly with the number of training samples. Moreover, the proposed system appears particularly robust to the input quality of the video. Indeed, despite our great attention to reduce the noise, a lot of artifacts could not be taken out by an automatic process. Nonetheless, it doesn't seem to dramatically affect the recognition process.

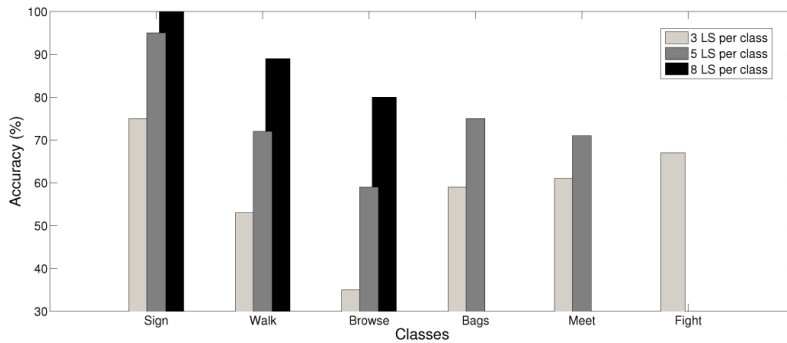


Fig. 6 Accuracy of the system for each class, when the number of learning samples increases.

The sequences we are using to train the system are often more than 10 seconds long, which lead to a great number of features, but still very few training samples. Reducing the dimensionality of the data is therefore a very challenging problem in our case, in order to avoid overfitting. From the huge set of available features in each data sample (i.e. a video sequence), it becomes barely impossible to find the appropriate combinations of specific features that can model each event. In order to reduce this input data, most of the recognition systems use a priori information that makes strong assumptions to support the detection of the foreground objects and their trajectories. When using a Markov process, they also make strong assumptions while choosing their states definition. At the opposite, our system makes no assumption prior to the classification stage.

8.2.2 Note on computational efficiency

The proposed framework is designed for an automated visual surveillance system, thus making it especially useful for real time application. This work has been implemented in Matlab, and thus obtaining a real time system was not realistic. Nevertheless, one could question about the computational efficiency of our framework.

Regarding the extraction of STVs features, we note that we are using a background subtraction algorithm that has been implemented in real-time in [14], whilst the rest of the feature extraction process simply computes the size and position of each blob. Therefore, our approach should compare favorably to methods relying on optical flows or on tracking algorithms.

Our machine learning algorithm is composed of two stages, during which ensembles of trees have to be trained.

In the first stage a small set of randomized clustering trees are built. According to Moosmann et al.'s work [18], the worst case complexity for building a tree is $O(T_{max}Nk)$, where T_{max} is the number of trials at each node, N is the size of the training set and k is the resulting number of clusters. However, in our framework we are forcing the clustering trees to be reasonably well-balanced, which gives a practical complexity of about $O(T_{max}N \log k)$. Moreover the value of $T_{max} \sim O(\sqrt{D})$, with D being the dimension of the data, as been suggested in [8], and is used in this work. It

leads to a total complexity of $O(\sqrt{DN} \log k)$ per trees, which can be compared with K-means complexity, equal to $O(DNk)$.

We obtain a similar complexity for the classification trees, meaning that the complexity of our machine learning framework is about $O(nb_{clust}\sqrt{DN} \log k + nb_{class}\sqrt{DN} \log k)$, where nb_{clust} and nb_{class} are the number of clustering trees and classification trees. The complexity of this process is therefore close to the one of the ERT algorithm [8], whose computing time is comparable to the Single CART Tree [4].

9 Conclusion and perspectives

This paper proposes a system that is able to recognize possibly sophisticated visual surveillance events. From the recorded video sequences, our system first extracts spatio-temporally local patterns of features, describing moving objects in the scene. It then maps those patterns to a set of clusters that have been learned during the training phase. Finally, an ensemble of randomized trees is used to learn and model the events as a discriminative hierarchical combination of those patterns. This architecture allows in a simple and generic way to deal with most of the challenges of visual event recognition.

Our experimental section 7 on simulated data proves the effectiveness of our framework. Nonetheless, one of the main drawbacks is given on table 1. It shows that the performance of the system decreases quite rapidly in crowded scenes. A solution was proposed by training the system with clean data, which means asking the user to indicate the objects actually taking part to the events. Further improvement could also analyze how to use histograms of similarity between *STV* to enrich our model, along with more appropriate features characterizing each *STV* or each frame more globally.

Finally, one important issue related to surveillance event recognition is the tedious and expensive work of labeling the stored video sequences. By using a limited training set, there is a danger of over-fitting the model, thus resulting in poor generalization to unlabeled data. Semi-supervised learning is an approach that has been used recently in event detection [30,35] for refining the models with both labeled and unlabeled data. In our case, our primary strategy is to use an unsupervised clustering method for dimensionality reduction purpose, prior to classify the data. We showed in section 8 that our system is especially well suited to deal with few training samples and embarrassing features. Future work will also look at using an active learning procedure in order to add new training data online.

References

1. Y. Amit and D. Geman. Shape quantization and recognition with randomized trees. *Neural Computation*, 9(7):1545–1588, 1997.
2. A. F. Bobick and J. W. Davis. The recognition of human movement using temporal templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23:257–267, 2001.
3. L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.
4. L. Breiman, J. Friedman, R. Olsen, and C. Stone. *Classification and Regression Trees*. CRC Press, 1984. ISBN 0412048418.
5. H. Buxton. Learning and understanding dynamic scene activity: a review. *Image and Vision Computing*, 21(1):125–136, 2003.

6. R. Cucchiara, C. Grana, M. Piccardi, and A. Prati. Detecting moving objects, ghosts and shadows in video stream. *IEEE Trans. Pattern Anal. Machine Intell.*, 25:1337–1342, 2003.
7. H. Dee and S. Velastin. How close are we to solving the problem of automated visual surveillance? a review of real-world surveillance, scientific progress and evaluative mechanisms. In *Machine Vision and Applications, Special Issue on Video Surveillance Research in Industry and Academic Springer*, 2007.
8. P. Geurts, D. Ernst, and L. Wehenkel. Extremely randomized trees. *Machine Learning*, 63(1):3–42, April 2006.
9. L. Gorelick, M. Blank, E. Shechtman, I. M., and R. Basri. Actions as space-time shapes. In *In ICCV*, pages 1395–1402, 2005.
10. N. Haering, P. L. Venetianer, and A. Lipton. The evolution of video surveillance: an overview. *Mach. Vision Appl.*, 19(5-6):279–290, 2008.
11. I. Haritaoglu, D. Harwood, and L. Davis. W4:real-time surveillance of people and their activities. *IEEE Trans. Pattern Anal. Machine Intell.*, 22:809–830, 2000.
12. W. Hu, T. Tan, L. Wang, and S. Maybank. A survey on visual surveillance of object motion and behaviors. *IEEE Trans. Syst., Man, Cybern.*, pages 334–352, 2004.
13. M. Kadous and C. Sammut. Classification of multivariate time series and structured data using constructive induction. *Machine Learning Journal*, 58:179–216, 2005.
14. D. Lee. Effective gaussian mixture learning for video background subtraction. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 27(5):827–832, 2005.
15. W. Lin, M.-T. Sun, R. Poovandran, and Z. Zhang. Human activity recognition for video surveillance. In *Circuits and Systems, 2008. ISCAS 2008. IEEE International Symposium on*, pages 2737–2740, 2008.
16. S. McKenna, S. Jabri, Z. Duric, A. Rosenfeld, and H. Wechsler. Tracking groups of people. *Computer Vision and Image Understanding*, 80:42–56, 2000.
17. G. Medioni, I. Cohen, F. Bremond, S. Hongeng, and R. Nevatia. Event detection and analysis from video streams. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(8):873–889, 2001.
18. F. Moosmann, E. Nowak, and F. Jurie. Randomized clustering forests for image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(9):1632–1646, 2008.
19. J. Nascimento and J. Marques. Performance evaluation of object detection algorithms for video surveillance. *IEEE Trans. Multimedia*, 8:761–774, 2006.
20. N. Ohta. A statistical approach to background suppression for surveillance systems. *ICCV01*, 1:481–486, 2001.
21. N. Oliver, B. Rosario, and A. Pentland. A bayesian computer vision system for modeling human interactions. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):831–843, 2000.
22. C. Orrite, F. Martinez, E. Herrero, H. Ragheb, and S. Velastin. Independent viewpoint silhouette-based human action modelling and recognition. In *1st Int Workshop on Machine Learning for Vision-based Motion Analysis (MLVMA'08)*, in *ECCV'08*, Marseille, France, 2008.
23. Ó. Pérez, M. Piccardi, J. García, and J. M. Molina. Comparison of classifiers for human activity recognition. In *IWINAC (2)*, pages 192–201, 2007.
24. O. Pérez, M. Piccardi, J. García, and J. M. Molina. Comparison of classifiers for human activity recognition. In *IWINAC '07: Proceedings of the 2nd international work-conference on Nature Inspired Problem-Solving Methods in Knowledge Engineering*, pages 192–201, 2007.
25. P. Remagnino, T. Tan, and K. Baker. Multi-agent visual surveillance of dynamic scenes. *Image Vision Computing*, 16(8):529–532, 1998.
26. P. Ribeiro and J. Santos-Victor. Human activities recognition from video: modeling, feature selection and classification architecture. In *Proc. Workshop on Human Activity Recognition and Modelling (HAREM 2005)*, pages 61–70, 2005.
27. C. Simon. Visual event recognition: Application on the caviar video clips and simulated trajectories. Technical report, TELE Lab, UCL, Belgium, 2009.
28. C. Simon, J. Meessen, D. Tzovaras, and C. De Vleeschouwer. Using decision trees for knowledge-assisted topologically structured data analysis. In *International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS)*, Santorini, June 2007.
29. C. Stauffer and W. Grimson. Adaptive background mixture models for real-time tracking. *IEEE Computer Vision and Pattern Recognition*, 2:25–258, 1999.

30. H. Veeraraghavan, N. Papanikolopoulos, and P. R. Schrater. Learning dynamic event descriptions in image sequences. In *CVPR*, 2007.
31. L. Wang and D. Suter. Learning and matching of dynamic shape manifolds for human action recognition. *Image Processing, IEEE Transactions on*, 16(6):1646–1661, 2007.
32. L. Wehenkel. *Automatic learning techniques in power systems*. Kluwer Academic, Boston, 1998.
33. T. Xiang and G. Shaogang. Beyond tracking: Modelling activity and understanding behaviour. *International Journal of Computer Vision*, 67(1):21–51, 2006.
34. A. Yilmaz and M. Shah. Actions sketch: A novel action representation. *Computer Vision and Pattern Recognition (CVPR 05), IEEE Computer Society Conference on*, 1:984–989, 2005.
35. D. Zhang, D. Gatica-Perez, S. Bengio, and I. McCowan. Semi-supervised adapted hmms for unusual event detection. In *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1*, pages 611–618, 2005.