VIDEO EVENT CLASSIFICATION AND DETECTION USING 2D TRAJECTORIES

Alexandre Hervieu, Patrick Bouthemy
INRIA, Centre Rennes - Bretagne Atlantique, Campus Universitaire de Beaulieu, 35042 Rennes Cedex, France
ahervieu@irisa.fr, bouthemy@irisa.fr

Jean-Pierre Le Cadre
INRIA, Centre Rennes - Bretagne Atlantique / CNRS, Campus Universitaire de Beaulieu, 35042 Rennes Cedex, France
lecadre@irisa.fr

Keywords: Image sequence analysis, Image motion analysis, Hidden Markov models, Pattern recognition.

Abstract: This paper describes an original statistical trajectory-based approach which can address several issues related to dynamic video content understanding: unsupervised clustering of events, recognition of events corresponding to learnt classes of dynamic video contents, and detection of unexpected events. Appropriate local differential features combining curvature and motion magnitude are robustly computed on the trajectories. They are invariant to image translation, in-the-plane rotation and scale transformation. The temporal causality of these features is then captured by hidden Markov models whose states are properly quantized values, and similarity between trajectories is expressed by exploiting the HMM framework. We report experiments on two sets of data, a first one composed of typical classes of synthetic (noised) trajectories (such as parabola or clothoid), and a second one formed with trajectories computed in sports videos. We have also favorably compared our method to other ones, including feature histogram comparison, use of the longest common subsequence (LCSS) distance and SVM-based classification.

1 Introduction

Content-based exploitation of video footage is of continuously increasing interest in numerous applications, e.g., for retrieving video sequences in huge TV archives, creating automatic video summarization of sports TV programs (Kokaram et al., 2006), or detecting specific actions or activities in video-surveillance (Boiman and Irani, 2005; Hu et al., 2007). It implies to shorten the well-known semantic gap between computed low-level features and high-level concepts. Considering 2D trajectories is attractive since they form computable image features which capture elaborate spatio-temporal information on the viewed actions. Methods for tracking moving objects in an image sequence are now available to get reliable enough 2D trajectories in various situations. These trajectories are given as a set of consecutive positions in the image plane (x,y) over time. If they are embedded in an appropriate modeling framework, high-level information on the dynamic scene can then be reachable.

We aim at designing a general trajectory classification method. It should take into account both the trajectory shape (geometrical information related to the type of motion and to variations in the motion direction) and the speed change of the moving object on its trajectory (dynamics-related information). Unless required by a specific application, it should not be affected by the location of the trajectory in the image plane (invariance to translation), by its direction in the image plane (invariance to rotation), by the distance of the viewed action to the camera (invariance to scale). It should also be robust enough, since local differential features computed on the extracted trajectories are prone to be noise corrupted. It should not exploit strong *a priori* information on the scene structure, the camera set-up, the 3D object motions.

In this paper we tackle three important tasks related to dynamic video content understanding within the same trajectory-based framework. The first one is clustering trajectories extracted from videos. An unsupervised solution is developed. The second considered problem is recognizing (or retrieving) events in videos. Semantic classes of dynamic video contents are first learnt from a set of representative training trajectories. The third task is detecting unexpected events by comparing the test trajectory to representa-

tive trajectories of known classes of events.

The remainder of the paper is organized as follows. In Section 2, we outline related work on trajectory-based video content analysis. In Section 3, we introduce the local differential features considered to represent 2D trajectories. We show that they are invariant to 2D translation, 2D rotation and scale transformation, and we also describe their computation. Section 4 presents our HMM-based framework to model trajectories. It can be viewed as a (statistical) quantization of the local features while accounting for their temporal evolution. We also describe the HMM-based similarity measure used to compare or to classify trajectories. Section 5 deals with the detection of unexpected events. Section 6 introduces other classification methods which will intervene in the comparative experimental evaluation of the proposed method. In Section 7, we present the two data sets used to test and compare the methods. The first one is composed of typical classes of synthetic (noised) trajectories (such as parabola or clothoid), and the second one includes trajectories computed in sports videos. Results are then reported and discussed. Concluding remarks are given in Section 8.

2 Related work

Trajectory analysis can help recognizing events, actions, or interactions between people and objects. First methods considered point coordinates and local orientations on image trajectories as input features (Bashir et al., 2007; Buzan et al., 2004; Chan et al., 2004; Porikli, 2004). Using these features leads to express strict spatial similarity between trajectories. Other methods use velocities as features to compare 2D trajectories (Hongeng et al., 2003; Wang et al., 2006), but visual velocity still depends on the distance of the viewed action to the camera.

Different methods have been developed to compare and cluster trajectories in order to analyze the content of video sequences. In (Buzan et al., 2004), authors resorted to the Longest Common Subsequence (LCSS) distance (Vlachos et al., 2002), to classify trajectories computed in an image sequence acquired by a single stationary camera for video surveillance. It is based on a hierarchical unsupervised clustering of trajectories where trajectory features are vectors of 2D coordinates of the trajectory points. Wang et al. introduced a novel similarity measure based on a modified Hausdorff distance and a comparison confidence measure (Wang et al., 2006). They compare the distributions of the spatial coordinates of the trajectory points, and also use other attributes, such as velocity and object size. In (Bashir et al., 2007) was presented a trajectory-based real-time indexing method, using PCA and spectral clustering. A system that learns patterns of activity from trajectories, and hierarchically classifies sequences using a codebook was developed in (Stauffer and Grimson, 2000). Other works (Li et al., 2006) considered statistical distributions of trajectory orientations exploited in a clustering algorithm. Recent work has explored modeling frameworks such as DPN (Dynamic Probabilistic Network) and HMM (Hidden Markov Model) to express the temporal information (causality) embedded in video trajectories and the semantic meaning that they convey. In (Hongeng et al., 2003) was described a complex event recognition method based on the definition of scenarios and on the use of Semi-Markov Chain (SMC). Chan et al. proposed a method for detecting rare events by representing motions and space-time relations between objects using HMMs (Chan et al., 2004). A recognition method for group activities was defined in (Gong and Xiang, 2003) relying on DPN to model and detect actions involving multiple objects. DPN are specially used to model the temporal relationships among different temporal events in the scene. Porikli defined distances to handle trajectories, especially HMM-based distances (Porikli, 2004). The methods based on HMMs, SMCs or DPNs developed so far are unable to treat short trajectories (see subsection 4.1). Let us also stress that all the aforementioned methods exploit features invariant to translation or scale transformation only.

The approach we have designed is different from those proposed so far in several points. First, we introduce local differential trajectory features which are able to jointly capture information on the trajectory shape and on the object speed. Besides, they are invariant to translation, rotation and scale transformations. We have also developed a procedure to compute them which is efficient and robust to noise. Second, temporal evolution of these features over the trajectory curve is explicitly accounted for by considering an original and effective HMM scheme. Indeed, the HMMs states are given by properly quantizing the real feature values. Our HMM method is also able to process trajectories of any sizes (especially small trajectories). Moreover, we have adopted a HMM distance which can be exploited both for clustering and recognizing dynamic video contents and for detecting unexpected events. All these elements make the overall framework we have defined general and flexible.

3 Invariant local trajectory features

A feature that represents both trajectory shape and object acceleration (more specifically, we mean velocity magnitude change) is required to capture the full intrinsic properties of a video trajectory. As stressed in the introduction, it should also be invariant

to 2D translation, 2D rotation and scale transformation, which will be helpful in most video applications.

3.1 Trajectory kernel smoothing

A trajectory T_k is defined by a set of n_k points $\{(x_1,y_1),..,(x_{n_k},y_{n_k})\}$ corresponding to the successive image positions of the tracked object in the image sequence (video shot). The term "object" must be understood in a broad sense, i.e., interest point, gravity center of a segmented region, window center,... To reliably compute the local differential trajectory features, we need a continuous representation of the curve formed by the trajectory. To this end, we perform a kernel approximation of T_k defined by

$$u_t = \frac{\sum_{j=1}^{n_k} e^{-(\frac{t-j}{h})^2} x_j}{\sum_{j=1}^{n_k} e^{-(\frac{t-j}{h})^2}},\ v_t = \frac{\sum_{j=1}^{n_k} e^{-(\frac{t-j}{h})^2} y_j}{\sum_{j=1}^{n_k} e^{-(\frac{t-j}{h})^2}}, \quad (1)$$

where (x_t,y_t) designates the coordinates of the tracked object at t and (u_t,v_t) its smoothed representation. h is a smoothing parameter to be set according to the observed noise magnitude. Explicit expressions can then be derived for the first- and second-order temporal derivatives of the trajectory positions: respectively, $\dot{u}_t$, $\dot{v}_t$, $\ddot{u}_t$ and $\ddot{v}_t$.

3.2 Derivation of the trajectory features

Let us first consider the local orientation of the curve given by $\gamma_t = arctan(\frac{\dot{v}_t}{\dot{u}_t})$. By construction, it is invariant to 2D translation and scale transformation. To add invariance to 2D rotation, let us now take the temporal derivative of γ_t, $\dot{\gamma}_t$, and let us analyze this quantity. We have $\frac{d(\tan\gamma_t)}{dt} = \frac{1}{\cos^2\gamma_t}\dot{\gamma}_t$. On the other hand :

$$\frac{d(\tan\gamma_t)}{dt} = \frac{\ddot{v}_t\dot{u}_t - \ddot{u}_t\dot{v}_t}{\dot{u}_t^2}.$$

Then
$$\dot{\gamma}_t = \cos^2\gamma_t \left(\frac{\ddot{v}_t\dot{u}_t - \ddot{u}_t\dot{v}_t}{\dot{u}_t^2}\right).$$

Also
$$\cos^2\gamma_t = (1+\tan^2\gamma_t)^{-1} = \frac{\dot{u}_t^2}{\dot{u}_t^2+\dot{v}_t^2}.$$

Hence
$$\dot{\gamma}_t = \frac{\ddot{v}_t\dot{u}_t - \ddot{u}_t\dot{v}_t}{\dot{u}_t^2+\dot{v}_t^2} = \kappa_t.\|w_t\| \quad (2)$$

where $\kappa_t = \frac{\ddot{v}_t\dot{u}_t - \ddot{u}_t\dot{v}_t}{(\dot{u}_t^2+\dot{v}_t^2)^{\frac{3}{2}}}$ is the local curvature of the trajectory and $\|w_t\| = (\dot{u}_t^2+\dot{v}_t^2)^{\frac{1}{2}}$ the local velocity magnitude at point (u_t,v_t). The numerator of $\dot{\gamma}_t$ is the determinant of matrix $\begin{pmatrix} \dot{u}_t & \ddot{u}_t \\ \dot{v}_t & \ddot{v}_t \end{pmatrix}$ and the denominator $\dot{u}_t^2+\dot{v}_t^2 = \|w_t\|^2$ is the squared velocity magnitude. Then, $\dot{\gamma}_t$ is also rotation invariant. We have also demonstrated that this local feature well captures both the trajectory shape and the object speed since it is the product of the local curvature and the instantaneous velocity magnitude.

4 Trajectory modeling and similarity

4.1 Design of the hidden Markov model

We resort to a hidden Markov model (HMM) to build the statistical framework we need since HMM naturally expresses temporal causality. The feature vector representing a trajectory T_k extracted in a video shot is the vector containing the n_k successive values of $\dot{\gamma}(t)$: $V_k = (\dot{\gamma}_1, \dot{\gamma}_2, ..., \dot{\gamma}_{n_k-1}, \dot{\gamma}_{n_k})$.

We also exploit the HMM framework in a somewhat original way since the HMMs states are given by properly quantized values of $\dot{\gamma}(t)$. To determine the HMMs state values, we first study the distribution of $\dot{\gamma}(t)$ on representative trajectories. We define an interval $[-S,S]$ containing a given percentage P_v of computed $\dot{\gamma}$ values in order to discard "outliers" and to control the number N of state values. Hence, a quantization is performed on $[-S,S]$ into a fixed number N of bins (whatever the value of S, in order to be able to compare trajectories using the estimated HMMs). This is illustrated in Fig.1 where synthetic trajectories of six different classes are drawn and their corresponding histograms are plotted restricted to $[-S,S]$. In contrast, in the HMM framework introduced in (Porikli, 2004) to model trajectories and their temporal evolution, the number of states remain difficult to set (it relies on a validity score that requires a balancing factor to be fixed), and the trajectory size should be much larger than the number of Gaussian mixture components (used for the conditional observation distribution) times the number of states, whereas our method is developed to handle trajectories of any sizes.

The HMM which models the trajectory T_k is now characterized by:

- the state transition matrix $A = \{a_{ij}\}$ with

$$a_{ij} = P[\ q_{t+1} = S_j \mid q_t = S_i\ ],\ \ 1 \le i,j \le N,$$

where q_t is the state variable at instant t and S_i is its value (i.e., the ith bin of the quantized histogram);

- the initial state distribution $\pi = \{\pi_i\}$, with $\pi_i = P[\ q_1 = S_i\ ],\ \ 1 \le i \le N$;

- the conditional observation probabilities $B = \{b_i(\dot{\gamma}_t)\}$, where $b_i(\dot{\gamma}_t) = P[\dot{\gamma}_t \mid q_t = S_i]$, since the computed $\dot{\gamma}_t$ are the observed values.

The conditional observation probability is defined as a Gaussian distribution of mean μ_i (i.e., the median value of the histogram bin S_i). Its standard deviation σ does not depend on the state and is specified so that the interval $[\mu_i-\sigma,\mu_i+\sigma]$ corresponds to the bin width. This conditional observation model can reasonably account for measurement uncertainty. It also prevents from having zero values when estimating matrix A in the training stage by lack of measures (especially in case of short trajectories). Otherwise,

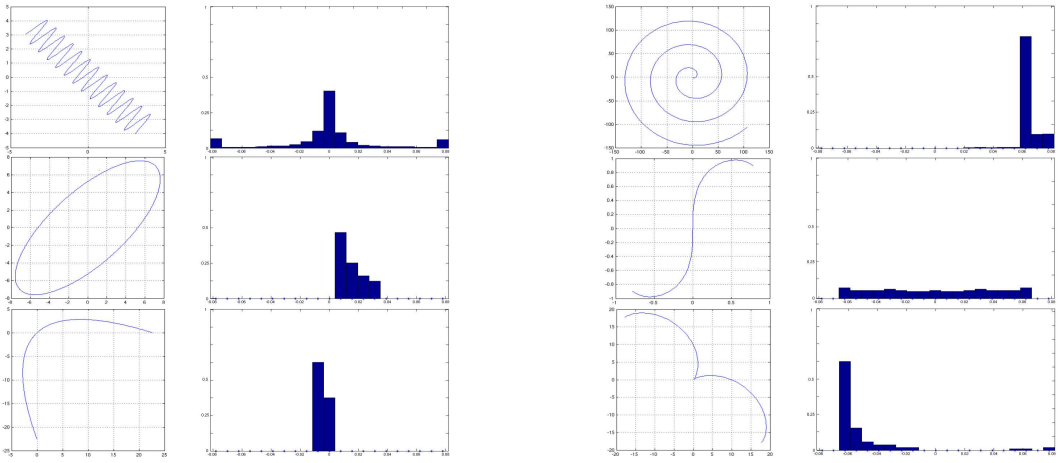


Figure 1: Six samples of synthetic trajectories (a sinusoid, an ellipse, a parabola, a spiral, a clothoid and a cycloid) and their associated normalized histograms plotted in $[-S,S]$, with $h = 3$, $P_v = 90\%$, and $N = 21$ (see text).

infinite distances would be found between trajectories (some coefficients of matrix A would be zero values).

To estimate A and π, we have adapted the least-squares technique proposed in (Ford and Moore, 1998) where the HMM is assimilated to a count process. If $H_t^{(i)} = P(\dot{\gamma}_t | q_t = i)$ (corresponding to a weight for the count process), empirical estimates of A and π, for a trajectory k of size n_k are given by

$$a_{ij} = \frac{\sum_{t=1}^{n_k-1} H_t^{(i)} H_{t+1}^{(j)}}{\sum_{t=1}^{n_k-1} H_t^{(i)}} \text{ and } \pi_i = \frac{\sum_{t=1}^{n_k} H_t^i}{n_k}. \quad (3)$$

As illustration, we show an example of a real trajectory in Fig. 2, its smoothed counterpart, the estimated values of A and π coefficients for its associated HMM.

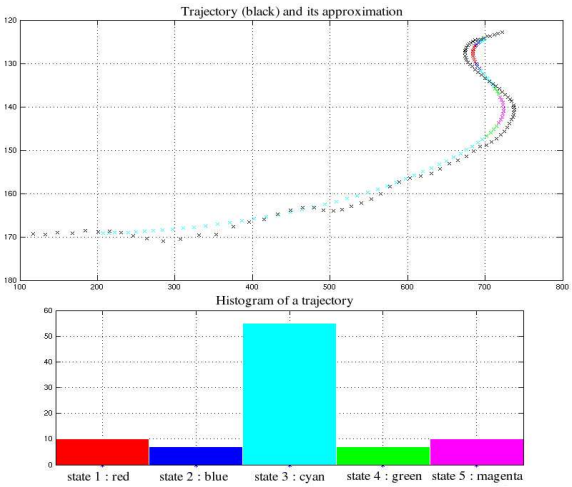
4.2 Similarity measure

To compare two trajectories, we have to define a similarity measure. To this end, we exploit the HMM framework we have built. We adopt the distance between HMMs proposed in (Rabiner, 1989). It can also be used to classify the trajectories since it is defined at the trajectory model level. Given two HMMs represented by their parameter sets λ_1 and λ_2 ($\lambda_i = (A_i, B_i, \pi_i), i = 1,2$), the distance D is defined by

$$D(\lambda_1, \lambda_2) = \frac{1}{T}[\log P(O^{(2)}|\lambda_2) - \log P(O^{(2)}|\lambda_1)] \quad (4)$$

where $O^{(j)} = \{\dot{\gamma}_1, \dot{\gamma}_2, ..., \dot{\gamma}_{n_j}\}$ is the sequence of measures used to train the model λ_j and $P(O^{(j)}|\lambda_i)$ expresses the probability of observing $O^{(j)}$ with model λ_i (computed with the Viterbi algorithm). To be used as a similarity measure, a symmetrized version is required:

$$D_s(\lambda_1, \lambda_2) = \frac{1}{2}[D(\lambda_1, \lambda_2) + D(\lambda_2, \lambda_1)]. \quad (5)$$



$$A = \begin{pmatrix} 0.80 & 0.12 & 0.05 & 0.02 & 0.00 \\ 0.14 & 0.61 & 0.16 & 0.08 & 0.01 \\ 0.02 & 0.11 & 0.71 & 0.13 & 0.03 \\ 0.01 & 0.09 & 0.17 & 0.56 & 0.17 \\ 0.01 & 0.02 & 0.07 & 0.13 & 0.77 \end{pmatrix}$$

$$\pi = [0.11\ 0.08\ 0.62\ 0.08\ 0.11]$$

Figure 2: Upper part: Plots of a real trajectory (extracted from a Formula One race video shot), its smoothed counterpart obtained with $h = 8$. Colors of the curve points stand for the different state values and correspond to the histogram bin colors. Middle part: histogram of the state values (with $N = 5$). Lower part: estimated transition matrix A and initial state distribution π.

5 Video understanding tasks

5.1 Unsupervised clustering of trajectories

We first describe how we address the unsupervised clustering task. Given a set of video shots (obtained by an automatic temporal video segmentation tech-

nique), we try to cluster the extracted trajectories in a sensible way to come out with relevant classes of dynamic video content. We first represent each trajectory by a HMM whose parameters are estimated as explained in Section 4. We then perform a classical binary ascendant hierarchical classification using the trajectory similarity measure introduced in the previous section. The distance between two groups of trajectories G_i and G_j is defined using an average link method, e.g, calculating the mean of the distance between all pairs of trajectories :

$$D_{average\ link}(G_i, G_j) = \frac{\sum_{T_k \in G_i, T_l \in G_j} D_s(T_k, T_l)}{\#G_i \#G_j}. \quad (6)$$

When achieving a binary ascendant hierarchical classification, the system needs to know when to stop the merging iterations. If the process is stopped too late, trajectories will be grouped in too few heterogeneous classes. Otherwise, if the process is terminated too early, classes will be too fragmented, and they would not correspond to relevant semantic classes. Two alternatives have been tested. We stop merging groups when a predefined number C of classes has been created, which means that the user has some knowledge on the diversity of the dynamic video contents. A second procedure is to fix a threshold τ, and merging is continued until the current minimum inter-classes distance passes the threshold τ.

5.2 Recognition of learnt classes of dynamic video content

Let us consider the problem of recognizing events, or equivalently, of retrieving instances of known classes of events in videos. It can be achieved in a supervised way (classes are learnt from training examples) or in an unsupervised way using the clustering stage described above. Each class is modeled by a set of HMMs corresponding to representative trajectories (those used in the training step, or those belonging to the corresponding cluster supplied by the initial clustering step applied on a subset of the video sequence base) which we will call the initial members in the sequel. Recognition is then performed by assigning the processed trajectory to the nearest class. As aforementioned, the distance to a class is defined using the average link method (see subsection 5.1).

5.3 Detection of unexpected events

Detecting unexpected (or equivalently, rare or abnormal) events is of interest in many applications. We tackle this issue with the same HMM-based framework. First, we consider a set of predefined (or learnt) classes represented again by the estimated HMMs of the initial class members. We compute for each class C_i its centroid G_i in the λ-parameter space from the estimated parameters λ_{l_i} of the HMMs of the initial members T_{l_i} of class C_i . Then we evaluate the distances of all the initial class members T_{l_i} to the centroid G_i using relation (5), and we denote R_i the maximum distance value. Let σ_i designate the standard deviation of these distance values. We decide that a test trajectory T_k corresponds to an unexpected event if, for every class C_i, $D_{average\ link}(T_k, C_i) > R_i + \sigma_i$, where $D_{average\ link}(T_k, C_i) = \frac{\sum_{T_l \in C_i} D_s(T_k, T_l)}{\#C_i}$.

6 Other methods for comparison purpose

6.1 Bhattacharyya distance between histograms

To assess the importance of introducing temporal causality, i.e., transitions between states, we have implemented a Bhattacharyya distance-based classification method. The Bhattacharyya distance D_b between two (normalized) histograms h_i and h_j of features $\dot{\gamma}_t$, respectively corresponding to two trajectories T_i and T_j, is defined by

$$D_b(h_i, h_j) = \sqrt{1 - \sqrt{\sum_{q=1}^{N} h_q^i h_q^j}} \quad (7)$$

where h_q^i is the histogram value of bin q for trajectory T_i. Similarly to the HMM-based method, we assign the test trajectory T_k to the nearest class and the distance to a class is defined using an average link method (see subsection 5.1).

6.2 SVM classification method

An efficient tool to perform supervised classification of patterns is the SVM method (Burges, 1998). As input for the SVM method, we take the HMM parameters corresponding to the trajectories. A SVM method needs patterns to be represented by vectors. Hence, for each trajectory T_k, a vector X_k containing the HMM parameters λ_k of the trajectory is created. Let us give an example with only $N = 3$ state values. We have

$$A_k = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}, \ \pi_k = [a_1\ a_2\ a_3],$$

and $X_k = [a_{11}\ a_{12}\ a_{13}\ a_{21}\ a_{22}\ a_{23}\ a_{31}\ a_{32}\ a_{33}\ a_1\ a_2\ a_3]$ is the vector representing the trajectory T_k. A SVM classification technique with a Gaussian RBF (radial basis function) kernel is used. The reported results are obtained using the "one against all" classification scheme.

7 Experiments

7.1 Synthetic trajectories

First, a set of typical trajectories has been generated to settle experiments with ground truth and easily specifiable data. More specifically, 8 classes (sinusoid,

parabola, hyperbola, ellipse, cycloid, spiral, straight line, and clothoid) have been considered and 8 different trajectories (of different sizes, including short trajectories) have been simulated for each class, with different parametrizations of the curves, and for several geometric transformations (rotation, scaling), as illustrated in Fig.1. Noised versions have been generated with different noise levels. Thus, we can evaluate the performance of the methods with respect to trajectory shape and length variations within a class, invariance to transformations and robustness to noise.

7.2 Video trajectories

Real trajectories have been extracted from a Formula One race TV program and from Alpine skiing TV programs, both filmed with several cameras. The trajectories are computed with the tracking method described in (Perez et al., 2002). The background motion due to camera panning, tilting and zooming is canceled. Trajectory shapes supplied by this method are thus nearly similar to the real 3D trajectories of Formula One cars (up to an homography, since the 3D motion is almost planar) and of skiers. Examples are plotted on Fig.3.

Figure 3: Images from video shots acquired by two different cameras at two different places on the circuit. The computed trajectories are overprinted on the images.

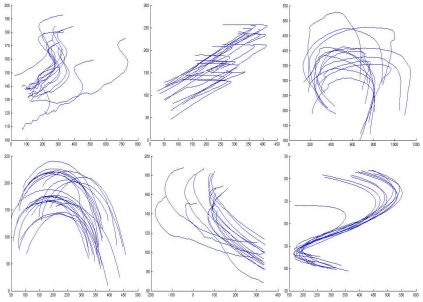


Figure 4: Plots of the 6 classes of dynamic content (trajectories) for a Formula 1 race video, each box contains a different class. A class of trajectories is composed of trajectories extracted from shots acquired by the same camera. The different classes correspond to different cameras placed throughout the circuit at different strategic turns.

7.3 Results on unsupervised clustering

We have applied the proposed unsupervised trajectory clustering scheme based on the distance between HMMs to trajectories extracted from a Formula One race TV program (Fig.3). In that real example, the different classes correspond to views supplied by six different cameras placed throughout the circuit at different strategic turns. Indeed, a given type of dynamic content is attached to a given view, and it can then be characterized by a specific trajectory shape and car speed which essentially depend on the turn configuration at the considered location of the circuit, whatever the passing car. Hence, ground truth is available while a real video (TV program) is processed. The clustering relying on the ascendant hierarchical binary classification (AHBC) technique supplies very good results with the two stopping criteria aforementioned. Due to page limitation, we will only report results obtained with the first stopping criteria. Table 1 contains results obtained with our HMM-based method and with the same AHBC technique but using the Bhattacharyya distance and the LCSS distance instead. Two cases were considered: 4 and 6 classes, the four first ones being nested subsets of the last six ones. Our method outperforms the Bhattacharyya distance based method (with approximately the same computation time), and the LCSS method while requiring a much lower computation time (at least five time faster). This experiment also demonstrates that our unsupervised classification method is able to form meaningful clusters since the later are very close to the ground truth groups presented in Fig.4.

	Percentage of correct clustering	
	4 classes	6 classes
HMM	100	92.2
Bhattacharyya	58.4	53.9
LCSS	84.4	72.3

Table 1: Results of unsupervised classification by an ascendant hierarchical binary classification (AHBC) technique, using the proposed HMM-based distance, the LCSS distance and the Bhattacharyya distance for Formula One cars trajectories. Two cases were considered: 4 and 6 classes, the four first ones being nested subsets of the last six ones (precisely the four classes on the left on Fig.4). Percentages correspond to rates of good classification for the extracted classes (the ground truth being known). As stopping criterion, the number of classes to create is provided to the AHBC technique.

7.4 Results on supervised recognition

We are now reporting results regarding the recognition task. We have compared our HMM-based method with the SVM classification method described in subsection 6.2 and the histogram comparison technique based on the Bhattacharyya distance outlined in subsection 6.1. To evaluate the performances, we have adopted the leave-one-out cross validation. Table 2 contains best classification results for the real sets of video trajectories (4 and 6 classes pre-

sented in Fig.4). Table 3 shows the performance of our HMM method for different levels of noise on the synthetic trajectories using different values of h.

Tests performed on the sets of synthetic trajectories gave very promising results, hence a perfect classification was performed for most parameter configurations (i.e., for N, h and P_v) with the SVM and HMM methods, where the technique based on the Bhattacharyya distance fail to efficiently classify synthetic trajectories (highlighting the importance of the temporal causalities modeled with HMM). The technique based on the Longest Common Subsequence distance (LCSS) (Buzan et al., 2004) gave good results but not perfect, and with a higher computation time (more than five times longer than with HMM based method). Recognition results with noised data (Table 3) shows that the parameter h helps handling efficiently noised data, by smoothing the trajectories.

For the evaluation on real videos (Fig.4), the same type of results have been obtained, very satisfying classification results for most parameter configurations with the SVM and HMM methods, and less accurate classification results with the techniques using the Bhattacharyya distance and the LCSS distance (Table 2). Besides, our HMM method is much more flexible than the SVM classification stage (e.g., adding a new class only requires to learn the parameters of that class).

	Percentage of correct classification	
	4 classes	6 classes
HMM	100	99.0
SVM	100	96.1
Bhattacharyya	100	93.1
LCSS	97.1	91.2

Table 2: Comparison of the best recognition percentages for the trajectories extracted from real video, using the leave-one-out cross validation technique.

σ \ h	1	2	3	5	8	10	15	20	25
0.1	0.53	0.73	0.83	0.92	0.98	0.98	0.95	0.98	1.00
0.2	0.47	0.73	0.78	0.74	0.94	0.94	0.94	1.00	1.00
0.5	0.33	0.56	0.67	0.78	0.78	0.89	0.88	0.94	0.97
1	0.22	0.52	0.67	0.60	0.69	0.72	0.86	0.90	0.94
2	0.20	0.41	0.44	0.67	0.59	0.67	0.73	0.84	0.91

Table 3: Classification results for the synthetic trajectories, with a HMM-based method, using the leave-one-out cross validation technique, for different values of h and σ (σ is the standard deviation of the added noise).

7.5 Results on the detection of unexpected events

We have conducted experiments on several real videos for the detection of different types of unexpected events. For the Formula One race video, we were able to detect incidents such as cars driving off the track (revealed by an abnormal trajectory shape) or intervention of the safety car (revealed by a quite different speed while the global trajectory shape remains unchanged). For the skiing competition, the objective was to detect falls of skiers. Fig.5 and Fig.6 respectively show three Formula One video sequences and two Alpine skiing race video sequences. Fig.6 also presents trajectories corresponding to a class (printed in blue) and to two unexpected events (printed in magenta and blue). In each case, the first one belongs to a regular event class while the other ones are examples of unexpected events. The criterion described in subsection 5.3 allowed us to correctly detect the unexpected events in all the processed examples. In Table 5 we supply the maximum intra-class distance R_i and the distance between the trajectory detected as unexpected event and the six considered classes C_i (presented in Fig.4), for several cases. Table 4 presents results corresponding to skiers trajectories, showing the difference between the maximum intra-class distance and the distance between unexpected events and this regular class. These results show that our HMM-based framework can be straightforwardly and successfully exploited for detecting unexpected events in videos.

Figure 5: Images from Formula One race video shots acquired by the same camera. Top row: images displaying instances of a regular class. Bottom rows: example of unexpected events (safety car, car driving off the track). The trajectories are overprinted on the images.

Figure 6: Images from Alpine skiing competition video shots acquired by the same camera. Trajectories are overprinted on the images. Top row: example of regular class. Bottom row: example of unexpected event (fall of a skier).

	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	status
$R_i + \sigma_i$	0.0753	0.1310	0.1180	0.0632	0.0225	0.0330	
Accident 1	0.2084	0.1427	0.1713	0.1306	0.0296	0.1992	detected
Veering off the track	0.1603	0.1991	0.3017	0.2068	0.0484	0.1556	detected
Safety car	0.2958	0.5200	0.6425	0.3088	0.2595	0.2788	detected
Accident 2	0.2474	0.3978	0.5141	0.2068	0.0716	0.2127	detected

Table 5: Detection thresholds are supplied in the second row for the (learnt) classes C_i used in the detection task of unexpected events. The following rows contain the distances between the unexpected events trajectories and the regular classes. The events 'Accident 1', 'Veering off the track' and 'Safety car' were shot by the camera corresponding to class 1, whereas 'Accident 2' corresponds to class 2. Last column shows the detection status.

	Class 1	Status
$R_1 + \sigma_1$	0.1434	
Accident	0.4412	detected
Skier veering off the pist	0.3307	detected

Table 4: Detection threshold is supplied in the second row for the class of ski trajectories used in the detection task of unexpected events. The following rows contain the distances between the unexpected events trajectories and the regular class. Last column shows the detection status.

8 Conclusion

We have proposed a trajectory-based HMM framework for video content understanding. We have shown that it is general and flexible enough to solve three tasks: unsupervised clustering of events, recognition of events corresponding to learnt classes of dynamic video contents, and detection of unexpected events. We have introduced appropriate local trajectory features invariant to translation, rotation and scale transformations, and we can reliably compute them in presence of noise. We have conducted an important set of comparative experiments both on synthetic examples and real videos (sports TV programs) with classification ground truth. We have shown that our method supplies accurate results and offers better performance and usability than other approaches such as SVM classification, histogram comparison or LCSS distance. Extensions of this work will investigate the hierarchical modeling of space-time groups of trajectories in association or in interaction to represent activities in videos.

REFERENCES

Bashir, F. I., Khokhar, A. A., and Schonfeld, D. (2007). Real-time motion trajectory-based indexing and retrieval of video sequences. *IEEE Trans. on Multimedia*, 9(1):58–65.

Boiman, O. and Irani, M. (2005). Detecting irregularities in images and in video. *IEEE Int. Conf. on Computer Vision, ICCV'05*, Beijing, Vol.1, pages 462–469.

C. Burges (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–167.

Buzan, D. , Sclaroff, S. , and Kollios, G. (2004). Extraction and clustering of motion trajectories in video. In *Proc. of the 17th Int. Conf. Pattern Recognition, ICPR'04*, pages 521–524, Cambridge, UK.

Chan, M. T., Hoogs, A. , Schmiederer, J. , and Peterson, M. (2004). Detecting rare events in video using semantic primitives with HMM. In *Proc. of the 17th Int. Conf. on Pattern Recognition, ICPR'04*,pages 150–154, Cambridge, UK.

Ford, J. , and Moore, J. (1998). Adaptive estimation of HMM transition probabilities. *IEEE Trans. on Signal Processing*, 46(5):1374–1385.

Perez, P. , Hue, C. , Vermaak, J. , and Gangnet, M. (2002). Color-based probabilistic tracking. *Proc. Europ. Conf. Computer Vision, ECCV'02*, Copenhaguen.

Gong, S. , and Xiang, T. (2003). Recognition of group activities using dynamic probabilistic networks. In *Proc. of the IEEE Int. Conf. on Computer Vision, ICCV'03*, pages 742– 749, Nice.

Hongeng, S. , Nevatia, R. , and Bremond, F. (2003). Video-based event recognition: Activity representation and probabilistic recognition methods. *Computer Vision and Image Understanding*, 96(2):129–162.

Hu, W. , Xie, D. , Fu, Z. , Zheng, W. , and Maybank, S. (2007). Semantic-based surveillance video retrieval. *IEEE Trans. on Image Processing*, 16(4):1168-1181.

Kokaram, A. , Rea, N. , Dahyot, R. , Tekalp, M. , Bouthemy, P. , Gros, P. , and Sezan, I. (2006). Browsing sports video (Trends in sports-related indexing and retrieval work). *IEEE Signal Processing Magazine*, 23(2):47–58.

Li, X. , Hu, W. , and Hu, W. (2006). A coarse-to-fine strategy for vehicle motion trajectory clustering. In *Proceedings of the 17th Int. Conf. on Pattern Recognition*, pages 591–594, Hong Kong.

Porikli, F. (2004). Trajectory distance metric using hidden Markov model based representation. In *PETS Workshop*, Prague.

Rabiner, L. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. IEEE*, 77(2):257–285.

Stauffer, C. , and Grimson, W. E. L. (2000). Learning patterns of activity using real-time tracking. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(8):747–757.

Vlachos, M. , Kollios, G. , and Gunopulos, D. (2002). Discovering similar multidimensional trajectories. In *Proc. of the 18th International Conference on Data Engineering*, San Jose.

Wang, X. , Tieu, K. , and Grimson, E. (2006). Learning semantic scene models by trajectory analysis. In *Proc. Eur. Conf. on Computer Vision, ECCV'06*, Graz.