A Real-Time System for Abnormal Path Detection

Simone Calderara*, Clara Alaimo*, Andrea Prati+, Rita Cucchiara*

* D.I.I., + Di.S.M.I., University of Modena and Reggio Emilia, Italy

Keywords: Abnormal path detection, video surveillance.

Abstract

This paper proposes a real-time system capable to extract and model object trajectories from a multi-camera setup with the aim of identifying abnormal paths. The trajectories are modeled as a sequence of positional distributions (2D Gaussians) and clustered in the training phase by exploiting an innovative distance measure based on a global alignment technique and Bhattacharyya distance between Gaussians. An on-line classification procedure is proposed in order to on-the-fly classify new trajectories into either "normal" or "abnormal" (in the sense of rarely seen before, thus unusual and potentially interesting). Experiments on a real scenario will be presented.

1 Introduction and Related Works

Recent advances in computational resources and algorithms have made distributed video surveillance very appealing to both the academia and the industry. As a consequence, there exist in the literature many works addressing some of or all the steps related to distributed video surveillance: from motion detection and moving object segmentation [4], to object tracking with occlusion handling [14], to fusion among multiple cameras, with either overlapped [1] or disjoint views [8], to higher-level reasoning modules to analyze the behaviors and the interactions, or to detect and classify events [7]. This latter step is of crucial importance for forensics and crime prevention/detection. In particular, the automatic detection and the a-posteriori retrieval of interesting events, trajectory/path analysis and human action recognition are nowadays frequently requested as tools for the aid of investigations.

Despite this considerable amount of papers and techniques, the focus of these proposals has been mainly on proposing innovative solutions capable to handle the most complex situation possible, with few (or none) attention to the real-time constraints or to the computational requirements in general. However, even if complexity is a requirement in order to propose significant advances with respect to the state of the art, real-time alarming is often a must in this type of systems, since off-line processing does not guarantee a timely response to relevant events. Obviously, a careful tuning of the trade-off between efficiency and accuracy must be achieved in order to preserve as much as possible the flexibility and the applicability of the system to different contexts.

With these premises, this paper describes a complete and full-working system for real-time detection of abnormal paths in real multi-camera scenarios. The system architecture (described in Section 2) consists of several static cameras with partially-overlapped views on which a "consistent labeling" module runs with the scope to obtain long and reliable trajectories of the people moving on a wide area. These trajectories are modeled as a sequence of positional distributions (2D Gaussians) and clustered in the training phase by exploiting an innovative distance measure based on a global alignment technique and Bhattacharyya distance between Gaussians. This allows the analysis of trajectories of different lengths, possibly affected by noise or segmentation errors. The main contribution of this work, however, resides in the on-line classification of new trajectories into either "normal" or "abnormal" (in the sense of rarely seen before, thus unusual and potentially interesting). This real-time approach is achieved thanks to an incremental updating of the distance matrix in an on-line fashion, i.e. for each new acquired point of the trajectory. Experiments on a real scenario have been carried out and the system is currently operating at our campus.

The real-time video surveillance systems presented in the past basically proposed quite complex techniques for fusing single (low-level) algorithms from multiple cameras, such as moving object detection and tracking [5]. Instead, the implementation of higher level tasks (e.g., trajectory/path classification) in real time has not been deeply explored. This section will mainly focus on related works in the field of (real-time) trajectory analysis, which is the main contribution of this paper.

Trajectory analysis has been studied in depth over the last years, especially for its application in people surveillance. Morris and Trivedi in [10] proposed a recent survey on state-of-art techniques for modeling, comparing and classifying trajectories in video surveillance. The simplest way to define a similarity measure between trajectories is the adoption of Euclidean distance between spatial coordinates as proposed in [3], while the Hausdorff distance was adopted by Junejo *et al.* [6]. However, both these measures only perform point-to-point comparison on trajectories of the same length and, additionally, the Euclidean distance requires the trajectories to have the same length, while Hausdorff distance does not need same length, but cannot distinguish the opposite directions. Chen et al. in [2] presented a method to compare the trajectories after projecting them in a null-space to obtain a representation insensitive to projective transformation of the trajectories themselves.

The similarity measures with or without alignment are typ-

ically defined in a statistical framework. Mecocci and Panozzo in [9] suitably modified the iterative *Altruistic Vector Quantization* algorithm to robustly cluster trajectories by pure spatial observations obtaining representative prototypes. The anomaly detection is based on fitting a spatial Gaussian on each prototype and statistically checking for fitness of new trajectory samples. In [6], Junejo *et al.* applied graph cuts to cluster trajectories with the Hausdorff distance. Porikli [12] proposed the use of a HMM-based similarity measure where each trajectory is modeled with a HMM and compared using the cross likelihood. The results are promising but, in general, a large amount of data is needed to avoid overfitting in the HMM training phase.

2 System Overview

The proposed system is mainly composed of two phases: the first has the objective to build the starting set of trajectory classes by performing the off-line learning of the trajectory model over a training set; the second phase works on-line and in real time, and aims at classifying new trajectories and continuously updating the classes and their prototypes. This latter continuous updating is crucial since in real and complex scenarios the "knowledge" of the scene and the known trajectory classes are neither a-priori known nor fixed. The scene layout can change resulting in new "normal" paths which were "abnormal" before. For this reason, we employ a *learn-and-predict* paradigm in which the knowledge (i.e., the trajectory clusters) are continuously updated.

The next Section will report a detailed description of the trajectory model which is learnt in the first phase and exploited for the classification in the second phase. This model is based on a sequence of spatial (x, y) coordinates on the ground plane. Thus, both the two phases above require that the data on which the clustering/classification is performed are extracted by the system.

Fig. 1 shows the scheme for the on-line testing phase, divided in the data extraction (left - common also to the off-line learning phase) and the trajectory classification (right - described in Section 3.3). Since the data extraction task aims at obtaining the points composing the trajectory, it requires to segment moving objects in all the cameras and to track them both in each single camera and across adjacent cameras.

Moving object detection and tracking from a single static camera is a well-known and almost-solved problem. Our system makes use of the approach proposed by us in [1]. The objects detected as moving are then tracked in each single view by means of an appearance-based algorithm proposed in [13]. Once each camera has processed the video stream and obtained the object tracks, there is the need to render the track labels/ids consistent among the different cameras: this step is crucial to keep track of the object when it moves across the fields of view of the different cameras and thus to obtain longer and more stable trajectories. This problem is also known as *consistent labeling* and we used our approach, proposed in [1], valid for cameras with partially-overlapped fields of view (FoVs), which adopts a geometric approach that exploits cameras' FoV relationships and constraints to impose identities consistency.

This procedure extracts from the multi-camera system the (x, y) coordinates on the ground plane for each tracked moving object in the scene. These data are then used for analyzing and classifying the object trajectories. The data points are first modeled using the trajectory model described in Section 3 and then normal behaviors are learnt by means of the iterative clustering procedure of Section 3.2.

During the classification new trajectories are compared with the prototypes (medoids) of the existing clusters computing the distances of each new trajectories with each of them. This distance computation is performed on-line, by updating the alignment table V (see Section 3.3) every time a new sample (i.e. a new point of the trajectory) is provided by the system. As a consequence, also the classification result (in either normal or abnormal) is continuously updated using a minimum distance classifier which assigns the new trajectory to the class at the minimum distance (Section 3.3). Finally, when the object exits from the scene (i.e. it is no more detected for several frames) its class assignment is used to update the class memberships and the corresponding prototypes.

3 Trajectory Model

The people trajectory projected on the ground plane is a very compact representation of patterns of movement, normally characterized by a sequence of 2D data ($\{(x_1, y_1), \cdots, (x_n, y_n)\}$ coordinates) and often associated with the motion status (punctual velocity or acceleration).

When large data are acquired in a real system they should be properly modeled to account for tracking errors, noise in the support point extraction and inaccuracies due to the multi-camera data fusion module. Positional trajectories must then be correctly extracted by the tracking system and analyzed in order to discriminate or aggregate different kinds of people behaviors. For these reasons a trivial model that simply performs a point-wise comparison in the rectified Euclidean plane will result extremely imprecise. We decided to adopt a model that employs statistics to model data points sequences, being consequently robust against measurement errors and data uncertainties, but imposing some constraint and limitation to achieve real-time performance. As stated before, this permits to achieve a good trade-off between efficiency and accuracy.

When observing a video surveillance scenario some paths are considerably more common than others. This is mainly due to two factors. First, the structure of the environment may condition significantly the way people move. Second, according to the scenario, people tend to reproduce frequent behaviors.

Given the k^{th} rectified trajectory projected on the ground plane $T^k = \left\{\mathbf{t_1^k} \dots \mathbf{t_{n_k}^k}\right\}$, where $\mathbf{t_i^k} = (x_i^k, y_i^k)$ with n_k the number of points of trajectory T^k, a bi-variate Gaussian centered on each data point $\mathbf{t_i^k}$ (i.e., having the mean equal to the point coordinates $\boldsymbol{\mu_k^i} = (x_i^k, y_i^k)$) and with fixed covariance matrix $\boldsymbol{\Sigma}$ can be defined as:

$$\mathcal{N}_i^k = \mathcal{N}(x, y \mid \boldsymbol{\mu_i^k}, \boldsymbol{\Sigma}) \tag{1}$$

An example of the fitting of Gaussians onto the trajectory

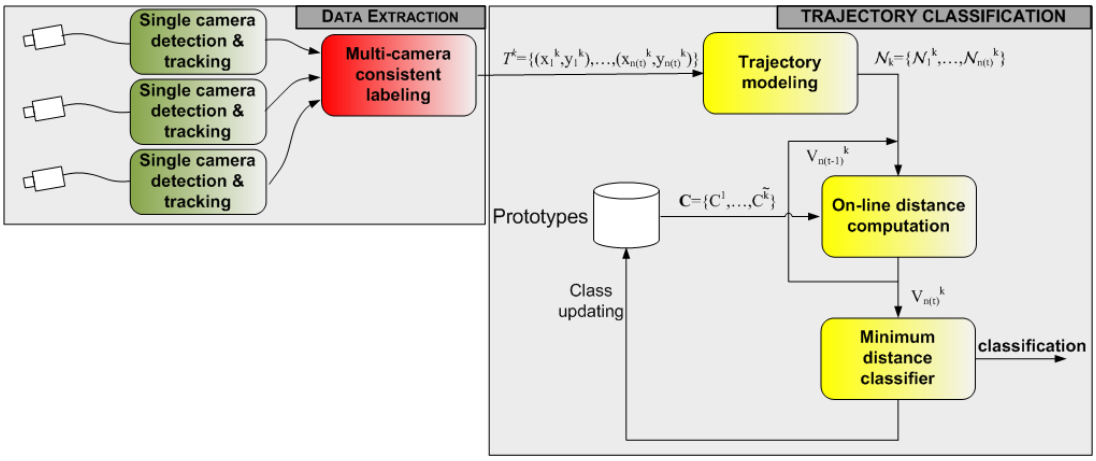

Figure 1. Diagram of the on-line testing phase.

points is shown in Fig. 2, where (a) shows an exemplar trajectory, (b) the 3D plot of the superimposed Gaussians and (c) the x-y projection of (b).

The main motivation for this modeling choice relies in the fact that when comparing two points belonging to different trajectories small spatial shifts may occur and trajectories never exactly overlap point-to-point. Using a sequence of Gaussians, one for each point, allows to build an envelope around the trajectory itself, obtaining a slight invariance against spatial shifts.

3.1 Sequence similarity measure

After assigning a Gaussian to each trajectory point, the trajectory can be modeled as a sequence of Gaussian distributions $\boldsymbol{\mathcal{N}}^k = \{\mathcal{N}_1^k, \mathcal{N}_2^k \ldots \mathcal{N}_{n_k}^k\}$, where each Gaussian $\mathcal{N}_i^k$ is modeled as in equation (1).

In order to compare two sequences an inexact matching technique is required. The main motivation resides in the fact that trajectories are never equal both in number and position of points. Small changes can occur between two similar sequences: for example, there may be some time stretches that result in sequences having different lengths; additionally, sequences may be piecewise-similar, sharing some common parts, but they can be different in other parts. In choosing the similarity measure it is desirable to gain control on the amount of common points that two sequences must share in order to be considered "similar".

For these motivations, the best way to compare two sequences is to identify the best alignment of the sequence data, based on a given point-to-point distance metrics. Point-to-point comparison can be made either directly on the data or by selecting a data representation which assigns a symbol (with a given "meaning") to each data and performing a symbol-to-symbol comparison.

Once a sequence of data/symbols is achieved, we can borrow from bioinformatics the method for comparing DNA sequences in order to find the best inexact matching between them, also accounting for gaps. Then, we propose to adopt the *global alignment*, specifically the well-known Needleman-Wunsch algorithm [11] for comparing sequences of Gaussians. A global alignment (over the entire sequence) is preferable over a local one, because preserves both global and local shape characteristics. Global alignment of two sequences S and T is obtained by first inserting spaces, either into or at the ends of S and T so that the length of the sequences will be the same; by doing this, every symbol (or space) in one of the sequences is matched to a unique symbol (or space) in the other.

The alignment is simply achieved by arranging the two sequences in a table, the first sequence row-wise and the second column-wise, starting from the base conditions:

$$\begin{aligned} V(a,0) &= \Omega(S_a, -) \\ V(0,b) &= \Omega(-, T_b) \end{aligned} \tag{2}$$

where Ω represents a suitable similarity measures, S_a and T_b are the symbols to be aligned, and $-$ indicates a zero-element or gap.

This is due to the fact that the only way to align the first i elements of the sequence S with zero elements of the sequence T (or viceversa) is to align each of the elements with a space in the sequence T.

Starting from these base conditions, the alignment is performed exploiting the recurrent equation of global alignment that computes the best alignment score for each subsequence of symbols:

$$V(a,b) = \max \begin{cases} V(a-1, b-1) + \Omega(S_a, T_b) \\ V(a-1, b) + \Omega(S_a, -) \\ V(a, b-1) + \Omega(-, T_b) \end{cases} \tag{3}$$

with $1 \leq a \leq n$ and $1 \leq b \leq m$ and where $V(a,b)$ is the score of the alignment between the subsequence of S up to the a^{th} symbol and the subsequence of T up to the b^{th} symbol.

In the case of symbol sequences that represent probability distributions, a proper symbol-to-symbol similarity measure must be defined in order to perform the global alignment. Among the possible metrics to compare probability distributions we chose to employ the Bhattacharyya coefficient, to

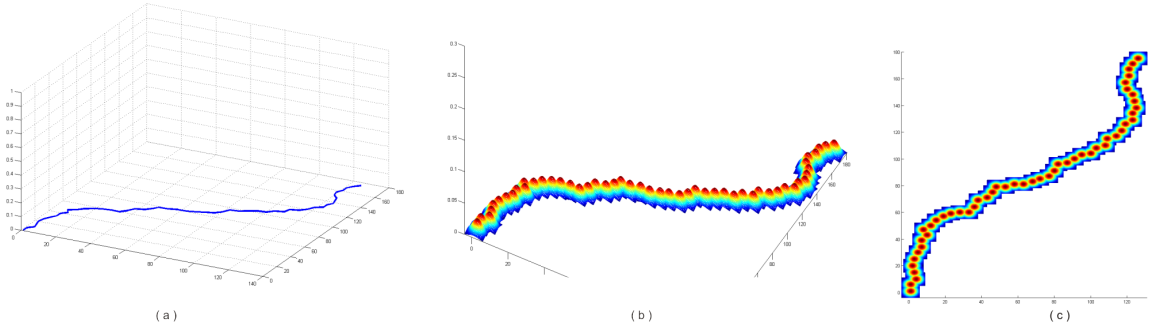

Figure 2. Example of the trajectory model.

measure the distance between the two normal distributions $\mathcal{N}_a^k$ and $\mathcal{N}_b^m$ corresponding to a^{th} and b^{th} symbols of sequences $\mathcal{N}^k$ and $\mathcal{N}^m$, respectively:

$$\begin{aligned} c_b\left(\mathcal{N}_a^k, \mathcal{N}_b^m\right) &= d_{BH}\left(\mathcal{N}\left(x, y|\boldsymbol{\mu}_a, \boldsymbol{\Sigma}_a\right), \mathcal{N}\left(x, y|\boldsymbol{\mu}_b, \boldsymbol{\Sigma}_b\right)\right) \\ &= \frac{1}{8}\left(\boldsymbol{\mu}_a - \boldsymbol{\mu}_b\right)^T \left(\overline{\boldsymbol{\Sigma}}\right)^{-1}\left(\boldsymbol{\mu}_a - \boldsymbol{\mu}_b\right) + \\ &\quad + \frac{1}{2}\ln\left(\frac{\det\overline{\boldsymbol{\Sigma}}}{\sqrt{\det\boldsymbol{\Sigma}_a \det\boldsymbol{\Sigma}_b}}\right) \end{aligned} \tag{4}$$

where $2 \cdot \overline{\boldsymbol{\Sigma}} = \boldsymbol{\Sigma}_a + \boldsymbol{\Sigma}_b$. Since in our case $\boldsymbol{\Sigma}_a = \boldsymbol{\Sigma}_b = \boldsymbol{\Sigma}$, we can rewrite the distance as:

$$c_b\left(\mathcal{N}_a^k, \mathcal{N}_b^m\right) = \frac{1}{8}\left(\boldsymbol{\mu}_a - \boldsymbol{\mu}_b\right)^T \boldsymbol{\Sigma}^{-1}\left(\boldsymbol{\mu}_a - \boldsymbol{\mu}_b\right) \tag{5}$$

Once the coefficient is computed, assuming that two distributions are sufficiently similar if the coefficient is above 0.5 and that the score for perfect match is +2, whereas the score (penalty) for the perfect mismatch is -1 (that are the typical values used in DNA sequence), it is trivial to build the similarity score Ω of equation (3):

$$\Omega\left(\mathcal{N}_a^k, \mathcal{N}_b^m\right) = \begin{cases} 2 \cdot (c_B) & \text{if} \quad c_B \geq 0.5 \\ 2 \cdot (c_B - 0.5) & \text{if} \quad c_B < 0.5 \\ -0.3 & \text{if} \quad S_a \text{ or } T_b \text{ are gaps} \end{cases} \tag{6}$$

3.2 Learning the Normal Paths

The trajectory analysis goes through two main phases: a training step to learn the classes of similarity and the classification of new samples (Fig. 1). This latter classification task can be specialized into a two-class classification problem to search for "normal"/"abnormal" trajectories in a statistical sense. The problem is to find trajectories that are "abnormal" in the sense that they have been never (or rarely) seen before (meaning of "abnormal" must not be considered as a high-level concept). Thus, a trajectory can be considered as abnormal if it belongs to a class with few elements or if it creates a new class.

During an initial off-line training step, a training set of trajectories is analyzed and clustered into groups of similarity. Fig. 4.(a,b) shows the training set used in our experiments. As previously described in Section 3, we model each trajectory T^j as a sequence of bivariate Gaussian distributions centered in the points coordinates in the rectified ground plane.

Each trajectory is coded in a sequence of symbols S^j, with each symbol S_a^j corresponding to a specific Gaussian distribution, $\mathcal{N}_a^j$ according to Section 3. Eventually, the findings reported in Section 3.1 are used to compute the similarity $\Omega\left(T^i, T^j\right)$ based on sequence alignment and a clustering algorithm is exploited to group trajectories into classes. Hereinafter we will use interchangeably the trajectory and its symbolic representation in the Ω measure to keep the notation light, but the similarity measure is obviously computed, as previously stated, on the symbolic representation of the trajectory and consequently on the Gaussians.

The choice of the clustering algorithm is not critical: we only need a method without a fixed number of clusters and capable of creating clusters of different cardinality, even containing a single sample. This is required to achieve the maximum generality by allowing also "abnormal" trajectories to be included in the training set. In such a case, those trajectories will be clustered in small classes, due to their diversity.

The score for the best global alignment of two sequences reported in equation (6) can be converted in a proper similarity measure $\Omega\left(T^i, T^j\right)$. This measure is used to cluster the trajectories in the training set by using *k-medoids* algorithm. This is a suitable modification of the well-known k-means algorithm which has the appreciable characteristic to compute, as prototype of the cluster, the element that minimizes the sum of intra-class distances. In other words, let us suppose to have a training set $TS = \left\{T^1, \cdots, T^{N_t}\right\}$ composed of N_t trajectories and set $i = 0$ and $k(0) = N_t$. As initialization, each trajectory is chosen as prototype (medoid) of the corresponding cluster. The k-medoids algorithm iteratively assigns each trajectory T^j to the cluster $C^{\widetilde{m}}$ at the minimum distance d, i.e. given $k(i)$ clusters $C^1, \cdots, C^{k(i)}$ and the corresponding medoids $M^1, \cdots, M^{k(i)}$, $\widetilde{m} = \underset{m=1,\cdots,k(i)}{\arg\min}\, d\left(T^j, T^{M^m}\right)$, where T^{M^m} is the trajectory corresponding to the medoid M^m. Once all the trajectories have been assigned to the correct cluster, the new medoid M^s for each cluster C^s is computed as that one which minimizes the intra-cluster distances, i.e. $T^{M^s} \equiv T^{\widetilde{p}} = \underset{\forall T^p \in C^s}{\arg\min} \sum_{\forall T^r \in C^s} d\left(T^p, T^r\right) = \underset{\forall T^p \in C^s}{\arg\max} \sum_{\forall T^r \in C^s} \Omega\left(T^p, T^r\right)$.

However, one of the limitations of k-medoids (as well as k-means) clustering is the choice of k. For this reason, we propose to use an *iterative k-medoids* algorithm, which iteratively

merges (starting from a number of clusters equal to the number of trajectories) similar clusters until convergence. In this way, the "optimal" number $\widetilde{k}$ of medoids is obtained.

The described approach obtains a robust unsupervised classification of trajectories, clustered in a variable number of similarity classes. Clusters containing a single or few trajectories represent the classes of abnormal trajectory shapes.

3.3 On-line Trajectories Classification

The described approach obtains a robust unsupervised classification of trajectories. They are clustered in a variable number of similarity clusters. Clusters containing a single or few trajectories represent the case of abnormal trajectories.

Then, during the on-line phase, whenever a new trajectory T^{new} is collected, its statistical model is computed and compared to the cluster medoids. Based on this comparison, the trajectory can be classified as either belonging to an existing cluster or representing a new cluster (a class of trajectories never seen before).

Since the classification is performed on-line, it is unfeasible to obtain real-time performance computing the distance among the new trajectory T^{new} and the clusters prototypes when the trajectory is finished, e.g. when the person exits the field of view of the cameras. This can be explained by the fact that, even using dynamic programming to speed up the distance computation process, the computational burden of computing distances is concentrated only in some specific moments and not well distributed. Another major drawback of classifying the trajectories once they are finished emerges when an abnormal trajectory occurs and must be quickly detected and signaled.

To overcome these issues we chose to adopt an on-line modification of the global alignment-based distance measure, that updates the distance as long as new trajectory points are provided by the tracking system. In more details, given a prototype trajectory (cluster center/medoid) C^h and a new trajectory $T^k_{n(t)}$ at time t (where $n(t)$ represents the number of points of the trajectory at time t), we store the last column of the alignment table V introduced in Section 3.1, as described in Fig. 3. Recalling the *recurrent relationship of the global alignment* reported in eq. (3) it is possible to observe that the score of aligning the sequence C^h with the subsequence $T^k_{n(t+1)}$ can be derived using the column of table V containing the costs of aligning sequence C^h with subsequence $T^k_{n(t)}$:

$$V\left(a, n(t+1)\right) = \max \begin{cases} V\left(a-1, n(t)\right) + \Omega\left(\mathcal{N}^h_a, \mathcal{N}^k_{n(t+1)}\right) \\ V\left(a-1, n(t+1)\right) + \Omega\left(\mathcal{N}^h_a, -\right) \\ V\left(a, n(t)\right) + \Omega\left(-, \mathcal{N}^k_{n(t+1)}\right) \end{cases} \tag{7}$$

where $\mathcal{N}^h_a$ and $\mathcal{N}^k_b$ are the a^{th} and the b^{th} Gaussians of the statistical model of C^h and T^k, respectively.

Using the proposed on-line distance updating allows real-time performance by balancing the computation efforts and distributing them uniformly in time. Once the distance is efficiently computed the new trajectory can be classified as normal/abnormal depending on the cardinality of the most similar cluster. Moreover, with the adopted learn-and-predict approach, at the beginning a trajectory different from the others must be included in a new class and considered abnormal. This label is maintained until the class cardinality is low but if that trajectory shape is detected often, it should be considered normal, since in our scenario the model of normality is neither a-priori known nor fixed. Finally, to avoid old and rare trajectories affecting our model, clusters with small cardinality and with no new trajectories assigned for a fixed-length time window are dropped.

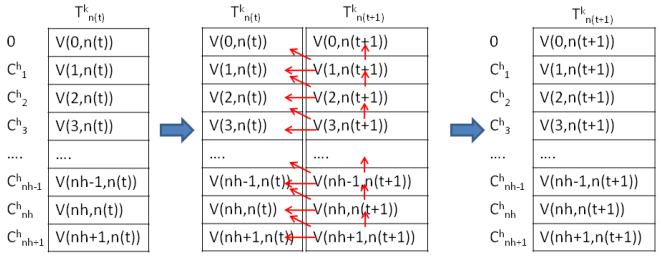

Figure 3. Scheme of the on-line updating of the alignment table V.

4 Experimental results

We tested our system in a two-cameras setup at our campus. We performed the learning phase (described in Section 3.2) during an ordinary working day on a dataset of about 900 trajectories, by learning the most frequent behaviors in the chosen scenario as shown in Fig. 4.

After the learning phase, we chose to discard all the classes having a low cardinality, i.e. representing infrequent paths: this is motivated by the fact that during on-line classification a reduced number of prototypes allows to reduce the time for distance computation for every new trajectories without affecting the normal/abnormal classification accuracy. During the tests, we ran our system for several working days and labeled manually the trajectories that appeared as abnormal.

Additionally, we chose to accept system classification only on sufficiently long trajectories to ensure a reliable system response. The system accuracy on a corpus of about 3000 trajectories was around 93% operating at 15 fps on a modern PC equipped with Core Duo processor. In Fig. 5.a exemplars of correct normal paths are depicted, while in Fig. 5.b abnormal paths are shown.

In conclusion, this system demonstrates that even a simplified approach like the fitting of a sequence of Gaussians on the positional data representing a trajectory can be sufficiently robust to allow the detection of abnormal paths. In addition, it allows both real-time performance and the on-line updating of the classification result.

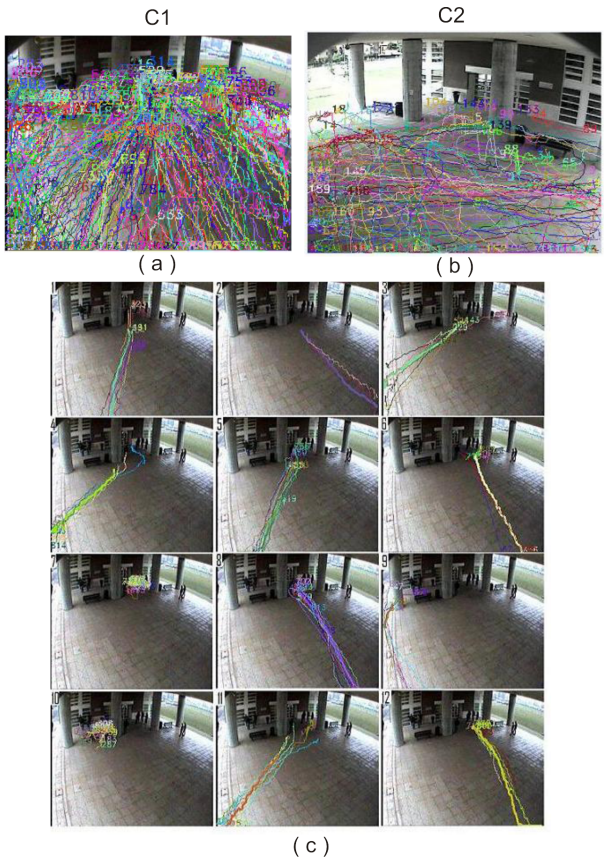

Figure 4. In (a) and (b) is shown the training set used during the learning stage. (c) shows the obtained most frequent behaviors projected on the $C1$ view.

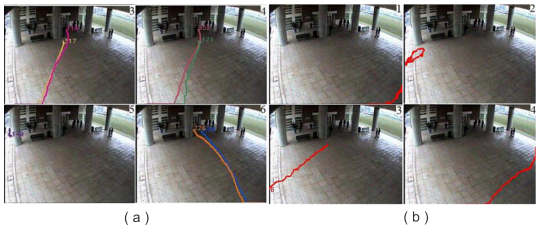

Figure 5. Examples of normal behavior correctly detected (a) and trajectories detected as abnormal (b).

References

[1] S. Calderara, A. Prati, and R. Cucchiara. Hecol: Homography and epipolar-based consistent labeling for outdoor park surveillance. *Computer Vision and Image Understanding*, 111(1):21–42, July 2008.

[2] X. Chen, D. Schonfeld, and A. Khokhar. Robust null space representation and sampling for view invariant motion trajectory analysis. In *Proc. of IEEE Int'l Conference on Computer Vision and Pattern Recognition*, 2008.

[3] H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E.J. Keogh. Querying and mining of time series data: experimental comparison of representations and distance measures. *Proceedings of the VLDB Endowment*, 1(2):1542–1552, 2008.

[4] W. Hu, T. Tan, L. Wang, and S. Maybank. A survey on visual surveillance of object motion and behaviors. *IEEE Trans. on Systems, Man, and Cybernetics - Part C*, 34(3):334–352, August 2004.

[5] O. Javed, Z. Rasheed, O. Alatas, and M. Shah. Knight: a real time surveillance system for multiple and non-overlapping cameras. In *Multimedia and Expo, 2003. ICME '03. Proceedings. 2003 International Conference on*, volume 1, pages I–649–52 vol.1, July 2003.

[6] I.N. Junejo, O. Javed, and M. Shah. Multi feature path modeling for video surveillance. In *Proc. of Int'l Conference on Pattern Recognition*, volume 2, pages 716– 719, Aug. 2004.

[7] G. Lavee, L. Khan, and B.M. Thuraisingham. A framework for a video analysis tool for suspicious event detection. *Multimedia Tools Appl.*, 35(1):109–123, 2007.

[8] C. Madden, E.D. Cheng, and M. Piccardi. Tracking people across disjoint camera views by an illumination-tolerant appearance representation. *Mach. Vis. Appl.*, 18(3-4):233–247, 2007.

[9] A. Mecocci and M. Pannozzo. A completely autonomous system that learns anomalous movements in advanced videosurveillance applications. In *Proc. of IEEE Int'l Conference on Image Processing*, volume 2, pages 586–589, Sept 2005.

[10] B.T. Morris and M.M. Trivedi. A survey of vision-based trajectory learning and analysis for surveillance. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(8):1114–1127, August 2008.

[11] S.B. Needleman and C.D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–453, 1970.

[12] F.M. Porikli and T. Haga. Event detection by eigenvector decomposition using object and frame features. In *Proc. of Computer Vision and Pattern Recognition (CVPR) Workshop*, volume 7, pages 114–121, 2004.

[13] R. Vezzani and R. Cucchiara. Ad-hoc: Appearance driven human tracking with occlusion handling. In *First International Workshop on Tracking Humans for the Evaluation of their Motion in Image Sequences (THEMIS'2008), in conjunction with BMVC 2008*, 2008.

[14] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Comput. Surv.*, 38(4):13, 2006.