Trajectory-Based Anomalous Event Detection

Claudio Piciarelli, *Member, IEEE*, Christian Micheloni, *Member, IEEE*, and Gian Luca Foresti, *Senior Member, IEEE*

***Abstract*—During the last years, the task of automatic event analysis in video sequences has gained an increasing attention among the research community. The application domains are disparate, ranging from video surveillance to automatic video annotation for sport videos or TV shots. Whatever the application field, most of the works in event analysis are based on two main approaches: the former based on explicit event recognition, focused on finding high-level, semantic interpretations of video sequences, and the latter based on anomaly detection. This paper deals with the second approach, where the final goal is not the explicit labeling of recognized events, but the detection of anomalous events differing from typical patterns. In particular, the proposed work addresses anomaly detection by means of trajectory analysis, an approach with several application fields, most notably video surveillance and traffic monitoring. The proposed approach is based on single-class support vector machine (SVM) clustering, where the novelty detection SVM capabilities are used for the identification of anomalous trajectories. Particular attention is given to trajectory classification in absence of *a priori* information on the distribution of outliers. Experimental results prove the validity of the proposed approach.**

***Index Terms*—Anomaly detection, event analysis, support vector machines (SVMs), trajectory clustering.**

I. Introduction

IN the last years, several works have been proposed on event analysis. Partial surveys on this topic can be found in the reviews by Buxton [1] and Hu *et al.* [2] in the field of video surveillance. Even though many different approaches have been developed, they can generally be classified in two categories, often referred to with the terms *explicit event recognition* and *anomaly detection*.

In the explicit event recognition approach, the system has an explicit knowledge of the events that must be identified. Once an event is detected, it can be properly labeled with a semantic description. The fundamental part of an explicit recognition system is thus an *a priori* knowledge base, where all the information about the recognizable events is stored, and the system behaves as a "parser" matching the incoming data with predefined templates found in the knowledge base. Explicit event recognition has its primary strength in its expressiveness, because it allows the modeling of heterogeneous events by labeling them with high-level semantic descriptions, thus giving a real interpretation of the analyzed scene. The main drawback of this approach is the need of a prior modeling for each possible instance of any event of interest. In real-world scenarios with uncontrolled and uncooperative environments, this could imply an exponential growth of the number of models as the scene complexity increases. Because of the nature of the explicit event recognition task, it is not surprising that most works on this topic are based on stochastic parsers for the identification of known patterns of atomic events, as in the works by Ivanov and Bobick [3], Minnen *et al.* [4] or Moore and Essa [5]. Similar to a parser-based approach is the work of Vu *et al.* [6], even though in their case the language used to express complex events is not a full grammar, but rather a set of subevents subject to temporal and logical constraints. Other works are based on the integration of prior knowledge and low-level data acquired by video processing (change detection, object tracking, etc.) as in the works by Ayers and Shah [7] or Medioni *et al.* [8]. Wada and Matsuyama proposed a work in which raw image data are directly used to infer behavior recognition, without intermediate processing as object detection or tracking [9]. Other works use *a priori* defined knowledge base partially modeled using machine learning techniques rather than being manually defined as in a stochastic grammar. Most of these works rely on the use of hidden Markov models for event recognition, as in the works by Oliver *et al.* [10], Galata *et al.* [11], Brand and Kettnaker [12], and Bashir *et al.* [13]. Finally, other works are based on more general stochastic models, such as Bayesian networks, as in Hongeng *et al.* [14] or Moënne-Loccoz *et al.* [15].

The other popular approach to event analysis is based on anomaly detection. In this case, the system does not require a prior knowledge base about the events that should be recognized. On the contrary, the event models are automatically built by the system itself in an unsupervised way, as the data are acquired. The models have a probabilistic nature, in the sense that the system is able to detect the most frequent patterns of activity occurred in the scene; the output of the event analysis process is no more an explicit description of the detected event (this would require a supervised learning), but only a measure of probability of that event, allowing to focus on the most rare ones (anomalies). A commonly used approach in this case is based on the clustering of the trajectories of the detected moving objects; the obtained clusters are then used as a normality model for anomaly detection. For a survey on trajectory (and more generally time series) clustering, see [16]. The work by Johnson and Hogg [17] was probably one of the first researches in this direction, using vector quantization for the compact representation of trajectories and multilayer neural networks for the identification of common patterns. The same authors later published another work on the same topic [18], where the probability density function of common patterns was approximated with a mixture of multivariate Gaussian densities. Another notable work on trajectory analysis has been done by Stauffer and Grimson [19], where

Manuscript received February 25, 2008; revised June 20, 2008. First published September 26, 2008; current version published October 29, 2008. This work was supported in part by the Italian Ministry of University and Scientific Research within the framework of the project "Ambient Intelligence: Event Analysis, Sensor Reconfiguration and Multimodal Interfaces" (2007–2008). This paper was recommended by Associate Editor D. Schonfeld.

The authors are with the Department of Mathematics and Computer Science, University of Udine, Udine 33100, Italy (e-mail: claudio.piciarelli@dimi.uniud.it).

Digital Object Identifier 10.1109/TCSVT.2008.2005599

Authorized licensed use limited to: IEEE Xplore. Downloaded on December 12, 2008 at 08:04 from IEEE Xplore. Restrictions apply.

vector-quantized trajectories are analyzed in terms of cooccurrence matrices of vector prototypes. Makris and Ellis [20], [21] proposed a method for labeling the scene with topological information, which is then used within a Bayesian approach to detect anomalous trajectories. Piciarelli and Foresti [22] proposed a trajectory clustering algorithm especially suited for online anomaly detection, which is a main issue in surveillance systems. Other relevant works on trajectory analysis for event detection include the ones by Porikli [23], which is based on eigenvalue decomposition of affinity matrices in order to find the correct number of clusters and compute conformity scores for anomaly detection; Hu *et al.* [24] use a hierarchic clustering of trajectories depending on spatial and temporal information; Lou *et al.* [25] use dynamic clustering techniques for traffic analysis; Morris and Hogg [26] analyze trajectories in order to detect interactions with known objects in the environment; Wang *et al.* [27] define an ad hoc trajectory similarity measure and an associated clustering technique; and Dee and Hogg [28] detect anomalies using inexplicability scores, a measure of how much a trajectory can be explained in terms of simple goals.

In this paper, we address the problem of trajectory analysis for anomaly detection using support vector machines (SVMs). Although SVMs have been a widely used tool for classification and clustering problems and can explicitly address the problem of anomaly (or novelty) detection [29], only few trajectory clustering papers rely on SVM, such as [30]. In a preliminary work, we proposed an anomalous trajectory detection technique based on single-class SVM [31] and we faced the problem of choosing the best training parameters for optimal clustering. The proposed solution, however, had a high computational complexity; in this work, we propose an alternative, faster technique for SVM training in presence of outliers. Moreover, we present further experimental results proving the validity of our approach.

This paper is structured as follows. In Section II, a short review of the SVM theory is given to clarify some concepts that will be recalled later in this paper. Section III discusses the possible applications of SVM techniques to trajectory analysis, and some preliminary results are given. Section IV addresses the problem of tuning SVM in presence of an unknown number of outliers, which is a common scenario in practical applications. Finally, Section V gives some experimental results proving the validity of the proposed approach.

II. Support Vector Machines Theory

SVMs are a set of classification and regression techniques based on the concepts of statistical learning theory and risk minimization, initially proposed by Vapnik *et al.* [32] and later extended to the nonlinear case with the introduction of kernel methods [33]. Single-class SVMs [29] are a particular type of SVM well suited for anomaly detection tasks, because they consider the training data as elements drawn from a single probability distribution, whose support has to be found while possibly discarding outliers. This approach can naturally be applied to anomalous trajectory detection: the detected trajectories are initially transformed in fixed-dimension feature vectors; then the training data are clustered using a single-class SVM, thus detecting the hypervolume in the feature space containing all the normal trajectories. Identifying anomalous trajectories is just a matter of checking if a new trajectory falls outside the computed hypervolume. To better understand the remaining sections of this work, a brief description of the SVM theory is here given. For further details on SVMs, see [34] and [35].

The main idea at the basis of SVM theory arises from the application of statistical learning theory results to linear classifiers. Let us consider the case of two-class hyperplane classifiers in some dot product space $\mathcal{H}$

$$(\mathbf{w} \cdot \mathbf{x}) + b = 0, \qquad \mathbf{w}, \mathbf{x} \in \mathcal{H}, \qquad b \in \mathbb{R} \tag{1}$$

corresponding to the decision function

$$f(\mathbf{x}) = \operatorname{sgn}\left((\mathbf{w} \cdot \mathbf{x}) + b\right)$$

as depicted in Fig. 1. Statistical learning theory states that the optimal classifier can be found by maximizing the *margin*, which is defined as the distance along the direction of $\mathbf{w}$ between the two hyperplanes parallel to the classification plane and passing through the points in the two class sets nearest to the classification plane (the dashed lines in Fig. 1). This can be expressed as a minimization problem

$$\begin{aligned} \min_{\mathbf{w}} \quad & \frac{1}{2}\|\mathbf{w}\|^2 \\ \text{subject to} \quad & y_i((\mathbf{w} \cdot \mathbf{x}_i) + b) \geq 1, \qquad i = 1, \ldots, m \end{aligned}$$

where m is the number of training data and $y_i \in \{-1, +1\}$ is the expected label for each element. The problem can be solved introducing the Lagrangian multipliers $\alpha_i \geq 0$ and setting to zero the partial derivatives of the Lagrangian with respect to (w.r.t.) the primal variables $\mathbf{w}$ and b, thus leading to the dual optimization problem

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & W(\boldsymbol{\alpha}) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) \\ \text{subject to} \quad & \alpha_i \geq 0, \qquad i = 1, \ldots, m \\ & \sum_{i=1}^{m} \alpha_i y_i = 0 \end{aligned} \tag{2}$$

which can be solved using standard quadratic optimization techniques. The decision function is now defined as

$$f(\mathbf{x}) = \operatorname{sgn}\left(\sum_{i=1}^{m} y_i \alpha_i (\mathbf{x} \cdot \mathbf{x}_i) + b\right) \tag{3}$$

where α_i are solutions of (2) and b can be found knowing that, for any $\mathbf{x}$ lying on the margin borders, the following equation holds:

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1 = 0.$$

The points lying on the margin borders are called *support vectors* and they are associated to nonzero values of α_i. This implies that the final solution (3) is *sparse*, being defined only in terms of a small subset of the data used in the training step.

The capability of solving only linear classification problems and the definition in a dot product space are the main drawbacks of this approach. Both problems can be addressed by defining a map $\Phi : \mathcal{X} \rightarrow \mathcal{H}$ from the nonempty set of the original input data $\mathcal{X}$ to a dot product space $\mathcal{H}$ (*feature space*) where

Authorized licensed use limited to: IEEE Xplore. Downloaded on December 12, 2008 at 08:04 from IEEE Xplore. Restrictions apply.

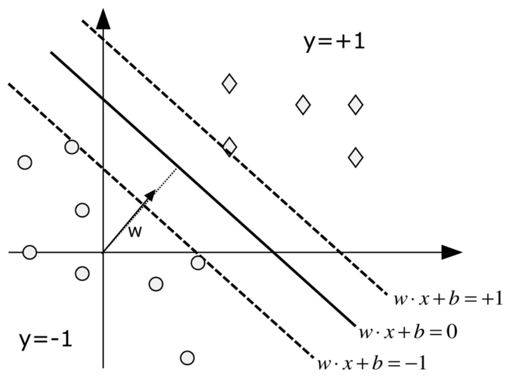

Fig. 1. Optimal classification hyperplane separating two classes. The optimal classification function is found by maximizing the distance between the two dashed hyperplanes in the direction of $\mathbf{w}$ (the *margin*).

the problem has a linear solution. The fundamental observation is that, in the dual problem (2) and in the decision function (3), the only operations performed in $\mathcal{H}$ are dot products. The classifier can thus be trained without doing any explicit computation in $\mathcal{H}$ if an explicit formula for dot products in $\mathcal{H}$ is given. This formula is traditionally called *kernel*, defined as $k(x,x') = \Phi(x) \cdot \Phi(x')$. Using kernels, the decision function (3) becomes

$$f(x) = \operatorname{sgn}\left(\sum_{i=1}^{m} y_i \alpha_i k(x, x_i) + b\right) \tag{4}$$

where $\boldsymbol{\alpha}$ is a solution of the optimization problem

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & W(\boldsymbol{\alpha}) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j k(x_i, x_j) \\ \text{subject to} \quad & \alpha_i \geq 0, \qquad i \in \{1 \ldots m\} \\ & \sum_{i=1}^{m} \alpha_i y_i = 0. \end{aligned} \tag{5}$$

Similar techniques can also be applied to nonclassification tasks such as anomaly detection. Given a set of unlabeled measures generated according to an unknown probability distribution P, single-class SVMs are used to estimate the support of a particular quantile of P. The goal is to find an appropriate region in the feature space $\mathcal{X}$ containing most of the data drawn from P, possibly leaving outliers outside this region. In the SVM framework, this can be obtained by searching for a decision hyperplane in the feature space $\mathcal{H}$, which maximizes its distance from the origin, while only a small fraction of data (the outliers) falls between the hyperplane and the origin. This can be expressed in terms of the constrained minimization problem

$$\begin{aligned} \min_{\mathbf{w},\boldsymbol{\xi},\rho} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + \frac{1}{\nu n}\sum_i \xi_i - b \\ \text{subject to} \quad & \mathbf{w} \cdot \Phi(\mathbf{x}_i) \geq b - \xi_i, \qquad \xi_i \geq 0 \end{aligned} \tag{6}$$

where $\mathbf{x}_i \in \mathcal{X}, i \in [1 \ldots n]$ are n training data in the data space $\mathcal{X}$, $\Phi : \mathcal{X} \rightarrow \mathcal{H}$ is the function mapping vectors $\mathbf{x}_i$ in the feature space $\mathcal{H}$, and $(\mathbf{w} \cdot \Phi(\mathbf{x})) - b = 0$ is the decision hyperplane in $\mathcal{H}$. In the minimization process, outliers are linearly penalized by the slack variables ξ_i, whose weight is controlled by the parameter $\nu \in (0,1]$. Introducing the Langrangian multipliers α_i, the minimization problem (6) can be reformulated in its dual form

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j) \\ \text{subject to} \quad & 0 \leq \alpha_i \leq \frac{1}{(\nu n)} \\ & \sum_i \alpha_i = 1. \end{aligned} \tag{7}$$

Values α_i can be found by solving the problem with standard quadratic programming methods; $\mathbf{w}$ is given by

$$\mathbf{w} = \sum_i \alpha_i \Phi(\mathbf{x}_i) \tag{8}$$

and, for any vector $\Phi(\mathbf{x}_i)$ with $\alpha_i \neq 0$, the following equations hold:

$$b - \xi_i = (\mathbf{w} \cdot \Phi(\mathbf{x}_i)) = \sum_j \alpha_j k(\mathbf{x_i}, \mathbf{x_j}) \tag{9}$$

with $\xi_i > 0$ for outliers and $\xi_i = 0$ for support vectors lying on the decision plane. The decision function in the data space $\mathcal{X}$ is thus defined as

$$\begin{aligned} f(\mathbf{x}) &= \operatorname{sign}((\mathbf{w} \cdot \Phi(\mathbf{x})) - b) \\ &= \operatorname{sign}\left(\sum_i \alpha_i k(\mathbf{x}, \mathbf{x_i}) - b\right). \end{aligned} \tag{10}$$

The solution is again sparse because $\alpha_i = 0$ for any $\mathbf{x}_i$ lying within the support region identified by the decision function: the majority of the training vectors do not contribute to the definition of the decision function. For the other vectors (*support vectors*), $0 < \alpha_i < 1/(\nu n)$ if the vector lies on the decision hyperplane, and $\alpha = 1/(\nu n)$ if the vector is an outlier.

The parameter ν has an intuitive meaning because it tunes the number of acceptable outliers. From (6), it can be seen that, when $v \rightarrow 0$, the outliers' penalization factors grow to infinity, thus leading to a hard margin solution, where no outliers are allowed at all. On the other hand, if $\nu = 1$, the constraints in (7) allow a single solution in which all the α_i are saturated to the maximum value $1/n$, thus all the vectors are outliers. More specifically, it can be proven that ν is an upper bound for the fraction of outliers and a lower bound for the fraction of support vectors (e.g., with $\nu = 0.1$, at least 10% of the training vectors will be support vectors, and at most 10% will be outliers).

III. SVM Trajectory Clustering

In this section, we will explore the possibility of using SVMs for event detection systems. The basic idea is to extract the trajectories of moving objects from video sequences, and cluster together groups of trajectories sharing similar features while leaving out the anomalous ones; testing a new trajectory for anomaly detection will thus be only a matter of comparing the new trajectory with the cluster model. Because a single class

Authorized licensed use limited to: IEEE Xplore. Downloaded on December 12, 2008 at 08:04 from IEEE Xplore. Restrictions apply.

(the normal trajectories) is searched for and the aim is to find outliers for that class, single-class SVMs seem to be well suited for this problem, due to their ability to detect a region in the data space enclosing the training data while possibly excluding outliers.

First, a proper input space must be defined. Trajectories are generally represented by variable-length sequences of 2-D coordinates, but the most commonly used kernels are defined on fixed-dimension spaces. A possible choice would be to define an ad hoc kernel working on sequences, and some efforts have recently been done in this direction, e.g., with the definition of *string kernels* [36]; however, in this work, we have chosen to work with standard kernels. This implies that trajectories must be represented by fixed-dimension feature vectors, something we obtained by trajectory subsampling (all the experiments in this work have been performed by evenly subsampling trajectories into a list of 16 2-D coordinates). Trajectories are also smoothed with a running-average filter before subsampling in order to remove the noise. Note that in this case we consider only spatial features, but other features could also be included, as long as the resulting feature vector has a fixed size. As an example, one of the last experimental results in Section V includes temporal information.

Regarding the choice of an appropriate kernel, it has been proven that all the traditionally used kernels (Gaussian, polynomial, and sigmoidal) have very similar performances given that the proper parameters are chosen [34]. Because we are working with spatial features, the Gaussian kernel seems a natural and intuitive choice, since it explicitly takes under consideration the Euclidean distance between feature vectors

$$k(x_1, x_2) = \exp\left(-\frac{\|x_1 - x_2\|^2}{2\sigma^2}\right), \qquad (x_1, x_2) \in \mathcal{X} \times \mathcal{X}.$$

The choice of σ heavily influences the final result because it represents the "scale factor" at which the data should be clustered. However, the scale factor is more related to the type of data (e.g., vehicle trajectories in a specific road, etc.) rather than on specific instances of data sets: σ could be estimated only once for each given scenario, without the need of reestimation when new data sets from the same scenario need to be processed. The choice of the SVM parameter ν is more critical, because it depends on the specific data set and thus cannot be estimated *a priori*. In Section IV, we propose a method to avoid an explicit choice of ν by automatically identifying the outliers in the training data.

A. Preliminary Tests

We will now present some preliminary experimental results on trajectory clustering with SVMs. The first test is based on labeled data sets, where training trajectories are grouped in several known clusters. Multiclass SVMs are trained using some training sets, and their generalization capabilities are evaluated on the corresponding test sets. Being a supervised learning problem, this classification approach cannot be directly applied to anomaly detection, and it has mainly been performed to confirm the validity of the choices described in the this section about trajectory representation and choice of kernel. If the test succeeds in separating different trajectories in different clusters, this implies that linear separability of the data in the feature space $\mathcal{H}$ is possible. This is not obvious, because $\mathcal{H}$ depends both on the initial feature space and on the kernel used; a success in this test does not imply good results in the final proposed method, but a failure would be a strong indicator that the choices described in the previous section could not lead to good results, whatever SVM technique is used. Tests have been done with increasing number of classes, ranging from a minimum of 2 to a maximum of 10. For each case, 50 different training and test sets have been randomly generated, each one with 100 trajectories in each class, for a total of 900 data sets. Each training set has been used for the training phase of a multiclass SVM with a Gaussian kernel ($\sigma = 0.5$), whose performances have been measured on the corresponding test set. Results are shown in Table I, reporting the total number of misclassifications in each run of 50 tests. The good results (with a worst case of 0.04% of classification errors) confirm that the combination of a Gaussian kernel with subsampling-based spatial trajectory representation can be a good choice for trajectory analysis with SVMs.

TABLE I
EXPERIMENTAL RESULTS WITH SUPERVISED SVM CLASSIFICATION

# clust.	Classification errors
2	0/10000 (0%)
3	0/15000 (0%)
4	0/20000 (0%)
5	0/25000 (0%)
6	1/30000 (0.02%)
7	0/35000 (0%)
8	1/40000 (0.02%)
9	2/45000 (0.04%)
10	0/50000 (0%)

We will now give some results on unsupervised clustering with single-class SVMs. Unlike in the previous experiment, this time no clustering labels will be available. The SVM will be used to identify the region in the feature space containing all the trajectories in the training sets, and then it will be used to evaluate if anomalous trajectories in the test sets fall outside this region. Because the tuning of the parameter ν, strictly correlated to the number of outliers, will be discussed only in Section IV, here it will be assumed that no outliers are present in the training sets. Data sets have been again randomly generated with increasing numbers of trajectory clusters, ranging from 1 to 10, each one containing 100 trajectories and representing the normal data; each test set also contains ten anomalous trajectories. The results are given in Table II in terms of true positives (correctly detected anomalous trajectories) and false positives (normal trajectories misdetected as anomalies). The number of false positives is stable in all the tests, while the true positives slowly decrease with the increasing number of clusters, because it becomes more and more difficult to detect a potentially anomalous trajectory among many different types of normal ones. In any case, detections results seem promising, especially with few clusters in the scene.

IV. Outlier Detection

The major problem in trajectory clustering with single-class SVMs lies in the choice of the parameter ν. This parameter is

Authorized licensed use limited to: IEEE Xplore. Downloaded on December 12, 2008 at 08:04 from IEEE Xplore. Restrictions apply.

TABLE II
EXPERIMENTAL RESULTS WITH UNSUPERVISED SVM CLUSTERING

# clust.	Classification results			
	True positives		False positives	
1	498/500	(99.6%)	0/5000	(0%)
2	480/500	(96.0%)	0/10000	(0%)
3	467/500	(93.4%)	1/15000	(0.006%)
4	474/500	(94.8%)	0/20000	(0%)
5	456/500	(91.2%)	2/25000	(0.008%)
6	452/500	(90.4%)	3/30000	(0.01%)
7	446/500	(89.2%)	3/35000	(0.008%)
8	443/500	(88.6%)	3/40000	(0.007%)
9	428/500	(85.6%)	4/45000	(0.008%)
10	437/500	(87.4%)	4/50000	(0.008%)

directly linked to the number of outliers in the training data, because it is an upper bound for the fraction of outliers and a lower bound for the fraction of support vectors. For example, if it is known that 10% of the training data will be outliers, this would imply that ν must necessarily be greater than 0.1. However, in practical trajectory clustering applications, there generally is no prior knowledge on the amount of outliers, and thus no bounds can be given.

Because an arbitrary choice of ν would lead to suboptimal results, this parameter should be properly estimated. There is some literature on the estimation of SVM parameters [37], but these works are generally based on cross validation or similar techniques that search the parameter space for a configuration that minimizes the generalization error. This approach can be used in supervised learning problems, where labeled training data are available for validation, but cannot be directly applied to unsupervised, single-class clustering techniques. In this section, we describe an approach to single-class SVM tuning not relying on labeled data; note that labeled data will actually be used for performance evaluation, but they are not involved in the algorithm itself. The proposed method will entirely avoid the problem of estimating a proper value for ν by automatically detecting outliers *before* applying SVM clustering techniques.

In a preliminary work, we already proposed a technique for optimal ν estimation [31], but it is affected by strong computational requirements because, for each classification problem, n SVMs must be trained, with n proportionally increasing with the required precision of the system. In this paper, we propose an alternative and more efficient approach requiring only a single SVM training by avoiding an explicit computation of ν. The main idea is that the outliers are by definition few in number and with an underlying distribution different from the normal data. We thus hypothesize that outliers can be detected by observing the shrink of the hypervolume enclosing all the normal data as we change the training conditions, for example, by varying the value of ν ranging from 0 to 1 (as we did in our previous work) or directly by removing from the training set the support vectors defining the boundaries of the classification function (the approach used in this paper). More precisely, given a support region initially forced to include all the data, its hypervolume in the feature space would shrink rapidly when removing real outliers, while it would shrink slowly when removing normal data, as shown in the 2-D toy example of Fig. 2. The detection of the

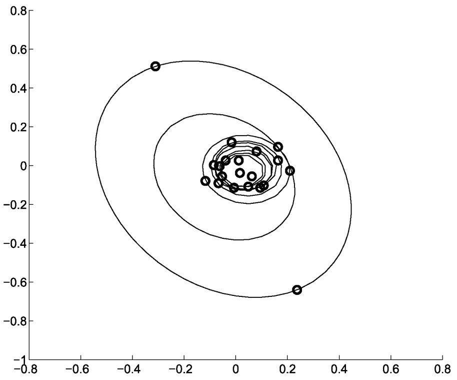

Fig. 2. Support regions found by training a single-class SVM with different values of ν. The largest region is obtained with $\nu = 10^{-7}$; the regions shrink as ν approaches to 1.

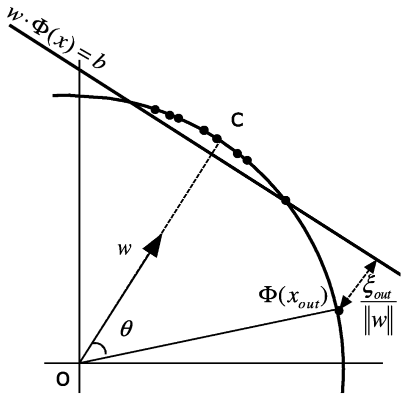

Fig. 3. Training data lie on the surface of an hypersphere in $\mathcal{H}$ when using normalized kernels. The classification hyperplane intersect the hypersphere thus defining an hyperspherical cap containing the majority of the data, while outliers (e.g., the element $\Phi(x_{\text{out}})$) lie outside the cap.

point of change in shrinking speed can thus lead to an optimal support region discarding only real outliers. The support region volume, however, cannot be easily computed in the data space $\mathcal{X}$, and thus we are forced to perform the computations in the feature space $\mathcal{H}$. Recall that $\mathcal{H}$ could be extremely complex or even unknown, but there is an easy way to compute dot products in $\mathcal{H}$ by means of kernels.

From now on, we will suppose to work with a normalized kernel; this is a kernel k such that $k(\mathbf{x},\mathbf{x}) = 1, x \in \mathcal{X}$. Note that the Gaussian kernel chosen in the previous section is always normalized, because

$$k_{\text{Gaussian}}(x,x) = \exp\left(-\frac{\|x-x\|^2}{2\sigma^2}\right) = \exp(0) = 1.$$

Moreover, any other kernel $k'(x_1,x_2) = \Phi(x_1)\cdot\Phi(x_2)$ has a normalized version k defined as

$$\begin{aligned} k(x_1,x_2) &= \frac{\Phi(x_1)}{\|\Phi(x_i)\|}\cdot\frac{\Phi(x_2)}{\|\Phi(x_2)\|} \\ &= \frac{\Phi(x_1)\cdot\Phi(x_2)}{\sqrt{\Phi(x_1)\cdot\Phi(x_1)}\sqrt{\Phi(x_2)\cdot\Phi(x_2)}} \\ &= \frac{k'(x_1,x_2)}{\sqrt{k'(x_1,x_1)}\sqrt{k'(x_2,x_2)}}. \end{aligned}$$

Authorized licensed use limited to: IEEE Xplore. Downloaded on December 12, 2008 at 08:04 from IEEE Xplore. Restrictions apply.

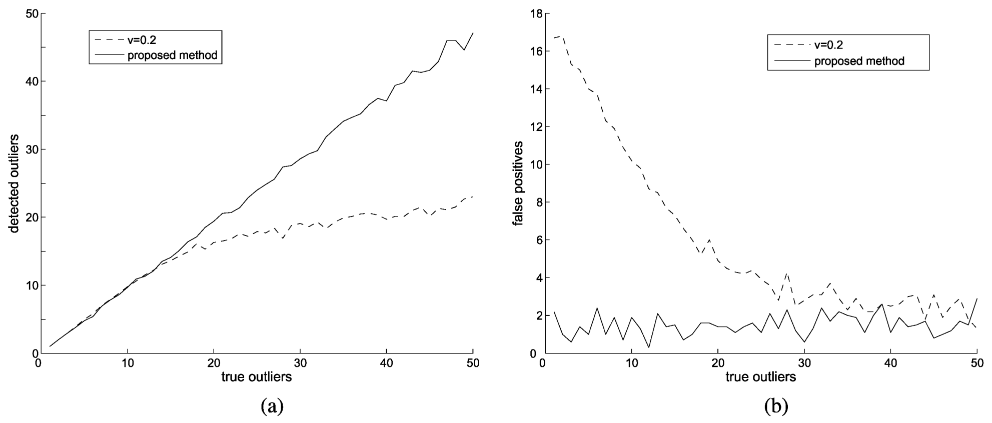

Fig. 4. Comparative results for trajectory clustering with increasing amounts of outliers in the training set. (a) True positives (correctly detected outliers), and (b) false positives (normal trajectories misclassified as outliers) with standard single-class SVM, $\nu = 0.2$ and with the proposed method.

The use of a normalized kernel influences the distribution of the data projected in $\mathcal{H}$, because

$$||\Phi(x)||^2 = \Phi(\mathbf{x}) \cdot \Phi(\mathbf{x}) = k(\mathbf{x}, \mathbf{x}) = 1, \qquad \mathbf{x} \in \mathcal{X}$$

and thus all the feature vectors in $\mathcal{H}$ lie on the surface of an hypersphere with unitary radius, and the decision hyperplane cuts the hypersphere so that all the nonoutlier data lie on the resulting hyperspherical cap (Fig. 3). With normalized kernels it is easy to compute the angle θ between two feature vectors $\Phi(x_1), \Phi(x_2), x_i \in \mathcal{X}$ lying on the hypersphere, because

$$\begin{aligned}\Phi(\mathbf{x}_1) \cdot \Phi(\mathbf{x}_2) &= ||\Phi(\mathbf{x}_1)|| ||\Phi(\mathbf{x}_2)|| \cos(\theta) = \cos(\theta)\\ \theta &= \arccos(\Phi(\mathbf{x}_1) \cdot \Phi(\mathbf{x}_2))\\ &= \arccos(k(\mathbf{x}_1, \mathbf{x}_2)) \qquad (11)\end{aligned}$$

(as a side note, observe that, if $k(x_1, x_2) \geq 0$ such as in the case of the Gaussian kernel, then $\cos(\theta) > 0$ for any $\Phi(x_1)$ and $\Phi(x_2)$, and thus all the data lie in the same orthant).

Let us now consider the center c of the hyperspherical cap containing the nonoutlier data. Because vector $\mathbf{w}$ is perpendicular to the decision hyperplane [see (1)], the center c is defined as

$$\mathbf{c} = \frac{\mathbf{w}}{||\mathbf{w}||}.$$

Using (11), it is possible to compute the angle between c and a generic outlier $\mathbf{x}_{\text{out}}$ (see Fig. 3), defined as

$$\begin{aligned}\theta_{\text{out}} &= \arccos(c \cdot \Phi(\mathbf{x}_{\text{out}}))\\ &= \arccos\left(\frac{\mathbf{w} \cdot \Phi(\mathbf{x}_{\text{out}})}{||\mathbf{w}||}\right).\end{aligned}$$

Using (9), this leads to

$$\theta_{\text{out}} = \arccos\left(\frac{b - \xi_{\text{out}}}{||\mathbf{w}||}\right).$$

Using (8) and (9) and knowing the values α_i, obtained as a solution of (7), θ_{out} can thus be defined as

$$\theta_{\text{out}} = \arccos\left(\frac{\sum_i \alpha_i k(\mathbf{x}_{\text{out}}, \mathbf{x_i})}{\sqrt{\sum_i \sum_j \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j)}}\right). \qquad (12)$$

We now observe that, in the degenerate case of $\nu = 1$, the constraints in (7) lead to the solution $\alpha_i = 1/n$, and thus all the data are considered outliers and the decision function (10) reduces to a Parzen window estimate of the underlying distribution. In this case, from (8), it follows that $\mathbf{w}/||\mathbf{w}||$ points to the mean of the feature vectors in $\mathcal{H}$; this can be intuitively seen in Fig. 2, where the support region converges toward the data center as ν approaches 1.

The proposed outlier detection technique can now be defined. Given a set of training data, a single-class SVM is ideally trained using $\nu = 1$ (we do not really need to compute the solution, because we already know it will be $\alpha_i = 1/n$); then, for each training vector $\mathbf{x}_i$, the angle θ_i between $\Phi(\mathbf{x}_i)$ and the data center c is calculated using (12). Note that (12) is defined only for outliers, but in this case, it can be applied to any data vector, because any vector is by definition an outlier when $\nu = 1$. The considerations made at the beginning of this section on support region hypervolumes can now be applied to the values of θ: we can assume that the majority of the data will have similar, small values of θ because, if the classification works correctly, they are enclosed in the same hyperspherical cap of the nonoutlier data, while few elements will have large values of θ, being outliers falling out of the searched distribution. We identify these elements by plotting the θ values sorted in ascending order (see Fig. 5): the plot will have a high-curvature "elbow" separating the normal data, characterized by a slow growth in the values of θ, from the outliers, characterized by a final quick growth of the plot. The elbow can be identified by searching for the point of the plot with the largest distance from a line connecting the two extrema of the plot itself. The elbow point has an associated θ_{th} angle (see again Fig. 5) that can be used as a threshold in order to separate normal data from anomalies: outliers in the

Authorized licensed use limited to: IEEE Xplore. Downloaded on December 12, 2008 at 08:04 from IEEE Xplore. Restrictions apply.

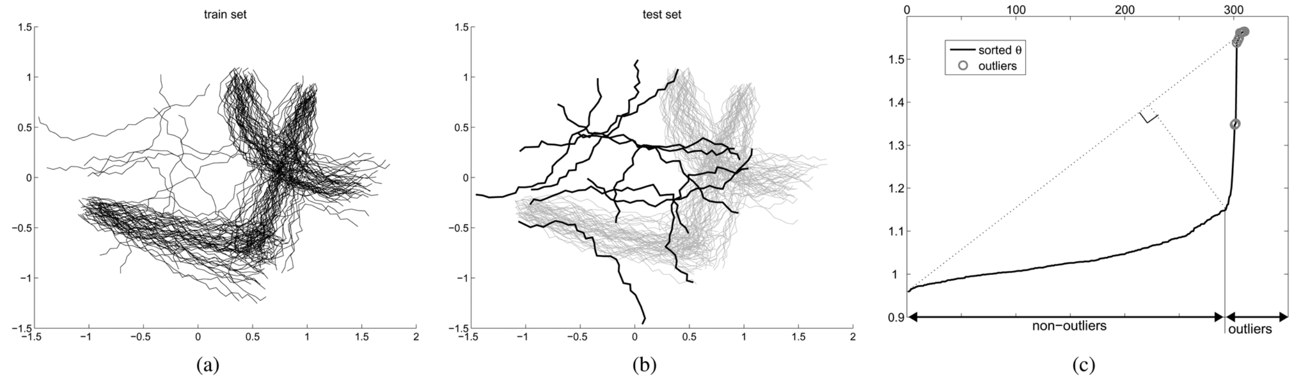

Fig. 5. Anomalous trajectory detection. (a) Training set: 300 normal trajectories are grouped in three main clusters and are perturbed with ten outliers. (b) Test set: with ten utliers correctly detected. (c) Sorted values of θ.

training set can be detected by finding the elements x_i such that $\theta_i > \theta_{th}$.

At this point there is no need to explicitly find a good ν value for outlier removal, because we have already detected which data should not be considered during the SVM training step. A single-class SVM can thus be trained with a near-to-zero value for ν on the training data set where the supposed outliers are removed; the small ν will enforce the decision function to enclose all the data previously classified as nonoutliers. The final classification function will not be influenced by the presence of outliers, because they have been explicitly detected and removed from the training set.

The process can thus be summarized as follows. First, compute the angle θ for each element in the training data set. Then, sort the obtained values and remove the outliers automatically detected by searching for the elbow point in the plot of θ. Finally, train an hard-margin single-class SVM on the remaining data. A single SVM training is needed, thus greatly reducing the computational complexity if compared to our preliminary work [31]. The overall computational time depends on the time required for the training of the SVM, which is $O(m^3)$, where m is the number of elements in the training set, if no particular optimizations are used and can be improved up to $O(m)$ with proper approximation techniques [38]. Computing the θ values can be done in $O(m^2)$, while sorting them and finding the bend points is $O(m \log m)$. Once the training step is completed, checking a new trajectory involves a standard SVM test, which is a very fast operation because it is linear in the number n_{sv} of support vectors, and typically $n_{\text{sv}} \ll m$.

V. Experimental Results

To test the validity of the proposed approach, experimental results on both synthetic and real-world data sets are here given.[1] Before focusing on automatic outlier discovery, we want to check if the angle θ, as defined in (12), is a good measure for outlier detection. To perform this test, we compared the proposed method with another simple yet very effective outlier detection technique, based on the concept of *discords* [39], [40]: a discord is defined as the trajectory maximizing its Euclidean distance from the nearest neighbor in the data space. The discord distance d has the same interpretation of θ in our work: the higher the value, the more probable it is for a trajectory to be an outlier. We randomly generated 1000 data sets, each one containing 260 trajectories: 250 trajectories grouped in five main clusters, plus ten outliers. For each data set, we computed both θ and d, and we checked if the outliers really fall within the ten trajectories with highest θ or d, respectively. Experimental results show that the performances of the two approaches are comparable, with a 3.70% error obtained with the proposed method and 2.96% error using discords. The use of discords has its main advantages in computational efficiency (in [39], a linear algorithm for discord discovery is proposed) and in the lack of tuning parameters, while the results of our method proved to be dependant on a good choice of σ ($\sigma = 0.5$ has been used in the previous test). On the other hand, our method has the advantage of generalization, because the analysis of previously unseen data drawn from the same probability distribution of the training data can be done without training again the system. Moreover, the major advantage of the proposed method relies in its ability of automatically detecting the number of outliers, while discord-based methods as proposed in [39] can only detect the k most anomalous trajectories, without giving any hint on a good choice of k.

1) Synthetic Data: In the rest of this section, we will test the automatic outlier detection capabilities of the proposed approach. To compare the proposed approach with a standard single-class SVM with fixed ν training, 50 experimental cases were considered by randomly generating training sets with 100 normal trajectories grouped in two clusters, in which anomalous trajectories were added, ranging from 1 to 50 anomalies; results have been averaged over ten runs for each test case. For each training set, two SVM were trained, one using the proposed method and one with $\nu = 0.2$. Both SVMs use a Gaussian kernel with $\sigma = 2$. Results are shown in Fig. 4 and are measured in terms of anomalies and normal data correctly identified in the training sets. Note that no test sets are used here, because we are not measuring the generalization capabilities of the proposed method, but only its ability to remove outliers from the training data sets. As it can be seen, the fixed ν approach initially has a large number of false positives (normal trajectories classified as anomalous) because $\nu = 0.2$

[1]The data sets used in this section are available at http://avires.dimi.uniud.it/papers/trclust

Authorized licensed use limited to: IEEE Xplore. Downloaded on December 12, 2008 at 08:04 from IEEE Xplore. Restrictions apply.

TABLE III
FALSE POSITIVES RESULTS FOR THE PROPOSED METHOD, EXPRESSED IN TERMS OF NORMAL TRAJECTORIES MISCLASSIFIED AS ANOMALIES OVER THE TOTAL AMOUNT OF NORMAL TRAJECTORIES IN EACH DATA SET. RESULTS HAVE BEEN AVERAGED OVER TEN RUNS

Anomalies	Clusters in the training set									
	1	2	3	4	5	6	7	8	9	10
1	3.10%	2.20%	2.20%	2.60%	1.78%	1.41%	1.42%	1.35%	1.03%	1.47%
2	1.70%	2.05%	1.53%	2.02%	1.80%	1.31%	1.25%	0.90%	1.24%	0.87%
3	2.70%	2.70%	2.10%	2.17%	1.72%	1.46%	1.40%	1.08%	0.82%	0.72%
4	2.30%	2.00%	2.26%	1.45%	1.32%	1.36%	1.40%	0.90%	1.05%	0.91%
5	3.30%	2.15%	2.26%	1.60%	1.82%	1.08%	1.12%	0.92%	1.33%	0.96%
6	3.30%	2.60%	2.73%	1.72%	1.36%	1.65%	1.47%	1.22%	0.83%	0.68%
7	1.90%	2.15%	1.40%	1.10%	1.52%	1.16%	1.62%	1.10%	0.95%	0.65%
8	1.10%	2.90%	1.76%	1.92%	1.42%	1.51%	1.32%	0.97%	1.08%	0.95%
9	2.70%	2.95%	1.50%	1.92%	1.98%	1.31%	1.48%	1.02%	0.94%	1.27%
10	2.70%	2.70%	1.46%	1.85%	1.32%	1.65%	1.57%	0.96%	0.96%	0.93%

TABLE IV
TRUE POSITIVES RESULTS FOR THE PROPOSED METHOD, EXPRESSED IN TERMS OF ANOMALOUS TRAJECTORIES CORRECTLY CLASSIFIED AS OUTLIERS OVER THE TOTAL AMOUNT OF ANOMALOUS TRAJECTORIES IN EACH DATA SET. RESULTS HAVE BEEN AVERAGED OVER TEN RUNS

Anomalies	Clusters in the training set									
	1	2	3	4	5	6	7	8	9	10
1	90.00%	100.00%	100.00%	100.00%	80.00%	100.00%	90.00%	70.00%	100.00%	80.00%
2	100.00%	100.00%	100.00%	95.00%	100.00%	100.00%	95.00%	100.00%	100.00%	85.00%
3	100.00%	100.00%	100.00%	100.00%	100.00%	96.66%	96.66%	96.66%	93.33%	93.33%
4	100.00%	97.50%	100.00%	97.50%	92.50%	95.00%	97.50%	85.00%	95.00%	90.00%
5	100.00%	98.00%	98.00%	100.00%	98.00%	88.00%	96.00%	92.00%	80.00%	94.00%
6	100.00%	95.00%	98.33%	100.00%	95.00%	93.33%	91.66%	90.00%	91.66%	86.66%
7	100.00%	100.00%	100.00%	98.57%	92.85%	92.85%	94.28%	91.42%	85.71%	84.28%
8	100.00%	97.50%	96.25%	98.75%	98.75%	95.00%	97.50%	97.50%	95.00%	91.25%
9	98.88%	98.88%	98.88%	97.77%	95.55%	97.77%	95.55%	96.66%	92.22%	87.77%
10	100.00%	100.00%	98.00%	98.00%	96.00%	97.00%	92.00%	96.00%	92.00%	86.00%

is an overestimated choice for a small amount of outliers. If we increase the number of real outliers, we face the opposite problem, with low true positives rates. On the other hand, the proposed approach always has a good true positives detection rate (more than 90%) with a negligible false positive rate.

Previous results show how the proposed method can handle outliers in the training data better than the naive approach, however they do not prove that the system has good generalization performances when classifying previously unseen patterns. We thus considered 100 different experimental cases, each one with a different number of groups of 100 normal trajectories (ranging from 1 to 10) and outliers (from 1 to 10). For each test, ten different training/test sets were created, for a total of 2000 data sets. Single-class SVMs have been trained using the proposed method on the training sets and the results were measured on the test sets (for an example, see Fig. 5). Note that outliers in the test sets are drawn from the same distribution used for outliers in the corresponding training sets, in order to ease the detection of anomalous training patterns being included in the normality class. Tables III and IV, respectively, show the generalization results of the proposed approach in terms of false positives and true positives. As it can be seen, the true positives detection always gives good results, with an average of 92% of correct detections. Both true positives and false positives results do not seem to be noticeably influenced by the increasing number of outliers, while there is a slight performance degradation in true positive detection when increasing the number of groups of normal trajectories.

Because the last experiment did not show any appreciable variation in the results with low numbers of outliers (from 1 to 10 outliers in each training/test set), another test has been performed to measure the robustness of the proposed method in presence of large amounts of outliers. We considered the case of three clusters of normal data, each one containing 100 trajectories, where an increasing number of outliers has been added, ranging from 1 to 300 (in the last case, the data sets thus contain 50% outliers). Fig. 6 shows the plot of the true positives detected by the proposed approach; as can be seen, the system has a high true positives detection rate even in presence of large amounts of outliers. The maximum number of detected false positives was 2.

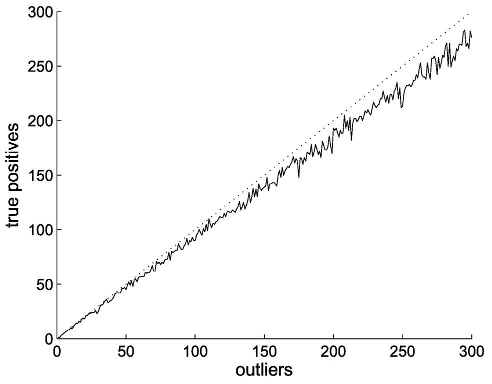

Fig. 6. True positives results with high amounts of outliers both in the training and in the test set and with 300 normal trajectories grouped in three clusters.

2) Real-World Data: The previously described experimental results were obtained on synthetic data to ease the evaluation of the algorithm performances. We now conclude this section

Authorized licensed use limited to: IEEE Xplore. Downloaded on December 12, 2008 at 08:04 from IEEE Xplore. Restrictions apply.

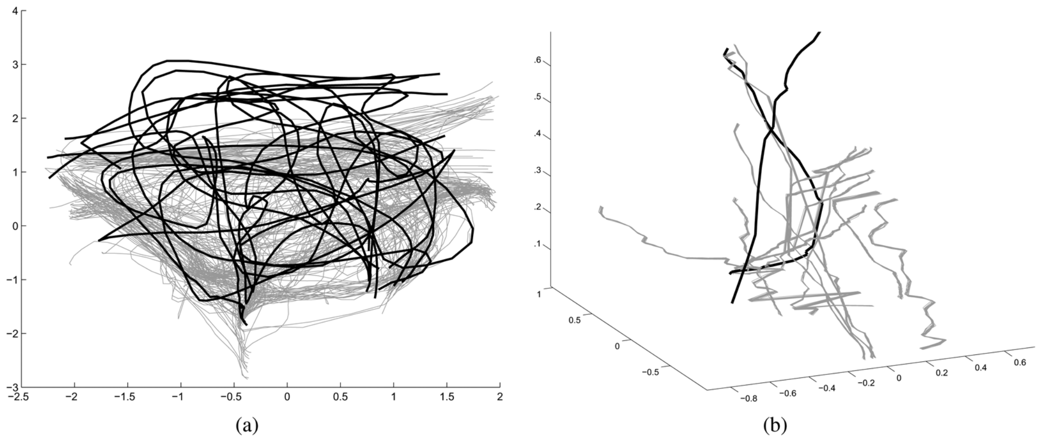

Fig. 7. Test results on publicly available data sets. The black thick trajectories are outliers. (a) Data set from [41]. (b) Data set from [42].

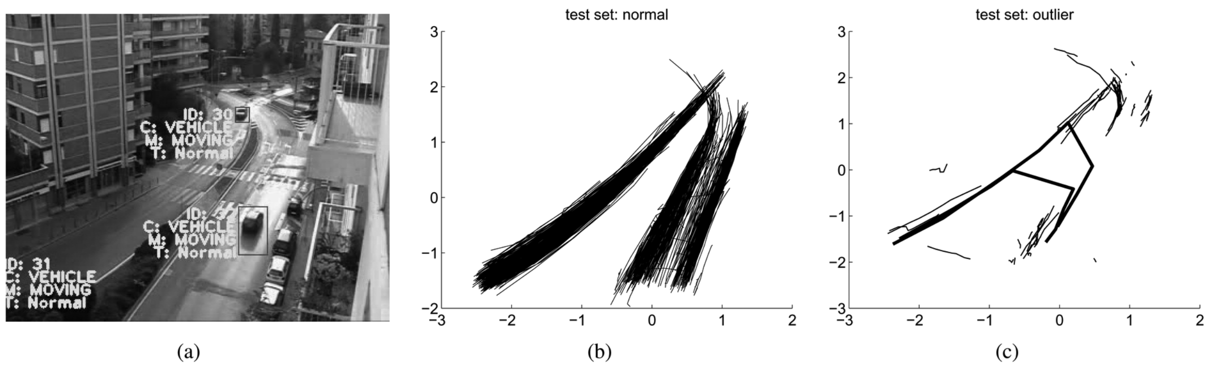

Fig. 8. Proposed method applied to real-world data. (a) Top view of an urban road. (b) Normal trajectories. (c) Anomalous trajectories detected in the test set (bold lines: the U-turn trajectories added to the test set).

with some examples of applications to real-world data, even though in this case there is no easy way to collect ground truth information and thus it is not possible to give objective performance measurements. We considered the case of an urban road, monitored by a static color camera [Fig. 8(a)]. The acquired video data consist in a 51-min-long sequence at 320×240 resolution, during which 1430 vehicles were detected by means of multi-Gaussian background-subtraction techniques as in [19] and tracked using a combination of Kalman-based and CamShift trackers. We simulated the presence of two forbidden U-turns obtained by merging the trajectories of vehicles moving in opposite directions. Moreover, the trackers are not error-free, and thus the acquired data are affected by noise, especially in the form of broken trajectories due to lost objects. The acquired trajectories have been split in a train and a test set, each one containing 715 trajectories, and the proposed method has been applied using $\sigma = 3$. Because the trajectory shapes are quite simple, we have chosen to subsample each trajectory using eight points, thus leading to feature vectors composed of 16 elements (the x and y coordinates for each subsampled point). The results on the test set are shown in Fig. 8(b) and (c). As can be seen, the system has correctly recognized as anomalies the two U-turn trajectories (bold lines); moreover, the broken trajectories generated by errors in the tracking system are marked as anomalies as well, because they significantly differ in shape from the normal data.

To test the system on other publicly available data sets, we also applied the proposed outlier detection algorithm to the trajectory data sets used in [41] and [42]. The data set from [41] consists in 160 trajectories of moving persons acquired in an indoor environment. The 152 trajectories are grouped in four main clusters, and the remaining eight are outliers. Fig. 7(a) shows the results obtained by our system: all the eight anomalous trajectories have been correctly detected, although also two false positives are present. The data set used in [42] is made up of trajectories extracted from infrared surveillance videos. Fig. 7(b) shows how the 237 normal trajectories and the two outliers are all correctly classified. To compare results under the same conditions of [42], time has also been included in the feature vector describing each trajectory.

VI. Conclusion

In this paper, we proposed a technique for anomalous event detection by means of trajectory analysis. The trajectories are subsampled to a fixed-dimension vector representation and clustered with a single-class SVM. The presence of outliers in the training data has led us to a novel technique for automatic detection of the outliers based on geometric considerations in the SVM feature space. The experimental results have proven the validity of the proposed approach.

Authorized licensed use limited to: IEEE Xplore. Downloaded on December 12, 2008 at 08:04 from IEEE Xplore. Restrictions apply.

References

[1] H. Buxton, "Learning and understanding dynamic scene activity: A review," *Image Vis. Comput.*, vol. 21, pp. 125–136, 2003.

[2] W. Hu, T. Tan, L. Wang, and S. Maybank, "A survey on visual surveillance of object motion and behaviors," *IEEE Trans. Syst. Man Cybern. C, Appl. Rev.*, vol. 34, no. 3, pp. 334–352, Aug. 2004.

[3] Y. Ivanov and A. Bobick, "Recognition of visual activities and interactions by stochastic parsing," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 852–872, Aug. 2000.

[4] D. Minnen, I. Essa, and T. Starner, "Expectation grammars: Leveraging high-level expectations for activity recognition," in *Comput. Vis. Pattern Recognit.*, 2003, pp. II-626–II-632.

[5] D. Moore and I. Essa, "Recognizing multitasked activities from video using stochastic context-free grammar," in *Proc. 18th Nat. Conf. Artif. Intell.*, 2002, pp. 770–776.

[6] V. Vu, F. Brémond, and M. Thonnat, "Automatic video interpretation: A novel algorithm for temporal scenario recognition," in *Proc. 8th Int. Joint Conf. Artif. Intell.*, Acapulco, Mexico, 2003, pp. 9–15.

[7] D. Ayers and M. Shah, "Monitoring human behavior from video taken in an office environment," *Image Vis. Comput.*, vol. 19, no. 2, pp. 833–846, 2001.

[8] G. Medioni, I. Cohen, F. Brémond, S. Hongeng, and R. Nevatia, "Event detection and analysis from video streams," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 8, pp. 873–889, Aug. 2001.

[9] T. Wada and T. Matsuyama, "Multiobject behavior recognition by event driven selective attention method," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 873–887, Aug. 2000.

[10] N. Oliver, B. Rosario, and A. Pentland, "A bayesian computer vision system for modeling human interactions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 831–843, Aug. 2000.

[11] A. Galata, N. Johnson, and D. Hogg, "Learning variable length Markov models of behavior," *Comput. Vis. Image Understand.*, vol. 81, no. 3, pp. 398–413, 2001.

[12] M. Brand and V. Kettnaker, "Discovery and segmentation of activities in video," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 844–851, Aug. 2000.

[13] F. Bashir, A. Khokhar, and D. Schonfeld, "Object trajectory-based activity classification and recognition using hidden Markov models," *IEEE Trans. Image Process.*, vol. 16, no. 7, pp. 1912–1919, Jul. 2007.

[14] S. Hongeng, R. Nevatia, and F. Brémond, "Video based event recognition: Activity representation and probabilistic methods," *Comput. Vis. Image Understand.*, vol. 96, pp. 129–162, 2004.

[15] N. Moenne-Loccoz, F. Brémond, and M. Thonnat, "Recurrent Bayesian network for the recognition of human behaviors from video," in *Proc. Int. Conf. O Comput. Vis. Syst.*, Graz, Austria, 2003, pp. 68–78.

[16] T. W. Liao, "Clustering of time series data—A survey," *Pattern Recognit.*, vol. 38, no. 11, pp. 1857–1874, 2005.

[17] N. Johnson and D. Hogg, "Learning the distribution of object trajectories for event recognition," *Image Vis. Comput.*, vol. 14, no. 8, pp. 609–615, 1996.

[18] N. Johnson and D. Hogg, "Representation and synthesis of behavior using Gaussian mixtures," *Image Vis. Comput.*, vol. 20, pp. 889–894, 2002.

[19] C. Stauffer and W. Grimson, "Learning patterns of activity using real-time tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 852–872, Aug. 2000.

[20] D. Makris and T. Ellis, "Path detection in video surveillance," *Image Vis. Comput.*, vol. 20, no. 12, pp. 895–903, 2002.

[21] D. Makris and T. Ellis, "Learning semantic scene models from observing activity in visual surveillance," *IEEE Trans. Syst. Man Cybern. B, Cybern.*, vol. 35, no. 3, pp. 397–408, Jun. 2005.

[22] C. Piciarelli and G. Foresti, "On-line trajectory clustering for anomalous events detection," *Pattern Recognit. Lett.*, vol. 27, pp. 1835–1842, 2006.

[23] F. Porikli, "Trajectory distance metric using hidden Markov model based representation," in *Proc. 8th Eur. Conf. Comput. Vis., PETS Workshop*, Prague, Czech Republic, 2004, pp. 9–16.

[24] W. Hu, X. Xiao, Z. Fu, D. Xie, T. Tan, and S. Maybank, "A system for learning statistical motion patterns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 9, pp. 1450–1464, Sep. 2006.

[25] J. Lou, Q. Liu, T. Tan, and W. Hu, "Semantic interpretation of object activities in a surveillance system," in *Proc. Int. Conf. Pattern Recognit,*, Quebec City, QC, Canada, 2002, pp. III:777–780.

[26] R. Morris and D. Hogg, "Statistical models of object interaction," *Int. J. Comput. Vis.*, vol. 37, no. 2, pp. 209–215, 2000.

[27] X. Wang, K. Tieu, and E. Grimson, "Learning semantic scene models by trajectory analysis," in *Proc. Eur. Conf. Comput. Vis.*, Graz, Austria, 2006, pp. 110–123.

[28] H. M. Dee and D. C. Hogg, "On the feasibility of using a cognitive model to filter surveillance data," in *Proc. IEEE Int. Conf. Adv. Video Signal Based Surveillance Syst.*, Como, Italy, 2005, pp. 34–39.

[29] B. Schölkopf, J. Platt, J. Shave-Taylor, A. Smola, and R. Williamson, "Estimating the support of a high-dimensional distribution," Microsoft Research, Redmon, WA, Tech. Rep. TR 87, 1999.

[30] D. Lee and J. Lee, "Trajectory-based support vector multicategory classifier," in *Proc. Int. Symp. Neural Netw.*, Chongquing, China, 2005, pp. 857–862.

[31] C. Piciarelli and G. Foresti, "Anomalous trajectory detection using support vector machines," in *Proc. IEEE Int. Conf. Adv. Video Signal-Based Surveillance*, London, U.K., 2007, pp. 153–158.

[32] V. Vapnik and A. Lerner, "Pattern recognition using generalized portrait method," *Autom. Remote Control*, vol. 24, pp. 774–780, 1963.

[33] B. Boser, I. Guyon, and V. Vapnik, "A training algorithm for optimal margin classifiers," in *Proc. 5th Annu. ACM Workshop Comput. Learn. Theory*, 1992, pp. 144–152.

[34] B. Schölkopf and A. Smola*, Learning With Kernels—Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA: MIT Press, 2002.

[35] K. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf, "An introduction to kernel-based learning algorithms," *IEEE Trans. Neural Netw.*, vol. 12, no. 2, pp. 181–201, Mar. 2001.

[36] J. Shave-Taylor and N. Cristianini*, Kernel Methods for Pattern Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[37] O. B. O. Chapelle, V. Vapnik, and S. Maukherjee, "Choosing multiple parameters for support vector machines," *Mach. Learn.*, vol. 46, no. 1–3, pp. 131–159, 2002.

[38] I. Tsang, J. Kwok, and P. Cheung, "Core vector machines: Fast SVM training on very large data sets," *J. Mach. Learn. Res.*, vol. 6, pp. 363–392, 2005.

[39] D. Yankov, E. Keogh, and U. Rebbapragada, "Disk aware discord discovery: Finding unusual time series in terabyte sized datasets," in *Proc. IEEE Int. Conf. Data Mining*, Omaha, NE, 2007, pp. 381–390.

[40] E. Keogh, J. Lin, and A. Fu, "Hot SAX: Efficiently finding the most unusual time series subsequence," in *Proc. IEEE Int. Conf. Data Mining*, Houston, TX, 2005, pp. 226–233.

[41] A. Naftel and S. Khalid, "Classifying spatiotemporal object trajectories using unsupervised learning in the coefficient feature space," *Multimedia Syst.*, vol. 12, no. 3, pp. 227–238, 2006.

[42] D. Pokrajac, A. Lazarevic, and L. Latecki, "Incremental local outlier detection for data streams," in *Proc. IEEE Symp. Comput. Intell. Data Mining*, Honolulu, HI, 2007, pp. 504–515.

Claudio Piciarelli (S'07–M'08) received the M.Sc. and Ph.D. degrees in computer science from University of Udine, Udine, Italy, in 2003 and 2008, respectively.

Since 1999 he has been working on several national and European projects in collaboration with the Artificial Vision and Real-time Lab, University of Udine. He coauthored more than 20 works published in international journals and conferences and has been a reviewer for several IEEE conferences and journals. His main research interests include computer vision, artificial intelligence, pattern recognition, and machine learning. He is currently mainly involved in the use of active vision in surveillance systems and in behavior analysis for automatic detection of anomalous events.

Authorized licensed use limited to: IEEE Xplore. Downloaded on December 12, 2008 at 08:04 from IEEE Xplore. Restrictions apply.

Christian Micheloni (S'02–M'06) received the Laurea degree *(cum laude)* and the Ph.D. degree in computer science from the University of Udine, Udine, Italy, in 2002 and 2006, respectively.

Currently, he is an Assistant Professor at the University of Udine. Since 2000, he has taken part in European research being under contract for several European Projects. He has coauthored more than 40 scientific works published in international journals and refereed international conferences. He serves as a reviewer for several international journals and conferences. His main interests involve active vision for understanding of the scene by images acquired by moving cameras and neural networks for the classification and recognition of the objects moving within the scene. He is also interested in pattern recognition techniques for both the automatic tuning of the camera parameters for an improved image acquisition and for the detection of the faces. All these techniques are mainly developed and applied for video surveillance purposes.

Dr. Micheloni is member of the International Association of Pattern Recognition (IAPR).

Gian Luca Foresti (S'93–M'95–SM'01) was born in Savona, Italy, in 1965. He received the laurea degree *(cum laude)* in electronic engineering and the Ph.D. in computer science from University of Genoa, Genoa, Italy, in 1990 and 1994, respectively.

In 1994, he was a visiting Professor at University of Trento, Italy. Since 2000, he has been Professor of Computer Science at the Department of Mathematics and Computer Science (DIMI), University of Udine, Udine, Italy, where he is also Director of the Artificial Vision and Real-Time Systems (AVIRES) Lab. His main interests involve computer vision and image processing, multisensor data and information fusion, and pattern recognition and neural networks. His proposed techniques found applications in the following fields: video-based surveillance systems, autonomous vehicle driving, road traffic control, human behavior understanding, and visual inspection. He is author or coauthor of more than 200 papers published in international journals and refereed international conferences and he has contributed in several books in his area of interest.

Dr. Foresti was General Chairman and member of Program Committees at several conferences where he has been coorganizer of several special sessions on computer vision, data fusion, and pattern recognition. He has been Finance Chair of the 2005 IEEE Conference on Image Processing, Genoa, Italy. He has been Guest Editor of a Special Issue of the IEEE PROCEEDINGS on "Video Communications, Processing and Understanding for Third Generation Surveillance Systems" and of a Special Issue of the IEEE TRANSACTIONS ON SYSTEMS, MAN AND CYBERNETICS on "Ambient Intelligence." He serves as a reviewer for several international journals, and for the European Union in different research programs. He is the Italian delegate of the NATO-RTO Information System Technology (IST) panel and member of the International Association of Pattern Recognition (IAPR).

Authorized licensed use limited to: IEEE Xplore. Downloaded on December 12, 2008 at 08:04 from IEEE Xplore. Restrictions apply.