Proceedings of the 12th International IEEE Conference on Intelligent Transportation Systems, St. Louis, MO, USA, October 3-7, 2009

Trajectory Learning and Analysis Based on Kernel Density Estimation

Jianying Zhou, Kunfeng Wang, Shuming Tang, *Member, IEEE,* Fei-Yue Wang, *Fellow, IEEE*

Abstract— **This paper presents a novel kernel density estimation approach to vehicle trajectory learning and motion analysis. The framework comprises a training stage and a testing stage. In the training stage, vehicle trajectories are first clustered by the hierarchical spectral clustering method. Then, through the proposed kernel density estimation approach, the average kernel density of one point on a trajectory can be estimated. In the testing stage, the compactness estimated by a Gaussian kernel function is introduced. Abnormal trajectories are detected with compactness lower than expected for a few consecutive frames. Vehicle motions are identified into multiple activities with their respective trajectory compactness.**

I. INTRODUCTION

A KEY GOAL of video surveillance is to understand the interactions and behaviors present in a scene. It is important to detect abnormal activities or behaviors for surveillance system that need continually monitor a site. Unfortunately, large amounts of video data are generated, making it a tedious and tiring job for human to manually process accurately and quickly. Computer vision technology helps to automate the process. Using computer vision techniques, in activity analysis systems, normal activity patterns in the video are filtered out automatically, and attention is focused on the data in which the activity is abnormal.

By observing and collecting tracking data, which is required in the activity analysis systems, normal motion patterns corresponding to lanes in the road can be learned off-line or on-line. Then the normal motion patterns can be used to detect anomalous actions if the activity happens with a low probability to match any normal activity.

A training set of trajectories is acquired by collecting tracking data for a period of time. Then the set is clustered to find the major spatial routes, which can be probabilistically modeled by several statistical models, such as hidden Markov model (HMM), Gaussian mixture models and so on. Over the past few years, based on trajectory clustering, a lot of work has been devoted to understanding activity patterns of vehicles [1], [2], [3], [4], [5], [6]. On the other hand, statistical models can be used to complement object tracking or shadow detection [7], [8].

Wang et al. [9] used spectral clustering method to build semantic regions for two scenes through trajectory clustering. They first used the size of objects and spatial information to cluster trajectories into vehicles and pedestrians. Then spatial and temporal information were used to cluster trajectories into more clusters. Wang et al. [10], [11] also used LDA (latent Dirichlet allocation) and HDP (hierarchical Dirichlet process) models to do activity analysis separately. LDA and HDP are two methods based on Bayesian models; they have been used very well for word-document analysis by supposing that words often co-existing in the same documents are clustered into the same topic. Supposing documents were trajectories and words were observations and topics were semantic regions, Wang et al used their improved HDP and LDA individually to analyze the traffic scenes.

Hu et al. [3] used FCM (fuzzy c-means) to analyze traffic scenes and modeled the probability density as Gaussian function to make prediction of the activity.

After clustering the trajectories, Morris et al. [5] and Bashir et al. [12] used HMM to describe the transitions between states and obtained good experimental results. They supposed that each activity path can be represented by several states in HMM, after learning the transition matrix by standard methods, such as the Expectation Maximum (EM) algorithm, the probability of moving from one state to another could be predicted. Instead of finding the transition probabilities, Saleemi et al. [7] used the learned density distribution, which is obtained in the training stage, as a generative model and sample feature points from the model to construct a sequence of tracks in the testing stage. In their work, kernel density estimation (KDE) was used to describe the transition distribution, and then the Metropolis Hastings Sampling was used to sample the trajectories to help object tracking or activity understanding. In [13], 4-D histograms were built to describe the scene. After learning the histogram of each class using their proposed kernel which was similar to KDE, the obtained statistical descriptions of motion patterns can be used to detect and classify the trajectory in the testing stage.

As depicted in [14], the work in the training stage can be summarized in Fig. 1. In the tracking stage, vehicles are detected and tracked between consecutive frames, and at the same time the spatial and velocity attributes for each frame of the vehicle are extracted. Before cluster, the trajectories

Manuscript received May 15, 2009. This work was supported in part by the MOST Grants 2006CB705500 and 2007AA11Z217, NNSFC Grants 90820000 and 60621001, CAS Grant 2F07N04, and Shandong Province Taishan Chair Professor Fund 011006005.

Jianying Zhou and Kunfeng Wang are with the Key Laboratory of Complex Systems and Intelligence Science, Institute of Automation, Chinese Academy of Science, Beijing 100190, China. (e-mail:zhoujianyingbest@gmail.com, kungfeng.wang@ia.ac.cn)

Shuming Tang is with the Shandong University of Science and Technology, Qingdao 266510, China; and the Key Laboratory of Complex Systems and Intelligence Science, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China (e-mail: sharron@ieee.org).

Fei-Yue Wang is with the Key Laboratory of Complex Systems and Intelligence Science, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China, and also with the University of Arizona, Tucson, AZ 85721 USA (e-mail: feiyue@sie.arizona.edu).

978-1-4244-5521-8/09/$26.00 ©2009 IEEE

should be preprocessed to remove noises or may be normalized with the same length. Based on the spatial and velocity attributes, the similarity measure between trajectories will be built by Euclidean distance or Hausdorff distance etc. Trajectory clustering could be done by unsupervised learning algorithms, such as k-means clustering, SOM, and spectral clustering. After trajectory clustering, semantic regions of each activity pattern are depicted by statistical distributions. After obtaining the statistical distributions, we can use the distributions to detect and understand vehicle motion in the testing stage or turn back to improve low-level tracking or detection as in [7], [8].

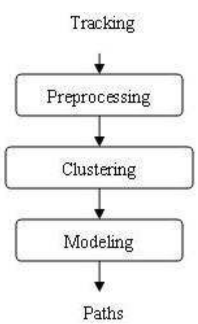


Fig. 1. Trajectory learning steps [14].

The rest of this paper is organized as follows. In section II, some theoretical background is introduced, and the details of spectral clustering algorithm we used are first given. Using the hierarchical spectral clustering method, the trajectories are clustered into five clusters. Section III describes our proposed method to build statistical description of each cluster and introduce a factor of compactness for trajectory classification and abnormal detection in the testing stage. In Section IV, some experimental results are shown and analyzed. Finally, in Section V, some conclusions and discussions bring the paper to completion.

II. THEORETICAL BACKGROUND AND TRAJECTORY CLUSTERING

A. Object Detection and Tracking

A lot of work of object detection in traffic surveillance has been done in the lab I am working for [15]. Gaussian mixture models of background modeling method [16], is used in our work, in each frame after background subtraction, objects are detected and in two frames matched according to their position and velocity attributes. Then vehicle detection and tracking have been completed. The trajectory of each observed object is a sequence of tracking states obtained in every frame, $A = \{\vec{a}_i\}$, where $\vec{a}_i$ can depict things like position, velocity, appearance, shape, or other object attributes. In our work we select the attributes with $\vec{a}_i =< x_i^a, y_i^a, u_i^a, v_i^a >$, where (x_i^a, y_i^a) are the spatial position of the ith observation, and (u_i^a, v_i^a) are the velocity attribute in x and y direction separately of the image. This trajectory information forms the basic block for further activity analysis.

Although tracking is well studied, there are still many difficulties due to various effects, such as occlusion, camera twittering and real-time adaptability to changing conditions. These cause errors in the form of noise measurements or trajectories breaking, which will be accounted for in the activity learning process.

B. Spectral Clustering Algorithm and Trajectories Clustering

Many kinds of clustering algorithm need the trajectories to be set with the same length. This will introduce noises in the clustering. Spectral clustering method avoid these problems.

In the past few years, spectral clustering method has become one of the most popular modern clustering algorithms. Spectral clustering is simple to implement and very often outperforms traditional clustering algorithms such as k-means [17]. In spectral clustering, we just need to get the similarity or dissimilarity matrix between different observations. Hence the length of every observation is not necessary to be equal differing from that of k-means or SOM. In spectral clustering, similarity graph matrix W between each two observations is first built, from which the Laplacian matrix L can be derived, and then the first K eigenvectors of L can be found. After that let the eigenvectors $v_1, \ldots, v_K$ be matrix V's columns and $\{y_i, i = 1, \ldots, n\}$, be the rows of matrix V. Subsequently, k-means clustering algorithm is used to cluster the rows $\{y_i, i = 1, \ldots, n\}$ of matrix V. After clustering the rows of matrix V into K clusters, $C_1, \ldots, C_K$, the clusters of observations $A_1, \ldots, A_K$, can be obtained with $A_i = \{j | y_j \in C_i\}$.

In this paper, hierarchical spectral clustering is used to cluster trajectories into several clusters. The trajectories are first clustered into three clusters and then the three clusters are divided into five final clusters as shown in Fig. 2.

As in [9], considering any two trajectories, $A = \{\vec{a}_i\}$ and $B = \{\vec{b}_i\}$, for an observation $\vec{a}_i$ on A, its nearest observation on B is $b_{\psi(i)}$ with,

$$\psi(i) = \operatorname*{argmin}_{j \in B} \|x_i^a - x_j^b, y_i^a - y_j^b\| \tag{1}$$

The directed spatial distance between A and B is

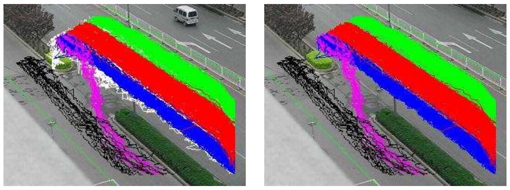
$$h(A, B) = \frac{1}{N_A} \sum_{\vec{a}_i \in A} \|x_i^a - x_{\psi(i)}^b, y_i^a - y_{\psi(i)}^b\| \tag{2}$$

where N_A is the total observation number on trajectory A. Considering the influence of velocity, the directed distance between A and B is

$$f(A, B) = \frac{1}{N_A} \sum_{\vec{a}_i \in A} (\|x_i^a - x_{\psi(i)}^b, y_i^a - y_{\psi(i)}^b\| + \gamma d(u_i^a, v_i^a, u_{\psi(i)}^b, v_{\psi(i)}^b)) \tag{3}$$

where $d(u_i^a, v_i^a, u_{\psi(i)}^b, v_{\psi(i)}^b)$is the dissimilarity measure between velocities of A and B. And γ is a weight coefficient to make sure that the spatial distance and velocity distance have similar scale before adding them. The systematic distance between A and B is

$$F(A, B) = \begin{cases} f(A, B) & \text{if} \quad h(A, B) < h(B, A) \\ f(B, A) & \text{if} \quad h(A, B) > h(B, A) \end{cases} \tag{4}$$



(a) (b)

Fig. 2. The distribution of trajectories in each class after clustering completed in the scene. In the figure, class 1 is defined as blue color, class 2 as green color, class 3 as red color, class 4 as magenta color meaning the vehicle is off the main road and class 5 as black color. Noise trajectories are shown in white color curves.

Then the systematic distance can be transformed to a similarity measure:

$$S(A, B) = \exp(-F(A, B)) \tag{5}$$

In the experiments, $d(u_i^a, v_i^a, u_{\psi(i)}^b, v_{\psi(i)}^b)$ is chosen as,

$$d(u_i^a, v_i^a, u_{\psi(i)}^b, v_{\psi(i)}^b) = 1 - \frac{u_i^a \cdot u_{\psi(i)}^b + v_i^a \cdot v_{\psi(i)}^b}{\sqrt{(u_i^a)^2 + (v_i^a)^2}\sqrt{(u_{\psi(i)}^b)^2 + (v_{\psi(i)}^b)^2}} \tag{6}$$

with $\gamma = 0.02$ in our experiments, and $d(u_i^a, v_i^a, u_{\psi(i)}^b, v_{\psi(i)}^b)$ ranges from 0 to 2. To make sure that the spatial distance and the velocity distance is similar in scale, the height and width of the image in the experiment are normalized to 1 before clustering.

Under the similarity measure detailed up, two trajectories are similar only if they are similar both in spatial position and velocity. Note that before clustering we do not need to resample trajectories with fixed equal length, and we also can deal with broken trajectories. In the training stage, as in [9], before clustering we first detect and remove noises in the trajectories. These noises may come from occlusion or camera twittering. Using directed distance, for each trajectory, after finding its N nearest trajectories, we compute the average distance, and then reject trajectories with large average distance to neighbors as noises as shown in Fig. 2(a) with white color curves. In our experiments, we select $N = 10$, and the average distance is 0.04.

After noises removal, similarity matrix for spectral clustering could be built. Then the trajectory clustering results are shown in Fig. 2(a) and Fig. 2(b). Fig. 2(a) shows the clustering results with noises shown and Fig. 2(b) shows the clustering results without noise shown.

C. Density Estimation

As in [3], [5], [12], after trajectory clustering, each cluster of trajectories are assumed to be distributed according to mixtures of Gaussian probability density function. Hence, each cluster k is characterized by its mean vector $\boldsymbol{\mu}_k$ and covariance matrix $\boldsymbol{C}_K$. Using the density distribution we can complete the detection and prediction of the activities. But this method will be failed when the trajectories can not be described in mixtures of Gaussian functions. Some nonparametric density estimation method, such as kernel density estimation (KDE) method, can be used to avoid the limitation.

Kernel density estimation is a nonparametric density estimation method [18]. In KDE, a kernel centered at each observation is used to obtain a continuous probability density function (PDF) of the data. In fact, we do not need the continuous probability density function in traffic surveillance, as the vehicles moving with regular patterns appear at similar positions on the image. In our paper, we proposed a method to reflect the regular pattern in each cluster avoiding the computation of continuous PDF. Our method considers the average density of one point in each cluster.

III. PROPOSED METHOD

Vehicle tracking, noises detection and trajectories clustering are detailed in previous section and the results are shown in Fig. 2. In all our experiments, we set manually an entry zone and an exit zone in the region of interesting (ROI). Only trajectories that begin in the entry zone and end in the exit zone are retained in the training stage for further processing. But all the trajectories which are detected in ROI are considered in the testing stage, even though the track is not begin in the entry zone or end in the exit zone.

Then, using the trajectories in each cluster we will estimate the density distribution of one point on trajectory to build the statistical description models. We introduce a compactness function to evaluate the compactness of the trajectory with the regular pattern in each cluster, then activity classification and abnormal detection will be completed according to the compactness. These details are given next.

A. Kernel Density Estimation

We use exponential kernel function to evaluate the density of one point on trajectories. For a point $(x_i^a, y_i^a, u_i^a, v_i^a)$, the kernel density can be computed using its nearest point on each trajectory in the same cluster as the kernel. So we call our method as kernel density function even though we do not compute the continuous probability density function. Our kernel density function is based on the separable sum of individual kernels in the spatial dimensions and the orientation dimensions and the speed magnitude dimensions. On the other hand,we just select the nearest point as the kernel to estimate the density. Our method can be described as follows.

Considering a trajectory A in class j and any trajectory B in the same class, the kernel density of the ith point $(x_i^a, y_i^a, u_i^a, v_i^a)$ on trajectory A based on B can be evaluated as

$$\begin{aligned} kde(i, B) &= (cos\_vel(u_i^a, v_i^a, u_{\psi(i)}^b, v_{\psi(i)}^b) \\ &+ 2spatial\_kde(x_i^a, y_i^a, x_{\psi(i)}^b, y_{\psi(i)}^b) \\ &+ vel\_ratio(u_i^a, v_i^a, u_{\psi(i)}^b, v_{\psi(i)}^b))/4.0 \end{aligned} \tag{7}$$

where the first item in the right of (7) describes the operation of the velocity orientation at point $(x_i^a, y_i^a, u_i^a, v_i^a)$. This item is assessed as

$$cos\_vel(u_i^a, v_i^a, u_{\psi(i)}^b, v_{\psi(i)}^b) = \exp(-0.5d(u_i^a, v_i^a, u_{\psi(i)}^b, v_{\psi(i)}^b)) \quad (8)$$

where the $d(u_i^a, v_i^a, u_{\psi(i)}^b, v_{\psi(i)}^b)$ is as same as Part B in Section II and $(x_{\psi(i)}^b, y_{\psi(i)}^b, u_{\psi(i)}^b, u_{\psi(i)}^b)$ is the nearest point on B of the ith point $(x_i^a, y_i^a, u_i^a, v_i^a)$ on A. This item depicts the similarity between the velocity orientation of two points.

The spatial kernel is evaluated as

$$spatial\_kde(x_i^a, y_i^a, x_{\psi(i)}^b, y_{\psi(i)}^b) = \exp(-\|x_i^a - x_{\psi(i)}^b, y_i^a - y_{\psi(i)}^b\|) \quad (9)$$

We double the second item to make sure that the spatial attribute have similar scale with the temporal information, which is denoted by the sum of the first and the third items in (7).

The third item of (7) describes the kernel in the speed magnitude, or the similarity between the magnitude of velocity at point $(x_i^a, y_i^a, u_i^a, v_i^a)$ and the magnitude of the velocity of its nearest point on trajectory B. This item is

$$vel\_ratio(u_i^a, v_i^a, u_{\psi(i)}^b, v_{\psi(i)}^b) = \exp(-\max(|1 - r_1/r_2|, |1 - r_2/r_1|)) \quad (10)$$

$$r_1 = \sqrt{(u_i^a)^2 + (v_i^a)^2} / \sqrt{(u_{\psi(i)}^b)^2 + (v_{\psi(i)}^b)^2} \quad (11)$$

$$r_2 = \sqrt{(u_{\psi(i)}^b)^2 + (v_{\psi(i)}^b)^2} / \sqrt{(u_i^a)^2 + (v_i^a)^2} \quad (12)$$

Here r_1 and r_2 are the two ratios between the velocity magnitude of the ith point on trajectory A and the velocity magnitude of its nearest point on trajectory B. As normal activity patterns, the velocity direction and magnitude of the ith point will be similar with that of its nearest point on trajectory B. Thus r_1 and r_2 should be close to 1. We select the larger one between $|1-r_1/r_2|$ and $|1-r_2/r_1|$ to describe the dissimilarity between the ith point on trajectory A and its nearest point on trajectory B.

So the kernel density for the ith point on A in class j, based on all the trajectories in class j can be evaluated by

$$k(i) = \sum_{B \in classj} kde(i, B) \quad (13)$$

Then the average kernel density for one point on trajectory A is computed by

$$\begin{aligned} kp(A) &= (\sum_{i \in A} k(i))/poin\_num \\ &= (\sum_{i \in A}(\sum_{B \in classj} kde(i, B)))/poin\_num \quad (14) \end{aligned}$$

where, the $poin\_num$ is the length of trajectory A.

For every trajectory we can get the average kernel density of one point in each class and its distribution is shown in Fig. 3.

Then the results of Fig. 3 are used to compute the mean kernel density of a point in each class. The results of the means are shown in table I. Where the mean μ_j of class j is evaluated by

$$\mu_j = (\sum_{A \in classj} kp(A))/Traj\_num \quad (15)$$

here $Traj\_num$ is the total number of trajectories in class j.

In table I, σ is the parameter selected according to experiments. These parameters will be used to evaluate the compactness of each trajectory in the testing stage.

According to the distribution of the average kernel density in each class, we can see that the kernel density distribution is very compact in the same class and have different ranges in different class. Each trajectory is composed by several points, and as normal trajectories should have similar spatial distribution with similar velocities when they move in the same region. Therefore we select the nearest point to evaluate the average kernel density for every point is feasible. To do this, we do not need to resample the trajectory with equal length, and in the testing stage, this make sure that the density is only related to the points before.

Along with section II and this part of section III, we gives the details of all the work in the training stage. Then, in the next part, introducing a factor of compactness, we will give the details of the work in the testing stage.

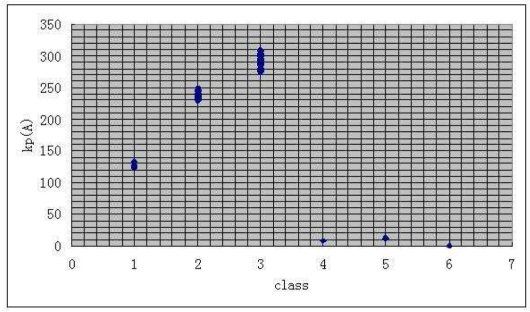


Fig. 3. The distribution of $kp(A)$, for any trajectory A in every class. In the graph, class 6 comprises noise trajectories and the density is set to 0.

TABLE I
PARAMETERS FOR GAUSSIAN FUNCTION

class	μ	σ
1	128.117	6
2	238.728	12
3	295.595	16
4	7.839	0.2
5	11.589	0.5

B. TRAJECTORY CLASSIFICATION AND ABNORMAL DETECTION

In the testing stage, in order to classify the trajectory we introduce the compactness to measure the similarity of kernel density of the trajectory with the mean of kernel density in each class.

When a vehicle moves in the ROI, through tracking, the trajectory T with spatial and velocity attributes can be got. Given that the vehicle moves with activity pattern of class j,

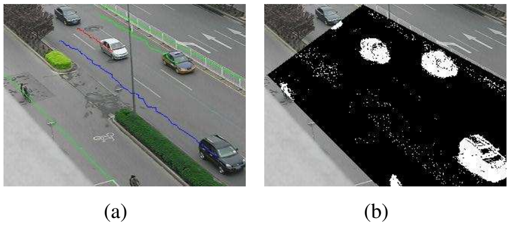


Fig. 4. An example for ROI. Two green curves are used to define ROI. As in Fig. 2, different color curves are used to depict different class the trajectory belongs to. In all of our results, the curves with different color mean that the car moves in different lanes, namely with different activity patterns as shown in Fig. 2.

then employing our kernel density function described previously, the average kernel density for one point $kp_{(t,j)}(T)$ at time t of the trajectory T is computed. Here t denotes time (in frames). Since the vehicle enter ROI, t will increase until the vehicle vanishes from tracking, then $kp_{(t,j)}(T)$ at time t is,

$$kp_{(t,j)}(T) = (\sum_{i=1}^{t} \sum_{B \in classj} kde(i, B))/t \qquad (16)$$

After computing the average kernel density of time t, the compactness of the trajectory T with class j is $P_T(t, j)$ which is computed as,

$$P_T(t, j) = \exp(-(kp_{(t,j)}(T) - \mu_j)^2/2\sigma_j^2) \qquad (17)$$

$P_T(t, j)$ is the compactness of the average kernel density at time t with μ_j, the mean of kernel density of class j computed in the training stage. $P_T(t, j)$ is similar with the role of probability that this trajectory belongs to class j, but the sum of $P_T(t, j)$ of all classes is not 1.

After computing the average kernel density $kp_{(t,j)}(T)$ and the corresponding compactness $P_T(t, j)$, for $j = 1, \ldots, 5$, the most compact class is,

$$c = \underset{j}{\operatorname{argmax}}(P_T(t, j)) \qquad (18)$$

So the vehicle at time t is moving with activity pattern of class $c, c \in \{1, \ldots, 5\}$, and if the maximum of compactness, $\max(P_T(t, j))$, is less than 0.05 for consecutive 15 frames, we set $c = 6$ as abnormal activity, when a vehicle moves with very high speed or in the opposite direction or very far away from the road.

IV. EXPERIMENTS

In the experiments, the data of video lasting more than one hour with size 320×240 is captured by an off-the-shelf CCD camera. The camera is fixed on a tall building and overlooks the traffic scene. The computer is equipped with Intel® Core$^{\text{TM}}$ 2 Duo processor P8700 (2.26GHz processor, 1.98GHz RAM). The algorithm is only applied in the ROI set manually shown in Fig. 4.

The clustering results of trajectories in the training stage are shown in Fig. 2(a) and Fig. 2(b) separately. In all the figures we use different color curves to represent different activity patterns. Blue curves represent class 1 where vehicles move straight along the lane near which the right-turn

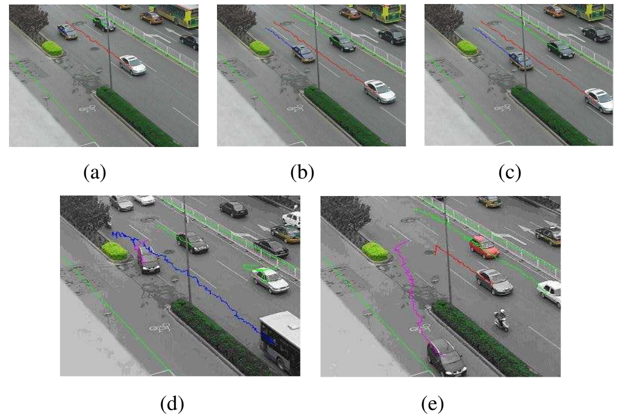


Fig. 5. Experiments results. The first three images are examples for vehicles move in different straight roads. And last two graphs show a car which is turning right.

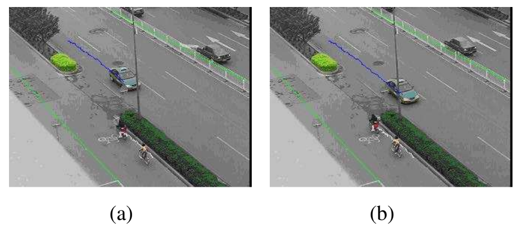


Fig. 6. An experimental result of abnormal activity detection. In the figure a bike with two people, as they are large enough to deal with, we detect the abnormal of the activity, as it moves in the opposite direction.

happened. Green curves mean trajectories in class 2 and red curves are for trajectories in class 3 in which the vehicle moves in the middle of three straight moving roads. Magenta curves are for trajectories in class 4 where vehicles move from the main road turning right to the side road. Black curves are for trajectories in class 5, and in this class vehicles move straight on the side road.

Using the compactness measure of (17) and (18), each trajectory is assigned to the cluster it most likely belongs to. Fig. 5, Fig. 6, Fig. 7 and Fig. 8, are the results in our experiments.

In the scene, there are five activity patterns, three straight moving on the main road and one straight moving on the side road, another is right-turning from the main road to the side

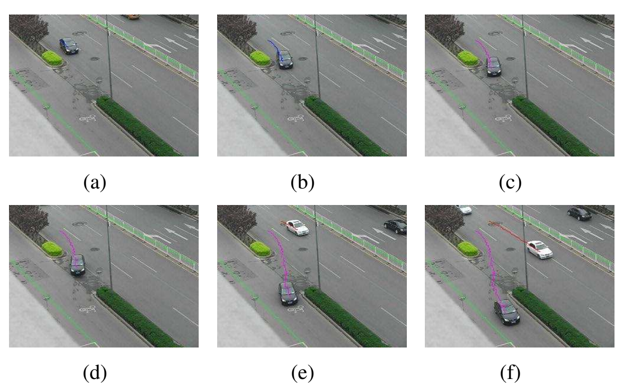
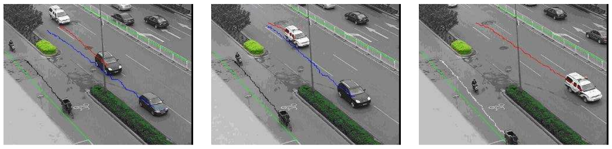


Fig. 7. Another example for right-turning detection.



(a) (b) (c)

Fig. 8. Some experimental results. In the figure an motor first moves normally in 5th class. But after a while it moves so near to one side. So it is detect as abnormal. And at the same time, lane-changing can be detected too, a black car first moving with activity pattern 3, and then it changes to activity pattern 1 in the second graph.

road. Fig. 5 gives the example of right-turning. Fig. 6 gives the detection of an object moving in the opposite direction with white color trajectory. Turning right from the main road to the side road, the car is detected and classified correctly in Fig. 7. Also the experiments show that lane-changing can be detected correctly, as shown in Fig. 8 where a black car first moves in the straight lane with activity pattern of class 3 in front of the white car, shown in Fig. 8(a) and then it changes its lane to the activity of class 1 in Fig. 8(b). Fig. 8 also shows an abnormal detection for the motor moves far away from the center of the road. The black car first moves in the straight lane, and then it turns right off the side road. All the process of the moving is detected correctly in the experiments.

As the average kernel density is considered, just one point can not decide the state of the moving. As shown in Fig. 7, the car first moves in class 1, as the right-turn is not obvious, it has almost equal opportunity of class 1 and class 4. After a few frames, the right-turn is obvious enough to decide the car is in class 4 activity pattern. It is the same when detect the abnormal activities, shown in Fig. 6 and Fig. 8 in which we consider the activity is abnormal only if the vehicle moves in total 15 consecutive frames with all the value of $P_T(t, j)$ in every class is less than the threshold value 0.05.

In the experiments, we successfully detect and classify each vehicle into its activity pattern. Even the lane-changing can be detected correctly when a vehicle moves from one lane to another. If there are 3 vehicles in the same frame to deal with, the time for processing is about 10 frames a second. We believe that after optimization we can do the experiments real-time.

V. CONCLUSIONS AND DISCUSSIONS

In this paper, a novel kernel density estimation method is proposed for understanding and detecting activity patterns. Using the spectral clustering algorithm, the trajectories are clustered into five clusters: three straight moving patterns, one right-turning pattern and one pattern moving on the side road. Based on our kernel density estimation method, the average kernel density of one point on each trajectory in each class is computed in the training stage. Gaussian kernel function is used to evaluate the compactness of the trajectory to each class. We say that the vehicle moves in the activity pattern j, if and only if the compactness to class j is the largest. Even though we have not got the probability distribution of each class, we detect and classify the vehicle activity successfully, obtaining a good understanding of activity patterns.

In the future, more work will be focused on the prediction of activities and more semantic explanation would be given to activity patterns. Also, we can use the high-level information to help to resolve the occlusion which we have not considered here.

REFERENCES

[1] C. Stauffer and Eric Grimson, Learning Patterns of Activity Using Real-Time Tracking, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, pp 747-757, 2000.

[2] Y.-K. Jung, K.-W. Lee and Y.-S. Ho, Content-Based Event Retrieval Using Semantic Scene Interpretation for Automated Traffic Surveillance, *IEEE Transactions on Intelligent Transportation Systems*, vol. 2, pp 151-163, 2001.

[3] W. Hu, X. Xiao, Z. Fu, D. Xie, T. Tan, and S. Maybank, A System for Learning Statistical Motion Patterns, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, pp 1450-1464, 2006.

[4] C. Piciarelli and G.L.Foresti, On-line trajectory clustering for anomalous events detection, *Pattern Recognition Letters*, vol.27, pp 1835-1842, 2006.

[5] B. T. Morris and M.M.Trivedi, Learning, Modeling, and Classification of Vehicle Track Patterns from Live Video, *IEEE Transactions on Intelligent Transportation Systems*, vol.9, pp 425-437, 2008.

[6] C. Piciarelli, C. Micheloni and G. L. Foresti, Trajectory-Based Anomalous Event Detection,*IEEE Transactions on Circuits and Systems for Video Technology*, vol.18, pp 1544-1554, 2008.

[7] I. Saleemi, K.Shafique, and M. Shah, Probabilistic Modeling of Scene Dynamics for Applications in Visual Surveillance, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.31, pp 1472-1485, 2009.

[8] A. J. Joshi, N.P. Pananikolopoulos, Learning to Detect Moving Shadows in Dynamic Environments, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, pp 2055-2063, 2008.

[9] X. Wang, K. Tieu, and Eric Grimson, Learning Semantic Scene Models by Trajectory Analysis, *in European Conference on Computer vision*, pp 110-123, 2006.

[10] X. Wang, X. Ma, and Eric Grimson, Unsupervised Activity Perception by Hierarchical Bayesian Models, *IEEE Computer Society Conference on Computer Vision and Pattern recognition*, pp 1-8, 2007.

[11] X. Wang, X. Ma, and Eric Grimson, Trajectory Analysis and Semantic Region Modeling Using A Nonparametric Bayesian Model, *IEEE Computer Society Conference on Computer Vision and Pattern recognition*, pp 1-8, 2008.

[12] F. I. Bashir, A.A. Khokhar, and D. Schonfeld, "Object Trajectory-Based Activity Classification and Recognition Using Hidden Markov Models, *IEEE Transactions on Image Processing*, vol. 16, pp 1912-1919, 2007.

[13] C. R. Jung, L. Hennemann, and S.R. Musse, Event Detection Using Trajectory Clustering and 4-D Histograms, *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, pp 1051-1575, 2008.

[14] B. T. Morris and M.M. Trivedi, A Survey of Vision-Based Trajectory Learning and Analysis for Surveillance, *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, pp 1114-1127, 2008.

[15] K. F. Wang, Z. J. Li, Q. M. Yao, X. Qiao, S. M. Tang, and Fei-Yue Wang, Moving Object Refining in Traffic Monitoring Applications, *IEEE International Conference on Intelligent Transportation Systems*, pp 540-545, 2007.

[16] C. Stauffer and Eric Grimson, Adaptive Background Mixture Models for Real-Time Tracking, *IEEE Computer Society Conference on Computer Vision and Pattern recognition*, pp 246-252, 1999.

[17] U. V. Luxburg, A Tutorial on Spectral Clustering, *Statistics and Computing*, vol. 17, pp 395-416, 2007.

[18] J. N. Hwang, S.R. Lay, and A. Lippman, Nonparametric Multivariate Density Estimation: A Comparative Study, *IEEE Transactions on Signal Processing*, pp 2795-2810, 1994.