Learning and Classification of Trajectories in Dynamic Scenes: A General Framework for Live Video Analysis

Brendan T. Morris and Mohan M. Trivedi
Computer Vision and Robotics Research Laboratory
University of California, San Diego
La Jolla, California 92093-0434
{b1morris, mtrivedi}@ucsd.edu

Abstract

This paper presents a general framework for live video analysis. The activities of surveillance subjects are described using a spatio-temporal vocabulary learned from recurrent motion patterns. The repetitive nature of object trajectories is used to build a topographical scene description where nodes are points of interest (POI) and the edges correspond to activity paths (AP). The POI are learned through as a mixture of Gaussians and AP by clustering trajectories. The paths are probabilistically represented by hidden Markov models and adapt to temporal variations using online maximum likelihood regression (MLLR) and through a periodic batch update. Using the scene graph, new trajectories can be analyzed in online fashion to categorize past and present activity, predict future behavior, and detect abnormalities.

1. Introduction

The dramatic decrease in cost for quality video equipment coupled with the ease of transmitting and storing video data has led to its widespread use. Cameras are in use all around, along highways to monitor traffic, for security of airports and other public places, and even in our homes. Because of these huge volumes of video data, it is necessary to develop efficient methods for video management. It becomes almost an impossible task to continually monitor these video sources manually. Researchers desire automatic methods to recognize events and activities of interest as they provide a means to compress the video data into a more manageable form as well as annotations for retrieval. This work seeks to relax the constraints on a priori scene information to develop a general framework to analyze surveillance video.

Rather than requiring specific domain knowledge when analyzing video, we restrict ourselves to surveillance applications where events of interest are typically evidenced by motion. Often the observed motion patterns in visual surveillance systems are not completely random but have some underlying structure. This structure dictates the types of activities expected in a scene. By observing motion trajectories over time it is possible to build up models that allow for accurate inferencing. Pioneering work by Johnson and Hogg [1] described outdoor motions with a flow vector $f = [x, y, dx, dy]$ and learned paths using a leaky neural network. Owens and Hunter [2] extended this work using a self organizing feature map to detect abnormal behavior. Stuaffer and Grimson [3] learned paths in a hierarchical fashion by building up a co-occurrence of codebook flows. Hu *et al*. [4] sped up the learning process by using an entire trajectory as input to their batch learning algorithm. They also introduced a method to make predictions based on their path models. Makris and Ellis [5] developed a method to learn the interesting regions in a scene as well as build spatial paths in an online fashion allowing adaption to new unseen trajectories. Picarelli and Foresti [6] developed another online system to decompose subpaths into a tree-like structure for efficient data sharing and simple prediction. Further references on trajectory learning and modeling can be found in the review by Morris and Trivedi [7].

This paper extends the above work to provide a general framework for automatically analyzing a surveillance scene. Low level object tracking is leveraged to learn a description of the scene. The interesting regions in an image are learned and recurrent trajectories build up activity paths between the regions. The learned paths define the spatio-temporal dynamics of a scene which can be used for detailed scene analysis.

978-0-7695-3341-4/08 $25.00 © 2008 IEEE
DOI 10.1109/AVSS.2008.65

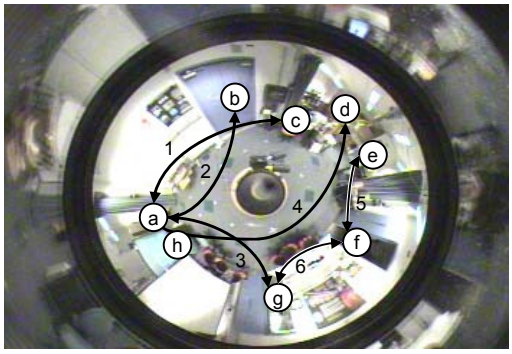


Figure 1. Topographical representation of a scene. Graph nodes represent points of interest (POI) and the edges depict typical motion encoded in an activity path (AP).

2. Topological Scene Description

Rather than manually specifying activities of interest for a particular scene, which can become prohibitive, a model can be automatically built through observation of motion. Object motions map out patterns that are often not random but drawn from some underlying distribution. This inherent structure and redundancy can be leveraged to describe the spatio-temporal characteristics of a scene. Trajectory information is extracted by visual tracking of moving objects and is the primary feature for scene model construction. This model is represented as a topographical map [5] with nodes corresponding to points of interest (POI) and the edges correspond to paths (AP) which encode activity. An example of this map is depicted in Fig. 1.

2.1. Points of Interest

The entry and exit zones are the locations where objects either enter or exit the camera field of view (FOV) or where tracking targets appear and disappear. These zones are modeled as a mixture of Gaussians (MoG), $Z \sim \sum_{i=1}^{W} w_i N(\mu_i, \Sigma_i)$ with W components. A zone is learned using expectation maximization (EM) [8] using a dataset formed from the initial position of a trajectory for start zones and the final position for stop zones. The zones are over mixed, large W, to cover all true zones and noise sources, which are separated with a density criterion [5]. Tight mixture components with high density indicate a true zone while low density mixtures imply tracking noise as they are not localized. Fig. 2(a) shows the entry/exit zones learned for an intersection with green denoting entry and red exit POI. Noise mixtures, drawn in black, exist in Figs. 6(a) and 7(a) depict broken tracks.

The second type of POI comes from scene landmarks where objects tend to idle or remain stationary for some time, *e.g.* queue of vehicles at a toll booth. These stop zones are locations that can be defined in two different ways. Either as any tracking points with speeds less than a predefined threshold [5] or as all the points that remain in a circle of radius R for more than τ seconds [9]. By defining a radius and time constant, the second measure ensures objects actually remain in a particular location wile the first could allow noisy points from slow moving targets, as would be the case in a congested scene. The stop-point dataset is compiled using both these methods and another MoG is learned for the stop zone as done above. In Fig. 7(a) the stop zones are shown in yellow. Notice the desk in the upper right and smart board in the lower left are correctly discovered but stop zones were also found overlapping with entry/exit zones because there is very little pixel deviation far away from the omni camera.

2.2. Route Clustering

The topographical scene map is completed by learning the edges between POI nodes. The edges depict the acceptable paths and specify how objects move between nodes. Each edge encodes a separate activity and can be learned by unsupervised clustering. The main difficulty is the time-varying nature of activities which lead to unequal length trajectories.

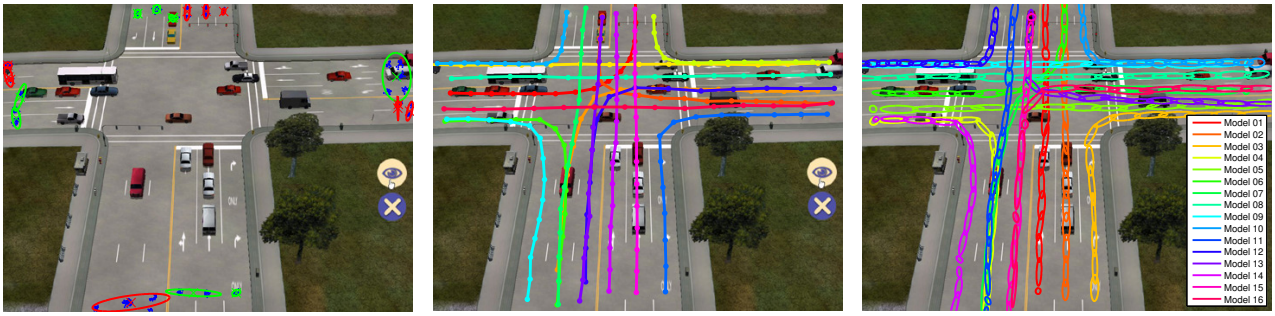
A activity training database is accumulated by collecting trajectories over sufficient time. The trajectories are processed to learn the POI which are used to filter the database. Bad trajectories are removed because they do not well represent the scene AP. Tracks are considered bad if they do not start and end in an entry and exit respectively. Trajectories that travel through a stop zone are split into separate tracks leading into and out of the stop zone. Initially activity dynamics are ignored and only the route or spatial location between POI is learned. Each trajectory $F_T = \{f_1, \ldots, f_T\}$ of length t is linearly resampled to a fixed length L in order to be more easily clustered. The new trajectory representation $\hat{F}_L = \{\hat{f}_1, \ldots, \hat{f}_T\}$ if designed such that

$$d(\hat{f}_i, \hat{f}_j) \sim \frac{1}{L-1} \sum_{l=1}^{T-1} d(f_l, f_{l+1}) \quad (1)$$

$$d(f_i, f_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}. \quad (2)$$

This determines the total xy distance traveled and divides it equally among the L sample points. This prevents regions of higher sample density from contributing bunches of points in a single area causing bias toward the manner an activity was performed rather than just the underlying activity.

A flow vector [1] $F = [x_1, y_1, \ldots, x_L, y_L]$, representing a point in the $\mathbb{R}^{2L}$ route space, is constructed for each of the N training trajectories. The training set is partitioned in N_c clusters using fuzzy C means (FCM) because it has soft class assignment that minimizes the effects of outliers. After clustering, the the major scene paths are encoded by



(a) (b) (c)

Figure 2. (a) Entry/Exit interest zones learned by fitting Gaussian mixture model. (b) Routes learned through fuzzy C mean (FCM) clustering of trajectories. (c) Spatio-temporal path dynamics modeled with hidden Markov models (HMM).

prototypes r_k, $k = 1, \ldots, N_c$, and the membership of a training example i to route k is given by u_{ik}.

2.2.1 Route Merging

The number of paths, N_p, in an arbitrary scene is not known a priori and must be estimated. Initially, FCM clusters into a large number of prototypes, $N_c > N_r$, and these are then refined to a smaller number of routes (N_r) by merging similar clusters (we will henceforth assume merging is accurate and $N_p = N_r$). The routes are linearly resampled for evenly distributed points then compared using dynamic time warping (DTW) [10] to find the optimal alignment between route points. Two clusters, r_m and r_n, are considered similar if after optimal alignment all consecutive points are within a small radius,

$$d_t(r_m, r_n) = \sqrt{(x_t^m - x_t^n)^2 + (y_t^m - y_t^n)^2} < \epsilon_d \quad \forall t, \tag{3}$$

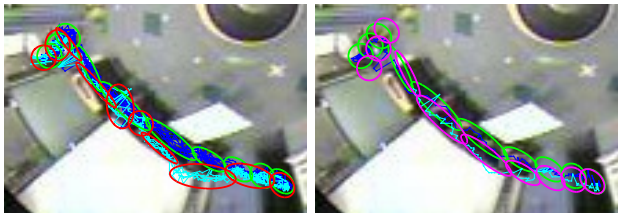
or if the total distance between tracks is small enough,

$$D = \sum_{t=1}^{L} d_t < \epsilon_D = L\epsilon_d. \tag{4}$$

In our experiments component cluster points are considered close enough for merging when $\epsilon_d = 5$ pixels. A cluster correspondence list is created from these pairwise similarities, forming similarity groups V_s. Each correspondence group is reduced to a single route

$$r(V_s) = \underset{z \in V_s}{\operatorname{argmin}} \sum_{i=1}^{N} |\hat{u}_{ik}(z) - \tilde{u}_{ik}(z)| \tag{5}$$

by retaining only the prototype that would cause maximal change in training membership if removed. The membership $\hat{u}_{ik}(z)$ represents the normalized membership when prototype z is removed from the route set and $\tilde{u}_{ik}(z)$ is the membership after using FCM to cluster with starting membership $\hat{u}_{ik}(z)$.



(a) (b)

Figure 3. Online MLLR update allows adjustments on the fly. The initial path was blocked by a table forcing a re-routing around the table. (a) Path learned by applying a batch MLLR update. (b) Path adapted using the incremental MLLR update.

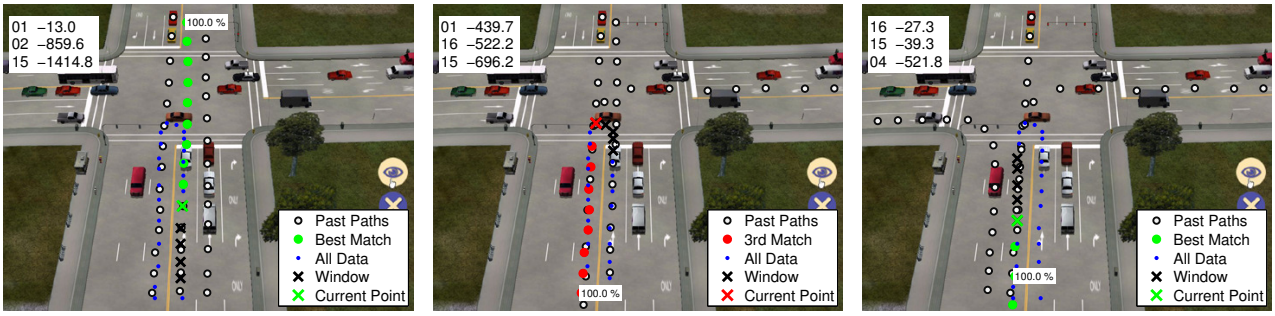
In practice we have found that FCM tends not to over fit the data but instead find several very similar clusters, making this simple merge algorithm effective. Fig. 2(b) shows the paths learned by the FCM clustering and merge procedure. The initial clustering used $N_c = 25$ but after merging only the $N_r = 16$ true lanes remain.

3. Path Modeling

The FCM procedure locates paths spatially but this is insufficient for behavior analysis. Not only do we need to know where objects are located by also the manner in which they travel along a route to completely characterize a behavior. Using HMMs, the spatio-temporal properties of every path is encoded, differentiating not only location but also dynamics. The advantage of modeling paths by HMMs is simplicity of training and evaluation. In addition, an HMM naturally compares different length tracks through optimal time normalization. Unlike with FCM clustering, the full unsampled trajectories containing position and velocity are used to incorporate dynamics into the activity path.

3.1. HMM Path

The HMM path is a hidden Markov model (HMM) based on the structure of trajectories. Each HMM is compactly



(a) (b) (c)

Figure 4. Unusual tracking behavior during illegal u-turn. This is viewed as an activity change because the path changes from 1 to 16. (a) Acceptably traveling north in Path 1. (b) Suspicious event marked with a red x during u-turn (Still in Path 1). (c) Recovered into acceptable southbound lane, Path 16.

represented as $\lambda_k = (A, B, \pi)$ and is designed to have Q states. The parameters A and π are manually defined by the inherent structure of paths.

$$\pi_i = e^{\alpha_p i} \tag{6}$$

$$A_{ij} = \begin{cases} e^{-\alpha_b(i-j+1)}, & j \leq i \\ e^{-\alpha_f(j-i-1)}, & j > i \end{cases} \tag{7}$$

The rows of π_i and A are normalized to be valid probabilities. The transition rates are chosen such that $\alpha_f \ll \alpha_b$ for strong left-right tendencies. These parameter definitions set up the model that mimics the natural progression from the beginning of a path to the end but also allow starts in the middle of a path (useful for broken tracks and online analysis). The internal HMM states $\{q_j\}_{j=1}^{Q}$ are assumed to be Gaussian mixtures with unknown mean and covariance $q_j \sum_{m=1}^{M} \sim N(\mu_{jm}, \Sigma_{jm})$ with M specifying the number of activities in each path (all our experiments use $M = 1$ because we assume just a single typical behavior for each path).

3.2. Training Procedure

The HMM states, relating the position and velocity, must be learned for an AP. Each HMM is automatically trained by dividing the training set into N_p disjoint sets, $D = \bigcup_{k=1}^{N_p} D_k$, corresponding to each route. The set D_k is constructed by collecting all trajectories classified into cluster r_k based on membership

$$r_i^\star = \underset{k}{\operatorname{argmax}}\, u_{ik} \quad \forall i. \tag{8}$$

Only those trajectories with $u_{r_i^\star k} > 0.9$ are retained when creating the path training set D_k because they can be confidently placed into route $r_i^\star$. The N_p HMMs ($\lambda_k = (A, B, \pi)$) to be learned using standard methods such as the Baum-Welch method or EM [10]. The set of HMMs learned for the traffic intersection are shown in Fig. 2(c).

The AP are learned in unsupervised manner because "good" training trajectories are found through POI filtering and the membership confidence threshold. Since no manual intervention is necessary, the topographical map is easily generated for quick deployment.

3.3. Model Update

The path models learned through the above procedure accurately depicts the scene at the time of training but in a surveillance scene there is no guarantee that processes are stationary. The paths will be dynamic and the model must track changes. There are two complimentary methods for updating the path models, the first in online fashion [11] and the second through periodic batch refresh [4].

When a trajectory is drawn from a particular activity it can be used to update the HMM with maximum likelihood linear regression (MLLR) [11] allowing non-stationary processes. MLLR computes a set of linear transformations that will reduce the mismatch between the initial model and the new adaption data. The adapted state mean is given by

$$\hat{\mu} = W\xi, \tag{9}$$

where W is the $d \times (d+1)$ transformation matrix and $\xi = [1, \mu_1, \ldots, \mu_d]^T$ is the extended mean vector. $W = [b\ A]$ produces and affine transformation for each Gaussian HMM state mixture component with A a $d \times d$ transformation and b a bias term. The transformation matrix W can be found using EM. Each time a trajectory is classified into path λ_k, a transformation is learned and applied to sequentially update the HMM. The update

$$\mu_{t+1} = (1-\alpha)\mu_t + \alpha_\mu W_t \xi_{kj} \quad j = 1, \ldots, Q \tag{10}$$

modifies the mean of existing path λ_k where the $\alpha_\mu \in [0, 1]$ is a learning rate parameter. The online regression update is demonstrated in Fig. 3, where a path is blocked by a table and people are forced to walk around.

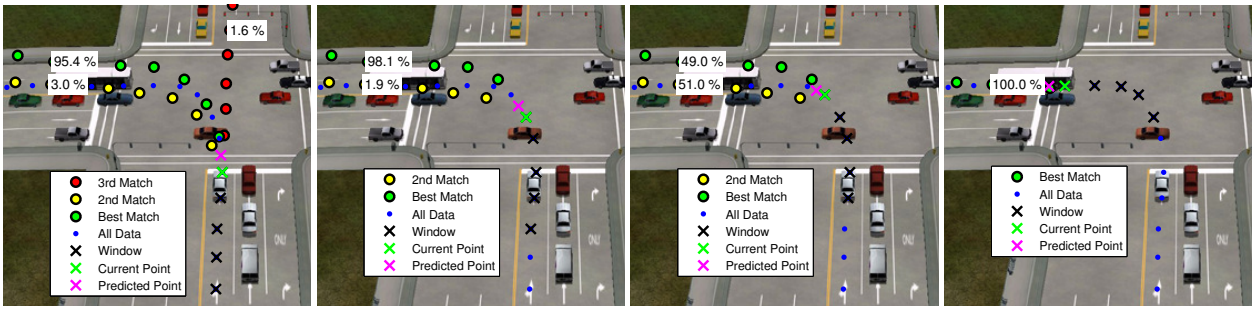


Figure 5. Left turn prediction with probability of best paths displayed.

Trajectories that do not fit any of the activity models well and are considered anomalous are collected into batch update database [4]. Once the database has grown large enough the AP learning procedure can be reapplied to add new edges in the scene graph. In this way, motions initially considered atypical could be assimilated into the scene activity models if it accumulated enough support.

4. Behavior Analysis

To analyze an object's behavior, it is necessary to place it into a corresponding path at all times. When objects do not fit into a path model well it indicates the detection of an anomalous event. This is detection is made more difficult when in an online setting because only a portion of the entire track is seen at a given time, meaning the the behavior inferencing must be done with incomplete data.

4.1. Trajectory Classification

Each trajectory can be placed into the appropriate path with probabilistically Bayesian inferencing. Each new trajectory is assigned the label of the path that best explains it

$$\lambda^\star = \underset{k}{\operatorname{argmax}}\, P(F|\lambda_k) \tag{11}$$

which can be solved for HMM λ_k using the forward-backward procedure [10]. The label of the maximum likelihood HMM determines the activity of each new datum.

4.1.1 Anomalous Trajectories

While every track will be classified into a path $\lambda^\star$, the quality of this assignment will be low for abnormal activities since outliers are not well modeled. These abnormal trajectories can be recognized as those with low likelihood, $\log P(F|\lambda^\star)$ less than a threshold LLT_p. The decision threshold is learned during training by comparing the average likelihood of samples in training set D_k to those outside.

$$LL_{in} = \frac{1}{|D_k|}\sum_{i\in D_k} \log P(F_i|\lambda_k) \tag{12}$$

$$LL_{out} = \frac{1}{N-|D_k|}\sum_{i\notin D_k} \log P(F_i|\lambda_k) \tag{13}$$

$$LLT_p = \beta(LL_{in} - LL_{out}) + LL_{out}. \tag{14}$$

The sensitivity factor $\beta \in [0,1]$ controls the abnormality rate. Larger β values will cause more trajectories to be considered anomalous by increasing the threshold.

4.2. Online Tracking Analysis

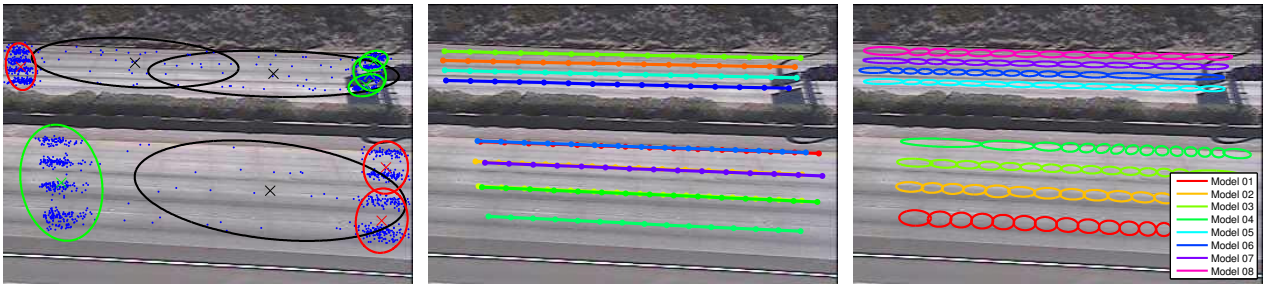
Although it is interesting to analyze complete tracks, it is often more important to recognize and evaluate behavior as it occurs. With each new frame a track is updated, a new path estimate can be made and refined as more information is gathered. Instead of using all the tracking points accumulated at a time t, only a small window is retained as old samples may not correlate well with current behavior. The windowed track consists of past and the present measurements as well as k predicted points, $F^k_{win} = \{f_{t-win}, \ldots, f_{t-1}, f_t, \hat{f}_{t+1}, \ldots, \hat{f}_{t+k}\}$. The future points $\hat{f}_{t+k}$ can be estimated by applying the tracking motion model k time steps ahead.

4.2.1 Live Tracking Classification

Using the windowed track ($k = 0$), the path an object is following at the current time t can be determined by evaluating (11) with F replaced by its windowed version F^k_{win},

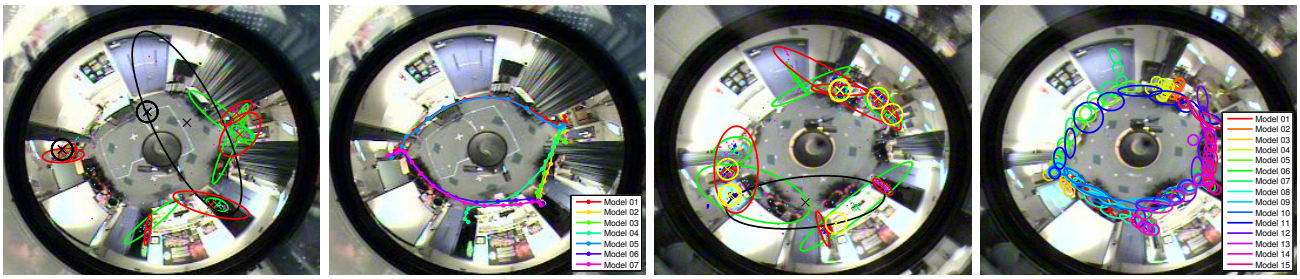
$$\lambda^\star_{win} = \underset{k}{\operatorname{argmax}}\, P(S_{win}|\lambda_k) \tag{15}$$

Over the life of a trajectory S, path estimates are made, encoding a complete path history $\{\lambda_1, \ldots, \lambda_T\}$. A number of consecutive labels indicates a consistent path. Using this transcription allows recognition of lane changes as points when the label changes between consistent paths. The track in Fig. 4 starts in Path 1 and ends in Path 16.



(a) (b) (c)

Figure 6. Interstate 5 (I5) experiments. (a) Learned Entry/Exit zones, enter in green, exit in red, and black indicates noise. (b) Routes after merging. (c) HMM path models.



(a) (b) (c) (d)

Figure 7. OMNI experiments, (a),(b) reference OMNI1 while (c),(d) show the updated OMNI2. (a) Learned zones. (b) Routes after merging. (c) Updated zones, notice the stop zones (yellow). (d) Larger set of OMNI2 activity paths.

4.2.2 Tracking Abnormalities

Since only a windowed version of a trajectory is used during online analysis the the log-likelihood threshold (14) needs to be adjusted. The new threshold is

$$LLT_p^t = \gamma_t \left[\beta_t (LL_{in} - LL_{out}) + LL_{out}\right], \quad (16)$$

$$\gamma_t = \frac{E[q|win]}{Q}. \quad (17)$$

The abnormality threshold is adjusted with γ_t, averaging the log-likelihood into every model state and adjusting for the expected number of states visited in the observation window $E[q|win]$. Here $\beta_t \in [0, 1]$ is again chosen to ensure detection of most suspicious tracking points. We choose the β_t that results in approximately 10% false positive rate on the training set (this assumes there are no usual events in the training set). As soon as an object strays from a path model, an unusual event is triggered allowing for timely detection. Fig. 4 demonstrates tracking abnormality detection where the red x marks the unusual event since u-turns were not allowed.

4.2.3 Path Prediction

Besides detecting abnormalities it is possible to predict behavior using the HMMs. Estimation can be extended further in time than standard one step prediction (Kalman prediction) and is governed by acceptable scene activities rather than a generic motion model. The future path is predicted by finding the probability of remaining in each of the top 3 best fit paths from (15). This probability is estimated by evaluating (15) again with $k > 0$. Fig. 5 shows and example of turn prediction. The probability of the correct path improves as the turn progresses, matching our intuition.

5. Studies and Experimental Analysis

We evaluate the accuracy of classification, prediction, and abnormality detection in the following section. The results are compiled from a set of experimental studies of different scenes; a simulated traffic intersection (SIM), Fig. 2, a highway traffic view of Interstate 5 (I5), Fig. 6, and an indoor laboratory scene from an omni-directional camera (OMNI), Fig. 7. Trajectories were generated by using an adaptive background subtraction scheme [12]. Table 1 summarizes the study results and the parameters used in the experiments are compiled in Table 2.

5.1. Quality of Paths

Before evaluating the performance of the dynamics, the quality of the learned paths is addressed. The true number of lanes in the traffic scenes is known and the number of paths in the lab scene defined based on the training set.

	N_p	lane assignment	abnormality	live lane assignment	live prediction	live abnormality
SIM	16	327/327 = 100%	5/5 = 100%	3669/3978 = 92.2%	2871/3978 = 72.2%	40/46 = 87.0%
I5	8	879/923 = 95%	-	14045/14876 = 94.4%	13859/14876 = 93.2 %	-
OMNI1	7	26/26 = 100%	10/14 = 71.4%	1741/3139 = 55.5%	1454/3139 = 46.3%	756/945 = 80.0%
OMNI2	15	12/16 = 75%	15/18 = 83.3%	1457/2693 = 54.1%	1013/2693 = 37.6%	-

Table 1. Experimental Results

	L	Q	win	k	β	β_t
SIM	25	15	5	3	0.85	0.95
I5	25	15	5	3	0.8	0.95
OMNI	15	15	30	15	0.9	0.65

Table 2. Experimental Parameters

The traffic scenes were able to effectively find lanes. All 16 of the intersection maneuvers were discovered. In the I5 experiment, all 8 of the lanes were discovered but there were also 2 false lanes identified. These extra lanes appear in the southbound direction closest to the camera where perspective distortion causes more variance in localization.

In contrast to the traffic scenes, the omni camera observed lightly constrained motion since there are no physical lanes but virtual lanes appear, mapped between doorways and desks. The first omni experiment (OMNI1) only contained 7 paths which were all correctly discovered. But the OMNI2 experiment was more complex, using the nodes shown in Fig. 1 there were 15 unique paths. Although the path learning system found 15 paths, 2 were not correct but unexpected noise paths (too small stop zone) around the table at node c. Two of the missing paths had little support in the training set and could be learned with more data. These datasets are particularly difficult because there is not very clear separation between paths as there is significant overlap.

5.2. Trajectory Classification and Abnormalities

Using (11) the each sample trajectory is placed into its most likely path. All 327 of the test trajectories from the intersection were correctly classified. The I5 experiment had 879 of 923 test tracks correctly given lane labels for 95% accuracy. Not suprisingly, most errors occured in the northbound lanes (top of Fig 6) where the lanes are quite close because of the camera view. The test set for the OMNI1 experiment was collected over a single Saturday withough participant awareness for natural tracks. The 26 typical trajectories all correctly classified. The OMNI2 test set was collected by test subjects walking through the lab for 30 minutes. In this set only 12 of the 16 (75%) modeled trajectories were correctly assigned to a path. But, 14/16 were in the top 2 best matches and 15/16 in the top 3.

Using $\beta = 0.85$ all 5 of the anomalous trajectories were correctly identified for the intersection experiment. In the first omni experiment, 10/14 abnormal trajectories were detected and 15/18 were discovered in the second experiment. Fig. 8 gives examples of abnormal trajectories int the lab omni experiments.

5.3. Tracking Classification

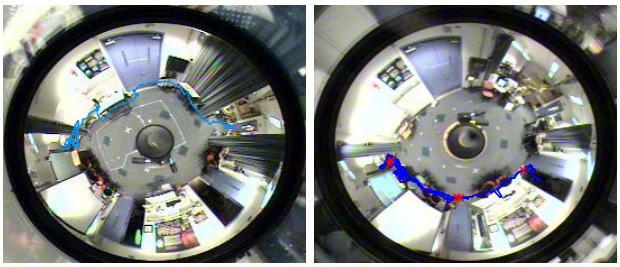
The tracking classification accuracy measures how many individual tracking points had the same label as the trajectory label. The traffic scenes, with the simpler lanes, had high tracking classification results. The accuracy was 92.2% for the intersection and 94.4% for the highway. By contrast, the two omni experiments had accuracy of 55.5% and 54.1% respectively. They suffered degraded performance due to the complexity of the scene such as overlapping paths. When viewing the windowed data, there is little distinction between going from node a to g or from a to f because the routes share significant space. This limited the influence of the velocity in distinguishing activities. Using more historical track data would improve classification but add delay.

5.4. Tracking Prediction

The prediction accuracy is measured by the number of prediction labels that share the same label as the full trajectory. The prediction accuracy for the intersection was 72.2% and was 93.2% for I5. The results for the lab were significantly lower, at 46.3% for OMNI1 and 37.5% for OMNI2. Clearly the prediction scheme works quite well for straight paths but is limited when analyzing more complex routes. The inferred tracking points do not fit the path as well when it is curved given just the local dynamics. If there were hairpin turn in a path, at the top of the u-turn, the assumption is to continue straight ahead rather than double back. The prediction accuracy for the omni scenes similarly suffers from path overlap.

5.5. Abnormalities During Tracking

These anomalies indicate deviation from an AP when it happens and are noted to occur in groups of successive



(a) (b)

Figure 8. (a) Abnormal trajectory along edge of room discovered in OMNI1 training set. (b) Abnormal trajectory because of backtracking along path with red X's indicate the beginning of a tracking abnormality.

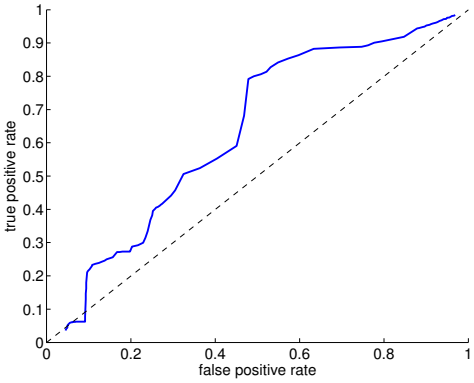


Figure 9. ROC characteristics for live tracking abnormality detection for the OMNI1 experiment.

points proportional to the duration of the abnormality. In the intersection there were 46 anomalous points out of 277 total trajectory points and 40 were correctly detected. While 6 points were missed they were at the beginning of an abnormality group and there was always a detection after a few samples of delay. There were only 3/277 false alarms.

Using $\beta_t = 0.65$ in the OMNI1 experiment, 756/945 abnormal points are detected. While this seems promising it comes at a steep price. The accompanying false positive rate is 49%. Fig. 9 shows the reciever operating characteristic (ROC) curve and dispalys much room for improvement. The curve may be a little misleading because it only accounts for direct correlation between the detected and true abnormality points. A better measure of accuracy might be the detection of any abnormality points within an appropriate delay window. In Fig. 8(b) we show only the first point in a group of live tracking abnormalities. Though it looks to be an acceptable path, the person backtracks between node a and g causing abnormalities before ending at node f.

6. Conclusion

This paper presents a general framework for live video analysis based on trajectory learning. A surveillance scene is described by a topographical map which indicates interesting image regions and the way objects move between these places. These descriptors provide the vocabulary to analyze spatio-temporal dynamics. This allows for activity classification, prediction, and abnormality detection in real-time. The experimental analysis of three varied scenes demonstrates the generality of the trajectory analysis procedure.

References

[1] N. Johnson and D. Hogg, "Learning the distribution of object trajectories for event recognition," in *Proc. British Conf. Machine Vision*, vol. 2, Sept. 1995, pp. 583–592.

[2] J. Owens and A. Hunter, "Application of the self-organising map to trajectory classification," in *Proc. IEEE Visual Surveillance*, July 2000, pp. 77–83.

[3] C. Stauffer and W. E. L. Grimson, "Learning patterns of activity using real-time tracking," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 22, no. 8, pp. 747–757, Aug. 2000.

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

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

[6] C. Piciarelli and G. L. Foresti, "On-line trajectory clustering for anomalous events detection," *Pattern Recognition Letters*, vol. 27, no. 15, pp. 1835–1842, Nov. 2006.

[7] B. T. Morris and M. M. Trivedi, "A survey of vision-based trajectory learning and analysis for surveillance," *IEEE Trans. Circuits Syst. Video Technol.*, 2008, to be published.

[8] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Royal Stat. Soc.*, vol. 39, pp. 1–38, 1977.

[9] N. Brandle, D. Bauer, and S. Seer, "Track-based finding of stopping pedestrians - a practical approach for analyzing a public infrastructure," in *Proc. IEEE Conf. Intell. Transport. Syst.*, Toronto, Canada, Sept. 2006, pp. 115–120.

[10] L. Rabiner and B. Juang, *Fundamentals of Speech Recognition*. Englewood Cliffs, New Jersey: Prentice-Hall, 1993.

[11] M. Gales, D. Pye, and P. Woodland, "Variance compensation within the MLLR framework for robust speech recognition and speaker adaptation," in *Proc. IEEE Intl. Conf. Spoken Language*, Oct. 1996, pp. 1832–1835.

[12] B. T. Morris and M. M. Trivedi, "Improved vehicle classification in long traffic video by cooperating tracker and classifier modules," in *Proc. IEEE International Conference on Advanced Video and Signal based Surveillance*, Sydney, Australia, Nov. 2006.