Correspondence-Free Activity Analysis and Scene Modeling in Multiple Camera Views

Xiaogang Wang, *Student Member, IEEE,* Kinh Tieu, and W. Eric L. Grimson, *Fellow, IEEE*

Abstract—We propose a novel approach for activity analysis in multiple synchronized but uncalibrated static camera views. In this paper, we refer to activities as motion patterns of objects, which correspond to paths in far-field scenes. We assume that the topology of cameras is unknown and quite arbitrary, the fields of views covered by these cameras may have no overlap or any amount of overlap, and objects may move on different ground planes. Using low-level cues, objects are first tracked in each camera view independently, and the positions and velocities of objects along trajectories are computed as features. Under a probabilistic model, our approach jointly learns the distribution of an activity in the feature spaces of different camera's views. Then it accomplishes the following tasks: (1) grouping trajectories, which belong to the same activity but may be in different camera views, into one cluster; (2) modeling paths commonly taken by objects across multiple camera views; (3) detecting abnormal activities. Advantages of this approach are that it does not require first solving the challenging correspondence problem, and that learning is unsupervised. Even though correspondence is not a prerequisite, after the models of activities have been learnt, they can help to solve the correspondence problem, since if two trajectories in different camera views belong to the same activity, they are likely to correspond to the same object. Our approach is evaluated on a simulated data set and two very large real data sets, which have $22,951$ and $14,985$ trajectories respectively.

Index Terms—Visual surveillance, Activity analysis in multiple camera views, Correspondence, Clustering.

1 Introduction

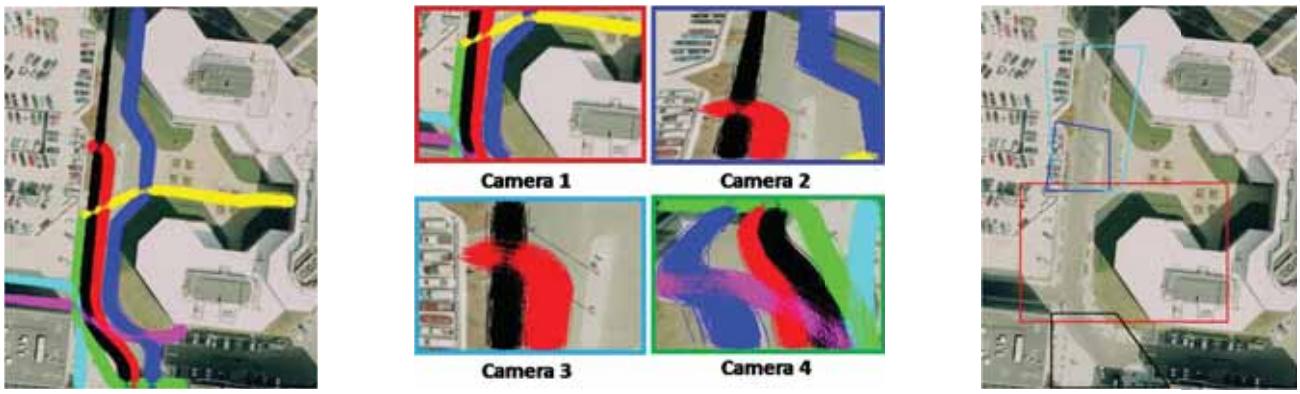
In visual surveillance, a key task is to monitor activities in the scene. People have interest in discovering typical and abnormal activities, detecting activities of some categories, and knowing some structures of the scenes, such as paths commonly taken by objects, sources and sinks where objects appear and disappear. Because visual surveillance systems collect a huge amount of data from many different scenes, people expect the algorithms to be unsupervised with little human labeling effort as possible. When modeling activities, while some approaches [1], [2], [3], [4], [5], [6], [7], [8] directly extract motion and appearance features from video streams without relying on tracking and object detection, in many surveillance systems [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], especially in far-field settings, objects are first detected and tracked, and the activity of an object is then treated as sequential movements along its trajectory. Through tracking, an activity executed by a single object can be separated from other co-occurring activities, and features related to the activity can be integrated as a track. In many far-field surveillance settings, the captured videos are of low resolution and poor quality, and it is difficult to compute more complicated features, such as gestures, local motions, or appearance of objects within the tracks. Usually only positions of objects are recorded along tracks, which are called trajectories. Although quite simple, the motion patterns of trajectories can distinguish many different activity categories, especially in far-field settings. Examples can be found in Figure 1 (a). The goal of this work is to model activities by trajectory analysis: clustering trajectories into different activities, detecting abnormal trajectories, and modeling paths commonly taken by objects.

Many approaches [10], [20], [21], [22], [23], [24], [25], [17], [19] were proposed to cluster or classify trajectories into activities. They used the spatial proximity between a pair of trajectories, measured in different ways, for clustering. Activities are often closely related to the structures of scenes, e.g. roads, paths, entry and exit points, which can help not only high-level description of activities [17], but also low-level tracking and classification [26]. The models of paths commonly taken by objects can be obtained by finding the spatial extents of trajectory clusters [21], [27], [28], [17], [29]. Entry and exit points are detected at the ends of paths [17].

All these clustering and modeling approaches assumed a single camera view whose visible area is finite and limited by the structures of the scene. In order to monitor activities in a wide area, video streams from multiple cameras have to be used. Examples of activities observed in different camera views can be found in Figure 1 (b). Many systems [30], [31], [32], [33], [34], [35], [36], [37], [38], [39], [40], [41], [42], [43], [44], [45], [46], [47], [48] using multiple cameras for visual surveillance have been developed in recent years and they are based on various assumptions on the number of cameras, the topology and geometry of camera views, and camera calibration. Most of these approaches focused on

• *Xiaogang Wang, Kinh Tieu and W. Eric L. Grimson are with the Department of Electrical Engineering and Computer Computer Science, Massachusetts Institute of Technology, Cambridge, MA, 02139.*
E-mail: {xgwang,tieu,welg}@csail.mit.edu

Manuscript received April 29, 2008; revised September 17, 2008; accepted September 18, 2008.



(a) (b) (c)

Fig. 1. Simulated examples of activity categories distinguished by motion patterns. Trajectories of different activity categories are marked by colors. Trajectories are obtained from simulation. (a) Activities observed in a single giant camera view, which is usually unavailable in real life. (b) The same activities as in (a) are observed in four different camera views, which are similar to the camera views available in real life. (c) The fields covered by four camera views. They are marked by ploygons in colors: red (camera 1), blue (camera 2), cyran (camera 3) and black (camera 4).

tracking objects across multiple camera views or doing correspondence of trajectories in different camera views. In general, this is a very difficult problem. Because of the structures of the scenes, the distribution and configuration of these cameras could be quite arbitrary and unknown. The camera views may have any combination of large, small, or even no overlap. The objects in the views may move on one or multiple ground planes. Analyzing activities over such a multi-camera network is quite challenging. A natural way of doing multi-camera surveillance is to first infer the topology of camera views [39], [44], solve the correspondence problem [30], [33], [34], [38], [35], [37], [40], [41], [42], [43], [45], [47], stitching the trajectories of the same object in different camera views into a complete long trajectory, and then analyze the stitched trajectories using the same approaches developed for a single camera view. However both inferring the topology of camera views and solving the multi-camera correspondence problem are notoriously difficult especially when the number of cameras is large and the topology of the cameras is arbitrary. The ultimate goal of some surveillance systems is activity analysis instead of solving correspondence. In this paper, we show that activity analysis in multiple camera views can be accomplished without solving the correspondence problem.

We propose an approach to group trajectories, which belong to the same activity but may be in different camera views, into one cluster and model the paths of objects across camera views. They are jointly learnt under a probabilistic model. Our approach is completely unsupervised and does not require the correspondence problem to be solved in advance. The camera settings in our approach are as follows.

- The cameras are static and synchronized but do not have to be calibrated.
- The fields of view covered by these cameras may have no overlap or any amount of overlap. However we assume that when an object exits a camera view, it is already in or will enter one of the other camera views within time T.
- Objects may move on different ground planes.

Examples of multi-camera settings are shown in Figure 2.

We briefly explain several basic concepts and assumptions held in this paper. There are paths in the physical world. Objects move along these paths and thus have different moving patterns (examples of different moving patterns can be found in Figure 1), which are called activities. A path may be observed in multiple camera views and has spatial distributions in these views. Although some paths, such as roads of vehicles can be recognized by their physical features, some paths cannot be. For example, pedestrians take a short cut on a grass field. A trajectory, which only records the positions of an object, is a history of the movement of an object in a camera view. The points on trajectories are called observations. In this work, trajectories are clustered into different activities, based on their spatial distributions and moving directions. A cluster of trajectories is related to a path in the physical world. The scene of a camera view is quantized into small cells. When an object moves, it connects two cells far apart in a camera view by its trajectory. Our probabilistic model is based on some simple, general assumptions on the spatial and temporal features related to activities: (1) cells located on the same path are likely to be connected by trajectories; (2) trajectories passing through the same path belong to the same activity; (3) it is likely for trajectories of the same object observed in different camera views to be on the same path in the real world and belong to the same activity; otherwise, objects might switch paths when crossing different camera views.

In our approach, a network is first built by connecting trajectories that are in different camera views and

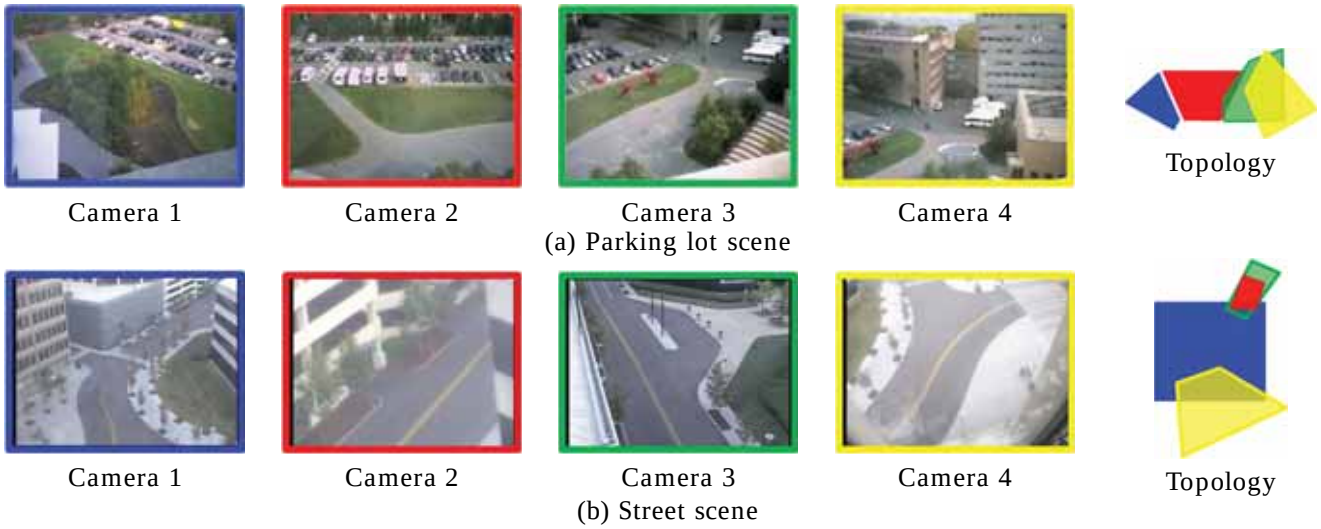


Fig. 2. Camera views and their topology in two data sets, a parking lot scene and a street scene. When the topology of camera views is plotted, the fields of camera views are represented by different colors: blue (camera 1), red (camera 2), green (camera 3), yellow (camera 4). However, our approach does not require knowledge of the topology of the cameras in advance.

whose temporal extents are close by edges. Then a probabilistic model, in which different kinds of activities have distributions in low-level feature spaces of different camera views, is built. A trajectory is treated as a set of observations that belong to different activities. A smoothness constraint requires that two neighboring trajectories connected by an edge have similar distributions over activities. Trajectories are clustered according to the assigned major activities among their observations. The distributions of activities over feature spaces of different camera views model the regions of paths across cameras.

1.1 Related Work

Many similarity-based trajectory clustering methods have been proposed in past years. The spatial proximity between a pair of trajectories is measured in different ways, such as Euclidean distance [24], Hausdorff distance [22] and its variations [17], hidden Markov model [23], and dynamic time warping [20]. A comparison of different similarity measures can be found in [25]. Based on the similarity matrix, some clustering algorithms such as spectral clustering and graph-cut were used to group trajectories in to different activity categories. The complexity in both time and space of these approaches is at least $O(M^2)$ where M is the number of trajectories. The complexity of labeling a new trajectory as one of the activity categories or an abnormality is $O(M)$, since similarity-based approaches required computing the similarity between the new trajectory and each of the trajectories in the training set. Visual surveillance systems often require processing data collected over weeks or even months. These approaches had difficulties with very large data sets. The spatial extents of paths related to activities can be estimated from trajectory clusters [28], [21], [17], [29]. All these approaches assumed that trajectories are observed in a single camera view. In order to extend these approaches to multiple camera views, trajectories observed in different camera views have to be stitched together.

Considerable work has been done to solve the challenging correspondence problem of trajectories observed in multiple camera views. One way is to manually label salient points in the scene and record their coordinates in the 3D world. After mapping 2D image planes to the 3D world [49], [50], objects can be tracked in multiple camera views. When the camera views overlap, static features can be selected to compute an assumed homography between two camera views [51] and calibrate camera views to a single global ground plane. Trajectories in different camera views can be stitched based on their spatial proximity on the common ground plane. In general, automatically finding correspondence of static features between different views is difficult.

Lee et al. [33], Sheikh and Shah [47], and Stauffer and Tieu [38] calibrated multiple camera views using tracking data from moving objects. They also assumed that camera views had significant overlap and that objects moved on the same ground plane. Lee et al. [33] and Sheikh and Shah [47] assumed that the topological arrangement of camera views was known. Stauffer and Tieu [38] could automatically infer it, but with high complexity ($O(N^2)$ where N is the number of camera views).

When the camera views are disjointed or their overlap is small, automatic calibration is difficult and the appearance of objects is often used as a cue to correspondence [37], [42], [41], [43], [45], [52]. This is a very challenging problem and not well solved yet. The appearance of objects may significantly change because of different cameras' settings and different poses of

objects. Many objects, such as cars or persons, have similar appearance, confusing correspondence. In far-field settings, objects may only cover a few pixels, making matching difficult. Other approaches [39], [44] inferred the topology of disjoint camera views using the transition time between cameras.

Even given similarities between trajectories observed in different camera views, solving the correspondence problem is still difficult because of the large search space, especially when there are many trajectories and cameras. It requires searching in the solution space of N-partite graphs, where N is the number of cameras [47]. In general, if there are more than two cameras, the problem is *NP* hard in the number of trajectories [53]. It has solution in polynomial time only with some particular topologies of camera views and the topology has to be known [37].

In summary, all these trajectory correspondence approaches had various assumptions on the topology and geometry of camera views, and they faced difficulties of camera calibrations, appearance matching, inference on the topology of camera views, and high computational cost to search for the optimal solution. The contributions of this paper are that we directly cluster trajectories into activities and model regions of paths over a multi-camera network without solving the correspondence problem. Given a general setting of a camera network, solving the correspondence problem is difficult. So our method has less restriction on the topology of camera views, the structures of the scene, and the number of cameras. Furthermore, in our approach each trajectory cluster is represented by a parametric probabilistic model. It does not require computing the similarity between each pair of trajectories. It has much lower space complexity compared with those similarity-based approaches. So it is more appropriate to process huge data sets often required in visual surveillance applications.

The paper is organized as following. Section 2 explains how to compute low-level features from trajectories and quantize them into visual words. In Section 3, a trajectory network is built by connecting trajectories which are in different camera views. In Section 4 introduces the models of activities (paths). In Section 5, our algorithms are evaluated on a simulated data set and two real data sets, a parking lot scene and a street scene, each of which has four cameras. The views and topology of these cameras are shown in Figure 2.

2 Feature Space

Objects are tracked in each of the camera views independently using the Stauffer-Grimson tracker [10]. A trajectory is treated as a set of observations. The locations and moving directions of observations of an object are computed as features and quantized to visual words according to a codebook of its camera view. In each camera view, the space of the view is uniformly quantized into small cells and the velocity of objects is quantized into several directions. A global codebook concatenates the codebooks of all the camera views. Thus the word value of an observation i is indexed by (c_i, x_i, y_i, d_i) in the global codebook. c_i is the camera view in which i is observed. (x_i, y_i) and d_i are the quantized coordinates and moving direction of observation i in camera c_i. The set of visual words on the trajectory are modeled as exchangeable (i.e., the distribution is invariant to a permutation of the observations). The moving direction encoded in the word value captures the first order temporal information among observations. Although quite simple, the position and velocity features can distinguish many different activity patterns especially in far-field settings.

3 Trajectory Network

A network is built connecting trajectories observed in multiple camera views based on their temporal extents. Each trajectory is a node on the network. Let t_{si} and t_{ei} be the starting and ending time of trajectory i. T is a positive temporal threshold. It is roughly the maximum transition time of objects crossing the gap between adjacent camera views. If trajectories a and b are observed in different camera views and their temporal extents are close,

$$(t_{sa} \leq t_{sb} \leq t_{ea} + T) \vee (t_{sb} \leq t_{sa} \leq t_{eb} + T), \qquad (1)$$

then a and b will be connected by an edge on the network. This means that a and b may be the same object since they are observed by cameras around the same time. There is no edge between two trajectories observed in the same camera view. An example can be found in Figure 3. As shown in (a), the views of cameras 1 and 2 overlap and are disjoint with the view of camera 3. Trajectories 1 and 2 observed by cameras 1 and 2 correspond to the same object moving across camera views. Their temporal extents overlap as shown in (b), so they are connected by an edge on the network as shown in (d). Trajectories 3 and 4 observed by cameras 1 and 3 correspond to an object crossing disjoint views. Their temporal extents have no overlap but the gap is smaller than T as shown in (c), so they are also connected. Trajectories 3 and 6, 5 and 7 do not correspond to the same objects, but their temporal extents are close, so they are also connected on the network. A single trajectory 3 can be connected to multiple trajectories (4 and 6) in other camera views. An edge on the network indicates a possible correspondence candidate only based on the temporal information of trajectories. But we do not really solve the correspondence problem when building the trajectory network, since many edges are actually false correspondences. The network simply keeps all of the possible candidates.

4 Probabilistic Model

In this section, we describe our probabilistic model which clusters trajectories in different camera views into

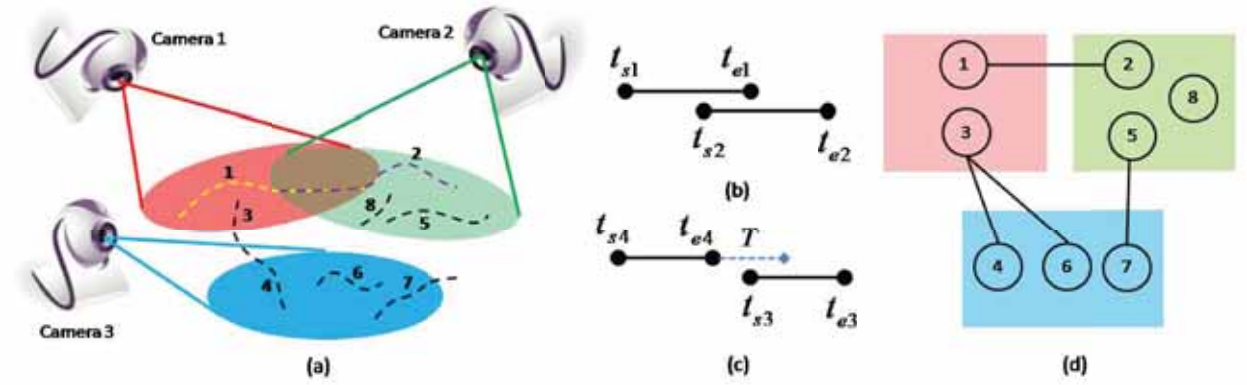


Fig. 3. An example of building a network connecting trajectories in multiple camera views. (a) Trajectories in three camera views. (b) The temporal extents of trajectories 1 and 2. (c) The temporal extents of trajectories 3 and 4. (d) The network connecting trajectories. See text for details.

activities and models paths across camera views. Our work is related to topic models, such as Probabilistic Latent Semantic Analysis (pLSA) [54] and Latent Dirichlet Allocation (LDA) [55], which were used for word-document analysis. These topic models assume that a document is a mixture of topics and cluster words, such as "professor" and "university", that often co-occur in the same documents into one topic, such as "education". In our domain, documents are trajectories, words are observations, and topics are activities (paths). Each activity has a distribution over locations and moving directions in different camera views, and corresponds to a path. If two word values, which are indices of locations and moving directions, often co-occur on the same trajectories, they are on the same path. Trajectories passing through the same path belong to the same activity. In previous topic models, documents are generated independently. However, we assume that if two trajectories in different camera views are connected by an edge in the network, which means that they may correspond to the same object since they are observed by cameras around the same time, they tend to have similar distributions over activities. Thus the distributions of an activity (a path of objects) in different camera views can be jointly modeled. In Figure 4, we use an example to describe the high level picture of our model. Trajectories a and b are observed in different camera views and connected by an edge on the network. Points on trajectories are assigned to activity categories by fitting activity models. Thus both a and b have distributions over activities. The smoothness constraint requires that their distributions over activities are similar in order to have small penalty. In this example, both trajectory a and b have a larger distribution on activity 1, so the models of activity 1 in two different camera views can be related to the same activity.

Let M be the number of trajectories. Each trajectory j has N_j observations. Each observation i on trajectory j has a visual word value w_{ji} which is an index of the global codebook. Observations will be clustered to one of the K activity categories. Let z_{ji} be the activity label of observation i on trajectory j. Each activity k has a multinomial distribution ϕ_k over the global codebook. So an activity is modeled as distributions over space and moving directions in multiple camera views. ϕ_k is sampled from a Dirichlet prior

$$p(\phi_k|\beta) = Dir(\phi_k;\beta), \tag{2}$$

where $Dir(\cdot;\cdot)$ is Dirichlet distribution and β is a flat hyperparameter. If a visual word w_{ji} has activity label z_{ji}, its data likelihood is

$$p(w_{ji}|z_{ji},\{\phi_k\}) = \phi_{z_{ji}w_{ji}}. \tag{3}$$

Eq 2 and 3 are the same as modeled in LDA.

Each trajectory has a random variable θ_j which is the parameter of a multinomial distribution over K activities. Activity labels $\{z_{ji}\}$ of observations are sampled from θ_j. If two trajectories j_1 and j_2 are connected by an edge on the network, they are neighbors and the smoothness constraint requires that θ_{j_1} and θ_{j_2} are similar and the distributions of $\{z_{j_1 i}\}$ and $\{z_{j_2 i}\}$ are similar. The joint distribution of $\{\theta_j\}$ and $\{z_{ji}\}$ are modeled as,

$$\begin{aligned}
&p(\{\theta_j\},\{z_{ji}\}|\alpha,\gamma)\\
&\propto \prod_{j=1}^{M}\prod_{k=1}^{K}(\theta_{jk})^{\alpha-1}\prod_{\{j_1,j_2\}\in E}\prod_{k=1}^{K}(\theta_{j_1 k})^{\gamma\cdot n_{j_2 k}}(\theta_{j_2 k})^{\gamma\cdot n_{j_1 k}}\\
&\prod_{j=1}^{M}\prod_{i=1}^{N_j}\theta_{jz_{ji}}\\
&=\prod_{j=1}^{M}\prod_{k=1}^{K}\theta_{jk}^{\alpha-1+\gamma\sum_{j'\in\Omega_j}n_{j'k}}\prod_{j=1}^{M}\prod_{i=1}^{N_j}\theta_{jz_{ji}}\\
&=\prod_{j=1}^{M}\Bigg[\frac{\prod_{k=1}^{K}\Gamma(\alpha+\gamma\sum_{j'\in\Omega_j}n_{j'k})}{\Gamma(K\cdot\alpha+\gamma\sum_{j'\in\Omega_j}\sum_{k=1}^{K}n_{j'k})}\\
&\mathrm{Dir}(\theta_j;\alpha+\gamma\sum_{j'\in\Omega_j}n_{j'1},\ldots,\alpha+\gamma\sum_{j'\in\Omega_j}n_{j'K})\prod_{i=1}^{N_j}\theta_{jz_{ji}}\Bigg]
\end{aligned} \tag{4}$$

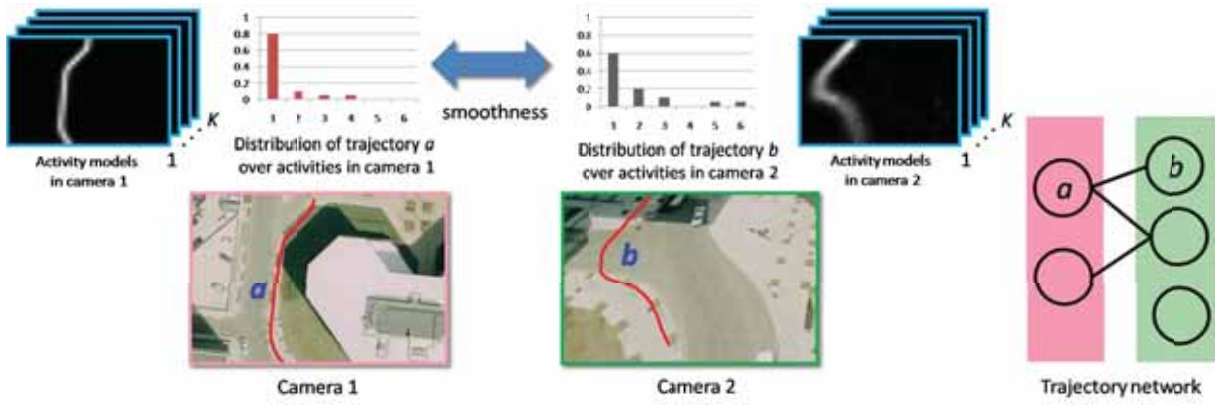


Fig. 4. An example to describe the high level picture of our model. See detail in the text.

$\Gamma(\cdot)$ is the Gamma function. n_{jk} is the number of observations assigned to activity k on trajectory j. E is the set of pairs of neighboring trajectories which are connected. Ω_j is the set of trajectories connected with j. α is a flat Dirichlet prior as a hyperparameter. $(\sum_{j'\in\Omega_j} n_{j'1}, \ldots, \sum_{j'\in\Omega_j} n_{j'K})$ is the histogram of observations assigned to K activity categories on the neighboring trajectories of j. It is used as the Dirichlet parameter for θ_j, after being weighted by a positive scalar γ and added to a flat prior α. Let $\rho_k = \alpha + \gamma \cdot \sum_{j'\in\Omega_j} n_{j'k}$. According to the properties of the Dirichlet distribution, if $\theta_j \sim \text{Dir}(\rho_1, \ldots, \rho_K)$, the expectation of θ_j is $(\rho_1/\sum\rho_k, \ldots, \rho_K/\sum\rho_k)$ and its variation is small if $\sum\rho_k$ is large. Notice that z_{ji} is sampled from θ_j and θ_j has a constraint added by $z_{j'i'}$ on its neighboring trajectories. So trajectory j tends to have a similar distribution over activities as its neighboring trajectories, which means that they are smooth. A larger γ puts a stronger smoothness constraint. If $\gamma = 0$, Eq 4 is the same as in LDA where $\{\theta_j\}$ are sampled from a Dirichlet prior $Dir(\cdot;\alpha)$ independently.

Given Eq 2, 3 and 4, finally the joint distribution of $\{\phi_k\}$, $\{\theta_j\}$, $\{z_{ji}\}$ and $\{w_{ji}\}$ is

$$
\begin{aligned}
&p(\{\phi_k\},\{\theta_j\},\{z_{ji}\},\{w_{ji}\}|\alpha,\beta,\gamma)\\
=&p(\{\theta_j\},\{z_{ji}\}|\alpha,\gamma)\prod_{k=1}^{K}p(\{\phi_k\}|\beta)\\
&\prod_{j=1}^{M}\prod_{i=1}^{N_j}p(\{w_{ji}\}|\{z_{ji}\},\{\phi_k\})\\
=&\prod_{j=1}^{M}\left[\frac{\prod_{k=1}^{K}\Gamma(\alpha+\gamma\sum_{j'\in\Omega_j}n_{j'k})}{\Gamma(K\cdot\alpha+\gamma\sum_{j'\in\Omega_j}\sum_{k=1}^{K}n_{j'k})}\right.\\
&\left.\text{Dir}(\theta_j;\alpha+\gamma\sum_{j'\in\Omega_j}n_{j'1},\ldots,\alpha+\gamma\sum_{j'\in\Omega_j}n_{j'K})\right]\\
&\prod_{k=1}^{K}\text{Dir}(\phi_k;\beta)\prod_{j=1}^{M}\prod_{i=1}^{N_j}\left(\theta_{jz_{ji}}\cdot\phi_{z_{ji}w_{ji}}\right).
\end{aligned}
\tag{5}
$$

4.1 Learning and Inference

Our goal is to estimate atcitivity labels $\{z_{ji}\}$ and activity models $\{\phi_k\}$. We do inference by Gibbs sampling. It turns out that $\{\theta_j\}$ and $\{\phi_k\}$ can be integrated out during the Gibbs sampling procedure.

$$
\begin{aligned}
&p(\{z_{ji}\},\{w_{ji}\}|\alpha,\beta,\gamma)\\
=&\int_{\{\phi_k\}}\int_{\{\theta_j\}}p(\{\theta_j\},\{\phi_k\},\{z_{ji}\},\{w_{ji}\}|\alpha,\beta,\gamma)d\{\theta_j\}d\{\phi_k\}\\
=&\int_{\{\phi_k\}}\prod_{k=1}^{K}p(\phi_k|\beta)\prod_{j=1}^{M}\prod_{i=1}^{N_j}p(\{w_{ji}\}|\{z_{ji}\},\{\phi_k\})d\{\phi_k\}\\
&\int_{\{\theta_j\}}p(\{\theta_j\},\{z_{ji}\}|\alpha,\gamma)d\{\theta_j\}\\
\propto&\int_{\{\phi_k\}}\prod_{k=1}^{K}\prod_{w=1}^{W}\phi_{kw}^{\beta-1}\prod_{j=1}^{M}\prod_{i=1}^{N_j}\phi_{z_{ji}w_{ji}}d\{\phi_k\}\\
&\int_{\{\theta_j\}}\prod_{j=1}^{M}\prod_{k=1}^{K}\theta_{jk}^{\alpha-1+\gamma\sum_{j'\in\Omega_j}n_{j'k}}\prod_{j=1}^{M}\prod_{i=1}^{N_j}\theta_{jz_{ji}}d\{\theta_j\}\\
=&\int_{\{\phi_k\}}\prod_{k=1}^{K}\prod_{w=1}^{W}(\phi_{kw})^{\beta+m_{kw}-1}d\{\phi_k\}\\
&\int_{\{\theta_j\}}\prod_{j}\prod_{k}(\theta_{jk})^{\alpha+n_{jk}+\gamma\cdot\sum_{j'\in\Omega_j}n_{j'k}-1}d\{\theta_j\}\\
=&\prod_{k}\frac{\prod_w\Gamma(\beta+m_{kw})}{\Gamma(W\cdot\beta+m_{k\cdot})}\\
&\prod_{j}\frac{\prod_k\Gamma\left(\alpha+n_{jk}+\gamma\cdot\sum_{j'\in\Omega_j}n_{j'k}\right)}{\Gamma\left(K\cdot\alpha+n_{j\cdot}+\gamma\cdot\sum_{j'\in\Omega_j}n_{j'\cdot}\right)},
\end{aligned}
\tag{6}
$$

where W is the size of the global codebook, m_{kw} is the number of observations assigned to activity k with value w, $m_{k\cdot}$ is the total number of observations assigned to activity k, n_{jk} is the number of observations assigned to activity k on trajectory j, and $n_{j\cdot}$ is the total number of observations on trajectory j. The conditional distribution

of z_{ji} given all the other activity labels $\mathbf{z}^{-ji}$ is

$$p(z_{ji}=k|\mathbf{z}^{-ji},\{w_{ji}\},\alpha,\beta,\gamma) \propto \frac{\beta+m^{-ji}_{k,w_{ji}}}{W\cdot\beta+m^{-ji}_{k,\cdot}}\cdot\frac{\alpha+n^{-ji}_{jk}+\gamma\sum_{j'\in\Omega_j}n_{j'k}}{K\cdot\alpha+n^{-ji}_{j\cdot}+\gamma\sum_{j'\in\Omega_j}n_{j'\cdot}}, \quad (7)$$

where $m^{-ji}_{kw_{ji}}$, $m^{-ji}_{k\cdot}$, n^{-ji}_{jk}, and $n^{-ji}_{j\cdot}$ are the same statistics as $m_{kw_{ji}}$, $m_{k\cdot}$, n_{jk}, and $n_{j\cdot}$ except that they have excluded observation i on trajectory j. To have a large posterior in Eq 7, the first term requires that the value of observation i should fit the model of activity k, and the second term requires that its activity label is consistent with those of observations on the same trajectory and neighboring trajectories, with γ controlling the weight of neighboring trajectories. The models of activities can be estimated from any single sample of $\{z_{ji}\}$,

$$\widehat{\phi}_{kw}=\frac{\beta+m_{kw}}{W\cdot\beta+m_{k\cdot}} \quad (8)$$

4.2 Labeling Trajectories into Activities

A trajectory is labeled as activity k, if most of its observations are assigned to k. The activity label of an observation can be obtained during the Gibbs sampling procedure based on Eq. 7. However, there may be an over smoothing effect, since in some cases most of the trajectories being the neighbors of trajectory j do not correspond to the same object as j. In this work, we adopt an alternative labeling approach which actually achieves better performance in experiments. As shown by the experimental results in Section 5, the activity models learnt from Gibbs sampling are distinctive enough to label trajectories. After the activity models have been learnt and fixed at the end of Gibbs sampling, which uses Eq. 7 and 8, we ignore the smoothness constraint among trajectories and label the observation as

$$z_{ji}=\arg\max_k \hat{\phi}_{kw_{ji}} \quad (9)$$

This is also used to label an unseen new trajectory.

4.3 Detection of Abnormal Trajectories

When detecting abnormal trajectories, we also ignore the smoothness constraint and fix the learnt activity models $\{\hat{\phi}_k\}$. A trajectory is detected as an abnormality if it does not fit any activity model well. Then abnormality detection is reduced to the Latent Dirichlet Allocation model proposed in [55]. The likelihood of a trajectory j under the learnt activity models $\{\hat{\phi}_k\}$ is

$$p(\mathbf{w}_j=\{w_{ji}\}|\alpha,\{\hat{\phi}_k\}) = \int_{\{\theta_j\}} p(\theta_j|\alpha)\left(\prod_{i=1}^{N_j}\sum_{z_{ji}}p(z_{ji}|\theta_j)p(w_{ji}|\hat{\phi}_{z_{ji}})\right), \quad (10)$$

where $p(\theta_j|\alpha)$ is a Dirichlet distribution, and both $p(z_{ji}|\theta_j)$ and $p(w_{ji}|\hat{\phi}_{z_{ji}})$ are discrete distributions. Since the computation of Eq. 10 is intractable, in [55] a variational approach was used to compute a lower bound of Eq. 10. A trajectory is flagged as abnormal if its lower bound is small.

4.4 Complexity

In order to simplify the notation, we assume that all the trajectories have the same number of observations, which is a fixed constant. The space complexity of our approach is $O(WK)+O(MK)$, while that of similarity based approaches is at least $O(M^2)$. The storage of similarity based approaches is unmanageable when M is huge. W is the size of the codebook, K is the number of activity categories, and M is the number of trajectories. In our approach, the time complexity of each Gibbs sampling iteration is $O(M)$, however it is difficult to provide theoretical analysis on the convergence of Gibbs sampling. Similarity based approaches have to compute the similarity of $O(M^2)$ pairs of trajectories and if spectral clustering is used, it is quite challenging to compute the eigenvectors of a huge $M\times M$ similarity matrix when M is large. The time complexity of our approach to label a new trajectory into one of the activity categories or detect a new trajectory as abnormal is $O(K)$[1], while the time complexity of similarity based approaches is at least $O(M)$. So our approach is much more efficient when the number of trajectories is huge.

5 Experimental Results

We evaluate our approach on two data sets, a parking lot scene and a street scene. Each has four camera views. Each camera view is in size of 320×240. To build the codebook, each camera view is quantized into 64×48 cells. Each cell is of size 5×5. The moving directions of moving pixels are quantized into four directions. There are tracking errors in both of the two data sets. For example, a track may break into fragments because of interactions among objects. In order to obtain more quantitative evaluation, we simulate some trajectories whose activity categories are known as the ground truth, and evaluate our approach on the simulated data.

5.1 Learning Activity Models

5.1.1 Parking Lot Scene

The parking lot data set has $22,951$ trajectories, collected from 10 hours during the day time over 3 days. Inspection shows that it is a fairly busy scene. The topology of its four camera views is shown in Figure 2 (a). The view of camera 1 has no overlap with other camera views. However, the gap between views of cameras 1 and 2 is small. The views of cameras 2 and 3 have small overlap. The views of cameras 3 and 4 have large overlap. Our approach does not require the knowledge of the topology of cameras. Fourteen different activities are learnt from this data set. Six of them are shown

1. In abnormality detection, a variational approach [55] is used to compute a lower bound of the data likelihood (Eq. 10) in an iterative process. We assume the number of iterations is small.

in Figure 5. For each activity, we plot its distribution over space and moving directions in the four camera views and the trajectories clustered into this activity. When visualizing activity models, moving directions are represented by different colors, and the density of distributions over space and moving directions is proportional to the brightness of colors (high brightness means high density). When plotting trajectories, random colors are used to distinguish individual trajectories.

In Figure 5, activity 1 captures vehicles and pedestrians entering the parking lot. It has a large extent in space and is observed by all four cameras. Activity 2 captures vehicles and pedestrians leaving the parking lot. In activities 3 and 5, pedestrians are walking in the same direction but on different paths. From the distributions of their models, it is observed that the two paths are side by side but well separated in space. The path of activity 4 occupies almost the same region as that of activity 3. However, pedestrians are moving in opposite directions in these two activities, so the distributions of their models are plotted in different colors. In activity 6, pedestrians appear from behind the trees and a building as observed by cameras 3 and 4 and disappear from a gate of the parking lot in the view of camera 2.

5.1.2 Street Scene

The topology of the four camera views of the street scene is shown in Figure 2 (b). Camera 1 has a distant view of the street. Camera 2 zooms in on the top-right part in the view of camera 1. The view of camera 3 has overlap with the views of cameras 1 and 2. It extends the top-right part of the view in camera 1 along the street. The view of camera 4 partially overlaps with the bottom region of the view in camera 1. There are $14,985$ trajectories in this data set, collected from 30 hours during day time in four days. Seventeen activities are learnt in this scene. Six of them are shown in Figure 6.

Activity 1 (Figure 6) captures vehicles moving on the road. It is observed by all of the four cameras. Vehicles first move from the top-right corner to the bottom-left corner in the view of camera 4. Then they enter the bottom region in the view of camera 1 and move upward. Some vehicles disappear at the exit points observed in the views of cameras 2 and 3, and some move further beyond the view of camera 3. In activities 3, 5 and 6, pedestrians first walk along the sidewalk in the view of camera 1, and then cross the street as observed by camera 4. The paths of activities 5 and 6 occupy similar regions in the view of camera 1, but their paths diverge in the view of camera 4. The paths of activities 2 and 3, 4 and 5 occupy the same regions but pedestrians are moving in opposite directions on them.

As shown in Figure 5 and 6, the models of activities reveal some structures, such as paths commonly taken by objects, and entrance and exit points in the scene. Some paths are less related to the appearance of the scene. For example, some paths cross the street outside the crosswalk in the street scene. Usually paths have spatial extents in multiple camera views. These regions can be detected by simply thresholding the density of the distributions of activities (ϕ_k in Eq 5). As observed, in these two very large data sets there are many outlier trajectories, which do not fit any activity model well, such as those crossing the grass fields in the parking lot scene. They are finally assigned some activity at random or because part of the trajectory fits a particular activity.

TABLE 1
Negative log likelihood under our approach and two alternative trajectory networks.

	Our approach	Unconnected	Random
Parking Lot	130.3	200.3	176.8
Street	85.7	228.8	135.2

TABLE 2
Negative log likelihood with models trained on a variable number of cameras. The test data is 200 trajectories from a single camera. The activity models in that camera are jointly learnt with different number of cameras (from 1 to 4). The last column is a baseline model trained on data whose cluster labels of trajectories are randomly assigned.

	1	2	3	4	Random
Parking Lot	120.9	121.3	122.8	123.3	425
Street	40.0	41.5	44.9	42.2	168

5.1.3 Negative log likelihood on testing data

Since clustering trajectories into activities is unsupervised learning, we compute the negative log likelihood on testing data to evaluate its performance. It is the log of perplexity proportion to the number of bits required to encode the testing data. It measures how unseen testing data fits the model learnt from training data. Two hundred trajectories randomly sampled from each camera serve as the test set; the remaining trajectories are used for training. To compare models with different trajectory networks, the activity models $\{\phi_k\}$ are learnt with smoothness constraint added by the trajectory network. Once $\{\phi_k\}$ are learnt and fixed, the negative log likelihood is computed on the test data ignoring the smoothness constraint.

First, we compare our approach with two alternatives: (1) unconnected network; (2) network with random correspondences[2]. The former completely abandons the smoothing constraint, so it cannot jointly model the distributions of a single activity in multiple camera views. The latter simulates the case when correspondence is poor. Both alternatives result in higher negative log likelihood as shown in Table 1.

2. First find correspondence candidates using Eq 1. Instead of fully connecting these candidates as in our model, a trajectory is randomly connected with only one of the candidates in a different camera view.

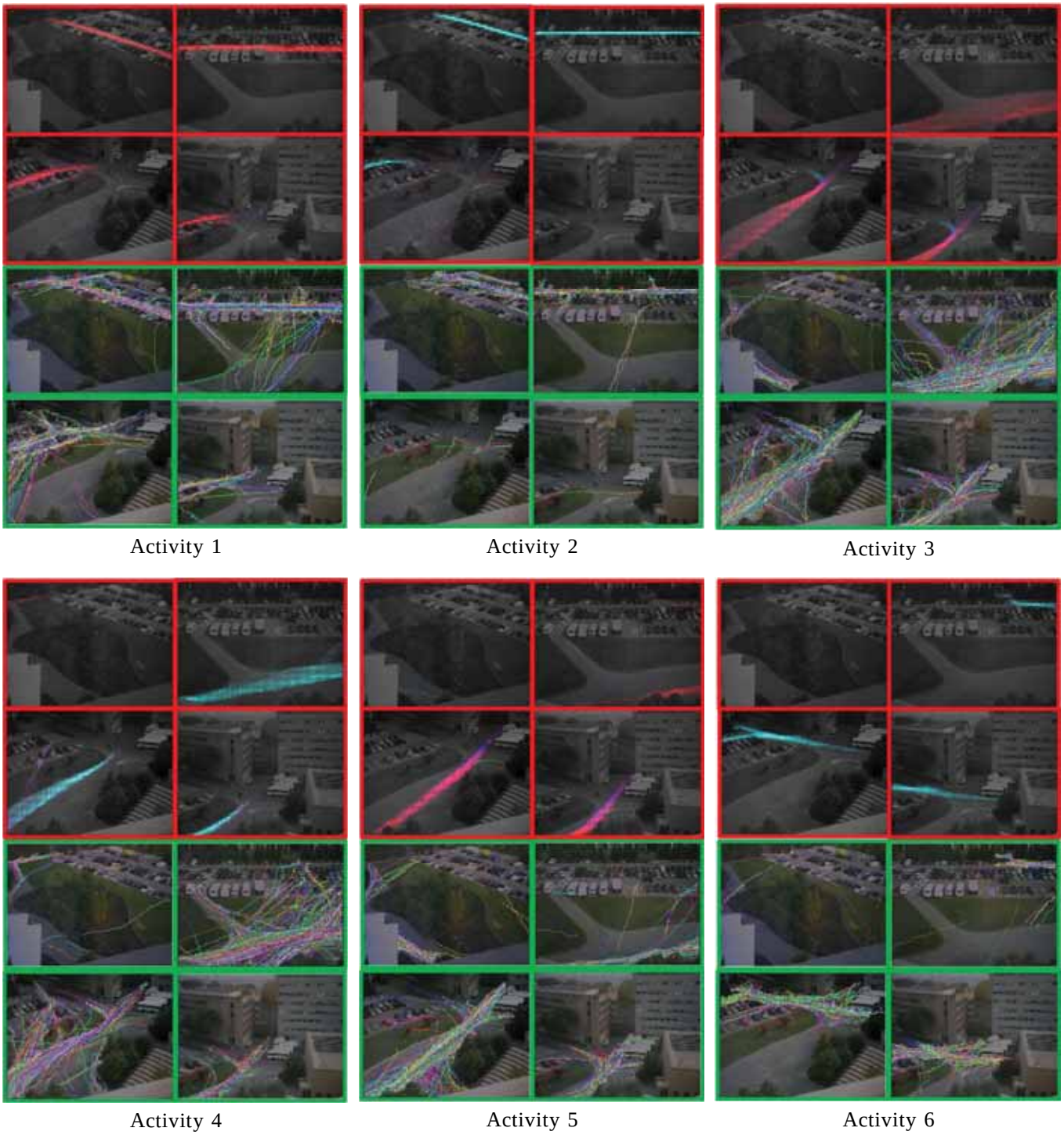


Fig. 5. Distributions of activity models ($1-6$) and clusters of trajectories of the parking lot scene. When plotting the distributions of activity models (in the four red windows on the top), different colors are used to represent different moving directions: $\rightarrow$ (red), $\leftarrow$ (cyan), $\uparrow$ (blue), $\downarrow$ (magenta). When plotting trajectories clustered into different activities (in the four green windows at the bottom), random colors are used to distinguish individual trajectories.

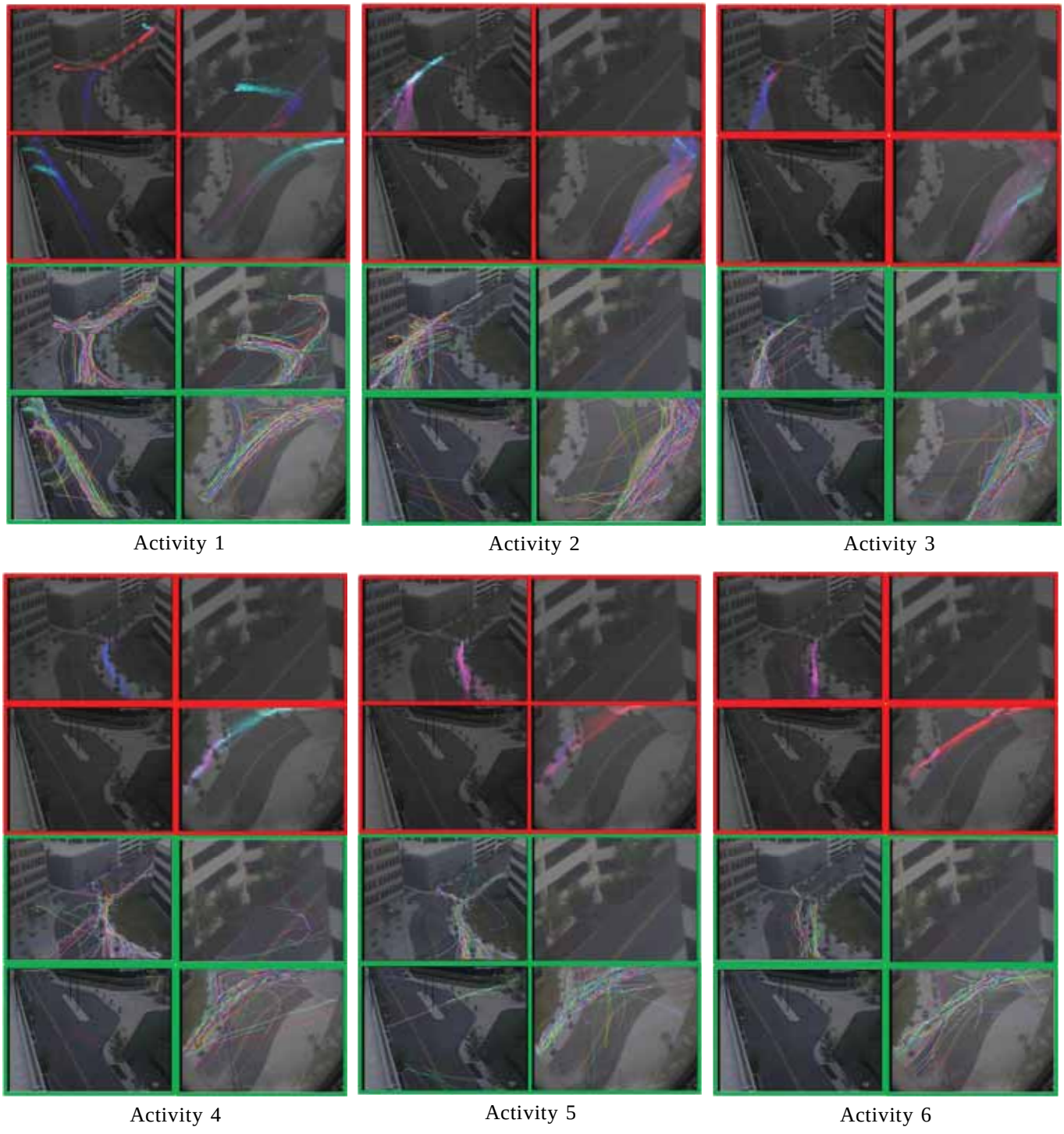


Fig. 6. Distributions of activity models ($1-6$) and clusters of trajectories of the street scene. The meaning of colors is the same as Figure 5.

Fig. 7. Activity models learnt in an unsupervised way help to solve the correspondence problem.

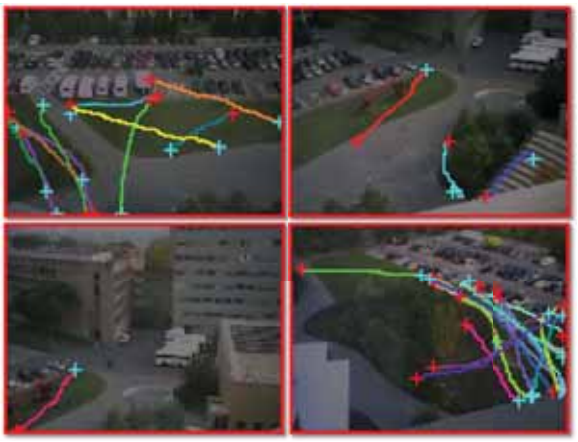


Fig. 8. Some trajectories with low likelihoods from the parking lot scene. Random colors are used to distinguish individual trajectories. In order to indicate the moving direction of a trajectory, the starting and ending points of a trajectory is marked by $+$ in red and cyan colors.

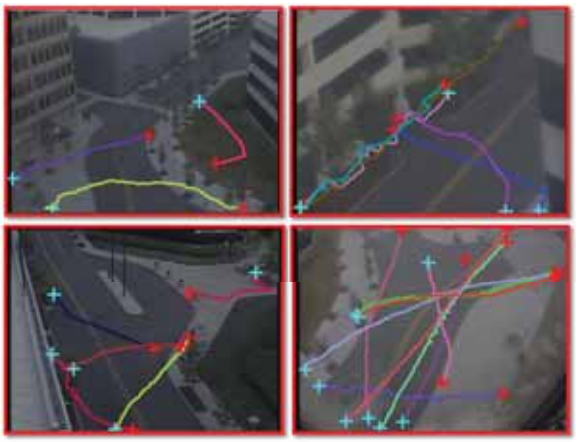


Fig. 9. Some trajectories with low likelihoods from the street scene. Random colors are used to distinguish individual trajectories. In order to indicate the moving direction of a trajectory, the starting and ending points of a trajectory is marked by $+$ in red and cyan colors.

We also compare against models learned with trajectories from a single to all of the camera views. Models learned from a subset of the cameras will necessarily have lower negative log likelihood for trajectories within those camera views; however, they are limited to modeling joint activities only in a subset of the camera views. Our model captures joint activities in all cameras simultaneously, and only exhibits a small increase in the negative log likelihood as shown in Table 2.

5.1.4 Temporal Threshold

The temporal threshold T in Eq 1 determines the connectivity on the trajectory network. If a camera view A is disjoint from other views and it takes objects more than T seconds to cross the smallest gap between A and other views, then there is no way to extend a path in A to other views. If T is large and the scene is busy, the network will have too many "noisy" edges which connect two trajectories actually corresponding to different objects. Under-smoothing could lead to the same activity separated into different clusters, while over-smoothing could lead to different activities joining into the same cluster. Empirically, we achieved similar results with a wide range of values for T: for the street scene data set, good results are achieved when T varies between 0 and 30 seconds; for the parking lot data set, the range of good values of T is roughly from 3 to 15 seconds because the parking lot scene is busier and the view of camera 1 is disjoint from other camera views. There is quantitative evaluation of T on a simulated data set in Section 5.5.

5.2 Correspondence

Although our activity analysis approach does not require correspondence among trajectories in different camera views, after the models of activities have been learnt in an unsupervised way, they can help to solve the correspondence problem, since if two trajectories belong to the same activity and are connected by an edge, they are likely to correspond to the same object. For example, see Figure 7. We pick a query trajectory from one of the camera views and mark it using green color and a star. All the trajectories in other camera views satisfying Eq 1 are plotted in random colors. The red color and red stars mark the trajectories with the same activity category as the query trajectory. They are likely to correspond to the same object. So the information on activity category can dramatically reduce the search space when solving the correspondence problem. There is a quantitative evaluation in Section 5.5.2.

5.3 Abnormality Detection

In Figure 8 and 9 we plot some trajectories with low data likelihoods, which have been normalized by the length of trajectories, and are detected as abnormality

from the parking lot scene and the street scene. All of the trajectories are sorted by abnormality and the top 30 are shown. Some very short trajectories most likely caused by tracking errors are not shown here. In the parking lot scene, most of the detected abnormal trajectories are pedestrians walking on the grass field. In the street scene, abnormal activities include pedestrians walking on the grass fields, pedestrians crossing the street, pedestrians walking in the middle of the street, and vehicles moving along a wrong lane.

5.4 Computational Cost

Running on a computer with 2GHz CPU, it takes about two hours to learn the activity models from $22,951$ trajectories of the parking lot data set and 40 minutes to learn the activity models from $14,985$ trajectories from the street scene. When the activity models are learnt and fixed, it takes less than 0.03 second to compute the likelihood of a trajectory in order to detect abnormality, and it is much faster to label a new trajectory as some activity category.

5.5 Simulated Data

In order to quantitatively evaluate our algorithm, we simulate data used as the ground truth. As shown in Figure 1 (c), we choose a scene which covers almost the same area of the street scene we used in Section 5.1.2. On a satellite image, we manually draw the fields covered by four camera views. The fields are convex four-sided polygons. These fields are converted to a standard camera view in size of 240×360 through projective transformation. The views observed by four cameras are shown in Figure 1 (b). We manually draw the central lines of eight paths on the satellite image (Figure 10 (a)) and simulate 8000 trajectories. We assume trajectories have almost the same speed, since speed does not play an important role in our algorithm. The starting points of trajectories are generated sequentially as follows.

$$t_{s(i+1)} = t_{si} + \Delta t_{i+1}, \tag{11}$$

$$\Delta t_{i+1} \sim Exponential(\lambda). \tag{12}$$

t_{si} is the starting time of the ith trajectory. The temporal difference $\Delta t_{i+1} = t_{s(i+1)} - t_{si}$ between two successive trajectories is sampled from a exponential distribution with mean λ. A trajectory i is randomly assigned to one of the eight predefined activities, k. Trajectory i samples the location of its starting point from a Gaussian distribution centered at the starting point of path k with variance σ_1 ($\sigma_1 = 5$ in this simulation). Then i samples the remaining points sequentially with the velocity specified by path k and being added to Gaussian noise with variance σ_2 ($\sigma = 2$ in this simulation). The simulated trajectories in the global views and each of the four camera views are shown in Figure 10 (b) and (c).

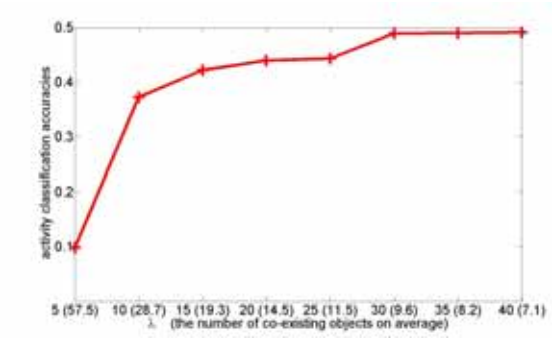
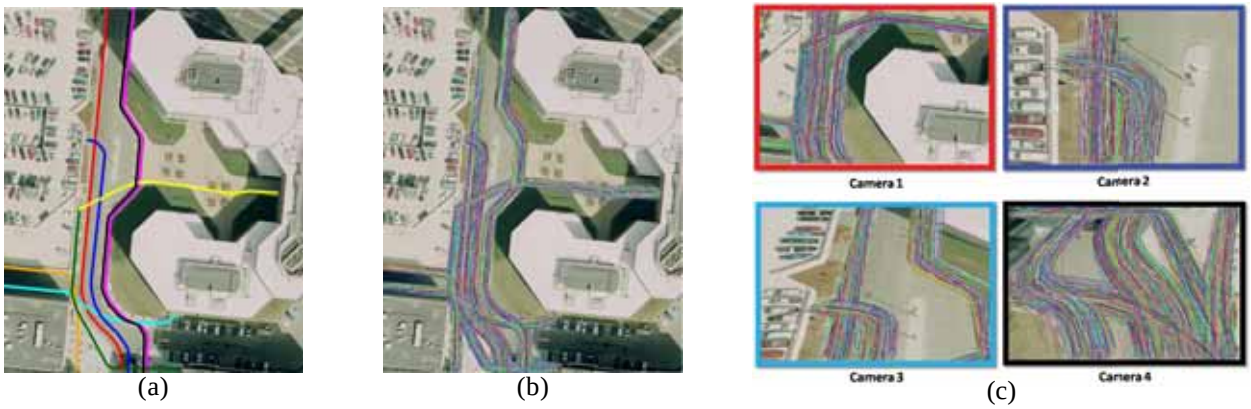


Fig. 11. The accuracies of classifying trajectories into different activities when λ takes different values and T is fixed as 0.

5.5.1 Learning activity models

λ is the parameter controlling how busy the scene is. When λ is smaller, more objects co-exist in the scene at the same time, which means that there are more edges on the trajectory network and it is harder for our algorithm to jointly learn the models of activities in different camera views. In our experiments, we change the value of λ from 5 seconds to 40 seconds. Based on the speed set for this experiment, the time an object spent to pass through a path varies from 170 seconds to 410 seconds. It depends on the length of the path. When λ takes values from 5 seconds to 40 seconds, the averaged number of objects co-existing in the scene varies from 57.5 to 7.1 (see Figure 11). After the trajectories are clustered by our algorithm, we manually specify each of the eight clusters as an activity category, so each trajectory is assigned an activity label by our algorithm. By comparing with the ground truth, the accuracy of activity classification is computed. The accuracies when choosing different λ values are shown Figure 11. The accuracy is high ($> 97.8\%$) when $\lambda \geq 30$ seconds. The models of activities in a single global view and four camera views learnt from the simulated data when $\lambda = 30$ seconds are shown in Figure 12 and Figure 13. Notice that when $\lambda = 30$ seconds, if we randomly sample a time point, there are around 9.6 objects co-existing in the scene on average. Each trajectory is connected to 12.4 trajectories by edges on the network on average. When λ decreases, some trajectories of different activities merge into one cluster. When $\lambda = 5$ seconds, the scene is very busy (there are 57.5 objects co-existing in the scene on average), all of the trajectories are merged into one cluster and our algorithm cannot learn any useful activity models from this data set. Each trajectory is connected to 73.0 trajectories by edges on the network on average.

We further look into the structure of the trajectory network constructed according to the temporal extents of trajectories in the data set when $\lambda = 30$ seconds. Figure 14 shows the number of edges which are related to different combinations of activities and camera views according to the ground truth. The entry of (k_1, k_2) on the table of camera views i_1 and i_2 are the number



(a) (b) (c)

Fig. 10. (a) The central lines of eight paths manually drawn in the scene. They are distinguished by colors: 1 (red), 2 (blue), 3 (dark green), 4 (magenta), 5 (black), 6 (cyan), 7 (yellow), and 8 (orange). (b) Trajectories generated from the eight paths. (c) Trajectories observed in four cameras.

of edges connecting two trajectories, one of which is in camera view i_1 and belongs to activity k_1, and the other of which is in camera view i_2 and belongs to activity k_2. There are six 8×8 tables. As we mentioned earlier, the edges on the trajectory network indicate possible correspondence candidates based on the temporal extents of trajectories. If the correspondence can be solved just using temporal information, all of the nonzero numbers in the table will be on diagonal. Actually many off diagonal entries have nonzero numbers, which indicate false correspondences, whose ambiguity cannot be solved by only using temporal information. The ratio between the numbers of edges on diagonal and off diagonal is 0.2732. This ratio can be understood as signal-to-noise ratio in some sense. There are many more false correspondences than true correspondences. However, these false correspondences almost uniformly distribute among different combinations of activities and work as background noise. So if a trajectory of activity k_1 is connected with another trajectory of activity k_2, k_2 is more likely to be the same as k_1 than any one of the other activties. When the scene is busier, the signal-to-noise ratio is lower. When $\lambda = 5$, the ratio is 0.1692 and our algorithm fails. Notice that the signal-to-noise ratio is $1/7 = 0.1429$, if trajectories are randomly connected without using any temporal information.

Figure 15 plots the classification accuracies when λ is fixed as 40 seconds and the temporal threshold T in Eq. 1 changes from 0 seconds to 300 seconds. The results stay at a high accuracy when T varies in a large range between 0 second and 40 seconds. There is some interesting correlation between Figure 11 and Figure 15. The performance of our algorithm drops if there are too many edges on the trajectory network, which means that the "signal-to-noise" ratio is low. The number of edges increases if λ decreases, which means that the scene is busier and there are more objects co-existing in the scene, or T increases. From Figure 11, when T is fixed at 0 second, $\lambda = 30$ seconds seems to be a turning point on the accuracy curve. On average, there are around

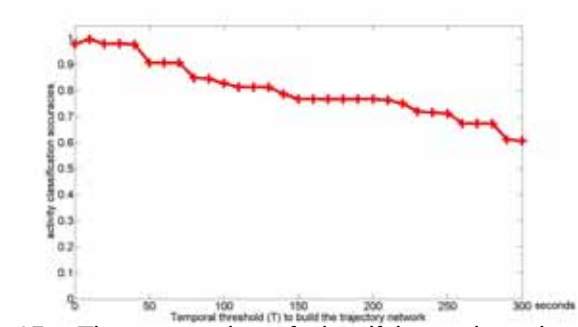


Fig. 15. The accuracies of classifying trajectories into different activities when the temporal threshold T change from 0 to 300 seconds. Here, $\lambda = 40$ seconds.

two more objects co-occurring when $\lambda = 30$ seconds compared with $\lambda = 40$ seconds. From Figure 15, when λ is fixed at 40 seconds, $T = 40$ seconds seems to be a turning point on the accuracy curve. Compared with $T = 0$ second, the temporal window in Eq. 1 extends for $2 \times T = 80$ seconds. In 80 seconds, there are around two more objects appearing on average when $\lambda = 40$ seconds. So there are approximately the same number of edges under two settings. For $(\lambda = 30, T = 0)$, on average each trajectory is connected to 12.4 trajectories by edges on the network, and for $(\lambda = 40, T = 40)$ this number is 13.0.

5.5.2 *Using activity models to solve the correspondence problem*

As mentioned in Section 5.2, the learnt activity models can help to solve the correspondence problem. We evaluate the performance on simulated data. When there are more than two cameras views, the correspondence problem is NP hard in the number of trajectories. Finding an approximate solution to this NP hard problem is not the focus of this paper. So we demonstrate the capability of our activity models by doing correspondence among trajectories in two camera views. Given the distances between trajectories, correspondence of trajectories in

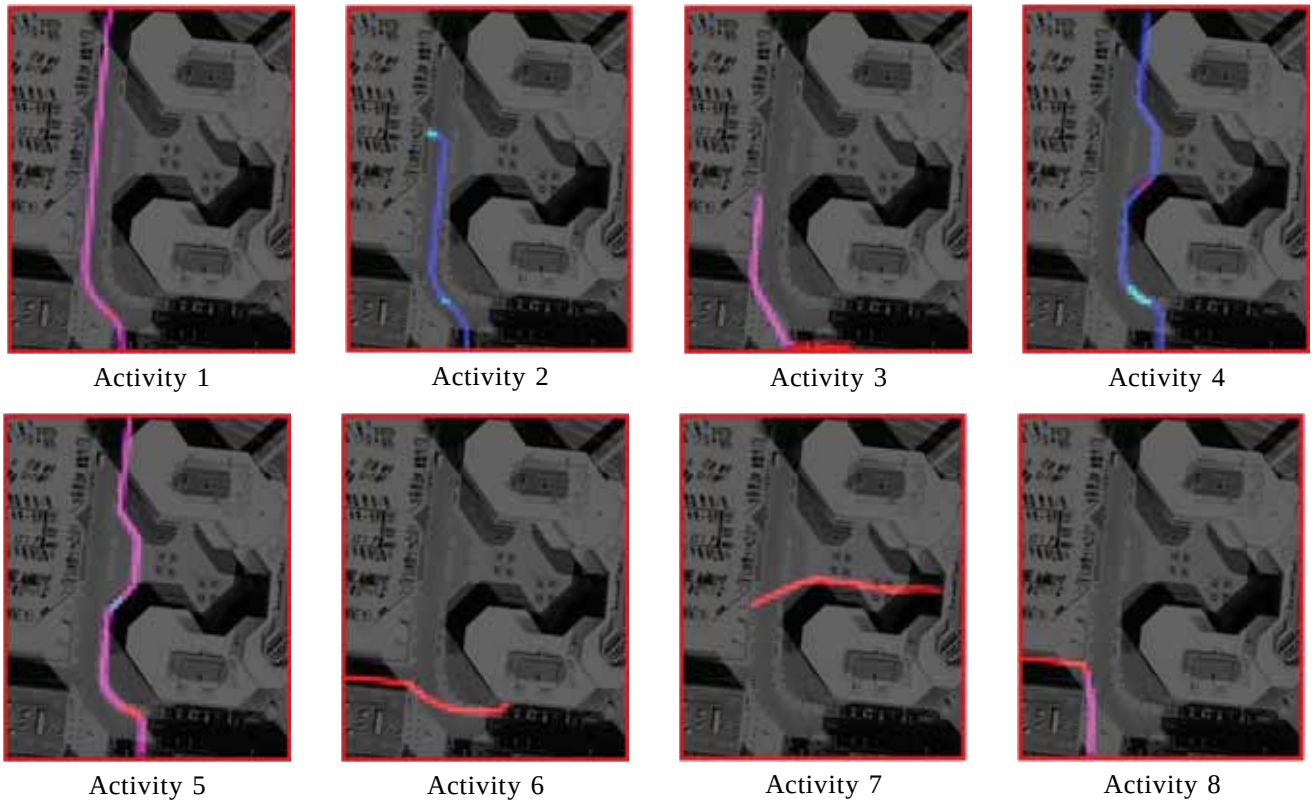


Fig. 12. Distributions of activity models in a single global views learnt from the simulated data. The meaning of colors is the same as Figure 5.

two cameras views can be solved by the Hungarian algorithm [56] in polynomial time. The distance $D(a,b)$ between two trajectories a and b which are in different views are define as follows. Each point on trajectories is assigned to one of the activities according to Eq. 9. Thus each trajectory j has a distribution p_j over activities. If trajectories a and b satisfy the temporal constraint Eq 1,

$$D(a,b) = \sum_{k=1}^{K} p_a(k) log\left(\frac{p_a(k)}{p_b(k)}\right) + \sum_{k=1}^{K} p_b(k) log\left(\frac{p_b(k)}{p_a(k)}\right), \quad (13)$$

which is Jensen-Shannon divergence; otherwise $D(a,b) = \infty$.

We choose camera views 1 and 4 which are shown in Figure 1. 1000 trajectories are simulated and they are not in the data set of 8000 trajectories used to learn the activity models. 1000 trajectories are observed in camera view 1 and around 880 trajectories are observed in camera view 4. Some trajectories of activity 7 (see Figure 10) observed in camera view 1 have no corresponding trajectories in camera view 4. We simulate different sets of data by changing the parameter λ. The accuracies of correspondence are plotted in Figure 16. It achieves very good correspondence accuracy (higher than 97%) when $\lambda \geq 30$. The accuracy drops when the scene is busier because of two reasons: (1) the activity models are not well learnt; (2) some objects of the same activity exist around the same time so they cannot be distinguished by activity categories and temporal extents.

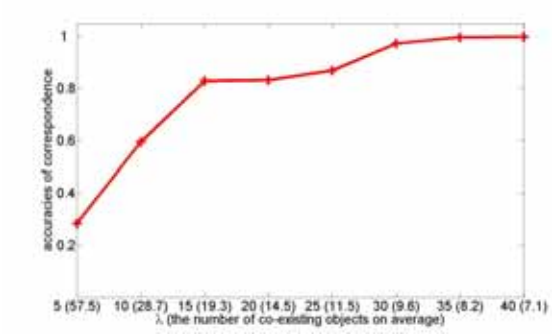


Fig. 16. Accuracies of correspondence. Solve the correspondence problem of trajectories observed in the views of camera 1 and camera 4. λ varies from 5 seconds to 40 seconds.

6 Discussion

The performance of our algorithm depends on the number of edges on the trajectory network. If on average a trajectory is connected to a large number of other trajectories, which means that there are many false correspondence candidates, the models of activities cannot be well learnt. The number of edges increases because of two reasons: the scene is busy or the temporal threshold T is large. A large T allows a large transition gap between camera views. So if a scene is busy, the transition gaps between cameras has to be small, which limits the topology of camera views in some sense. In this

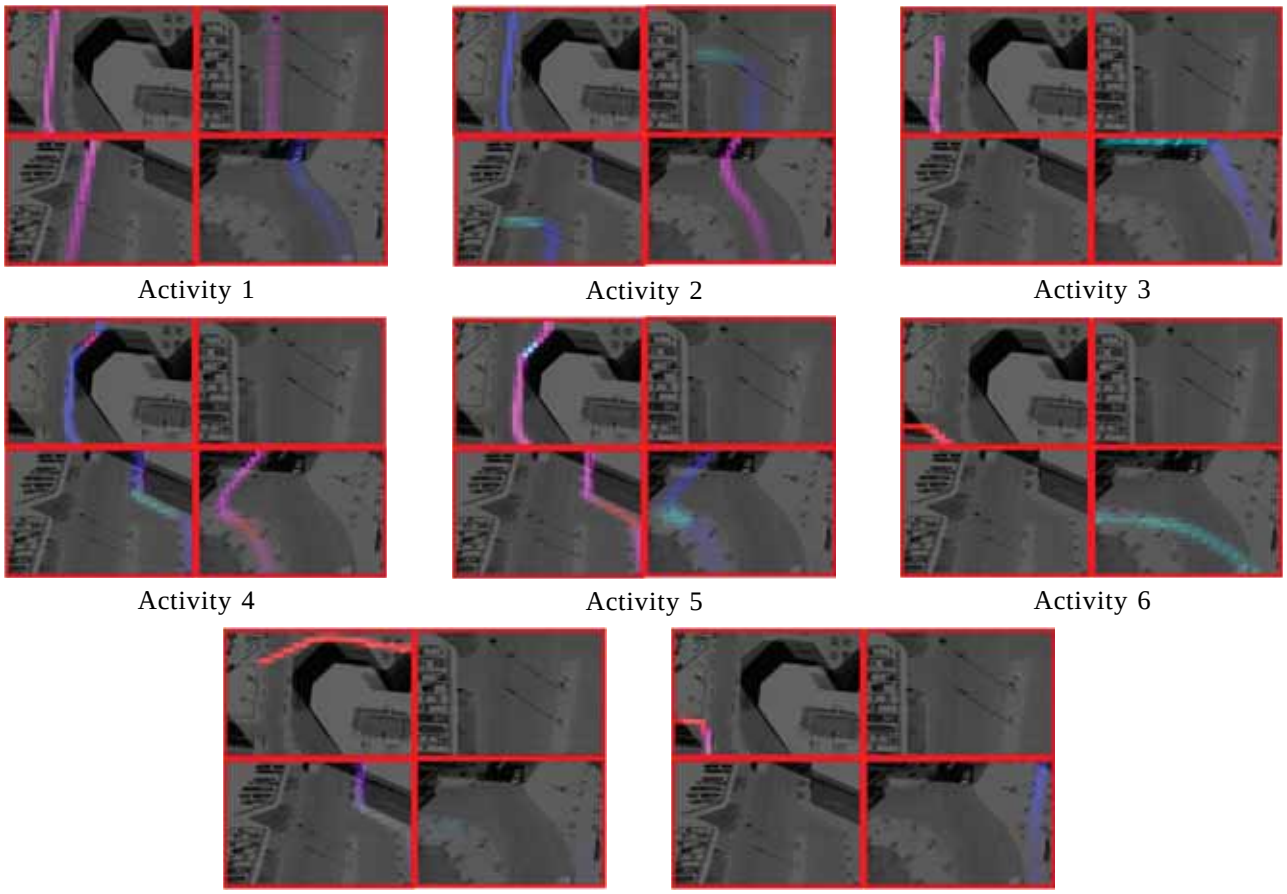


Fig. 13. Distribution of activity models in four camera views learnt from the simulated data. The meaning of colors is the same as Figure 5.

work, only temporal information is used to build the trajectory network. That is why the algorithm is sensitive to how busy the scene is. Some other features, such as appearance, can also be used to eliminate some edges. If two objects observed in different camera views are poorly matched by appearance, their trajectories are not connected by an edge even though their temporal extents are close. Thus activity models may be well learnt even in a busy scene. However, in this case the problem of matching appearance across camera views has to be addressed. It is a direction of our future study.

In this work, only positions and moving directions are computed as features of trajectories, since in our application they are enough to model paths of objects. Some other features such as size of objects and speed can also be added into this framework. They can be used to cluster motion patterns into more categories, such as separating vehicles and pedestrians moving on the same path. However, the size of the codebook will increase quickly as more features are included and thus the computational cost will increase.

In our clustering method, the number of clusters K has to be manually chosen. Some nonparametric models such as Hierarchical Dirichlet Processes [57], [8] can learn the number of clusters from data. They could be used to improve our clustering method in the future work.

7 Conclusion

We propose a framework to model activities and cluster trajectories over a multi-camera network. The models of activities can be used to detect scene structures. It is unsupervised and does not require first solving the challenging multi-camera correspondence problem. When the activity models have been learnt without supervision, it can help to solve the correspondence problem. Experiments on a simulated data set and two data sets including a very large number of trajectories demonstrate the effectiveness of this approach.

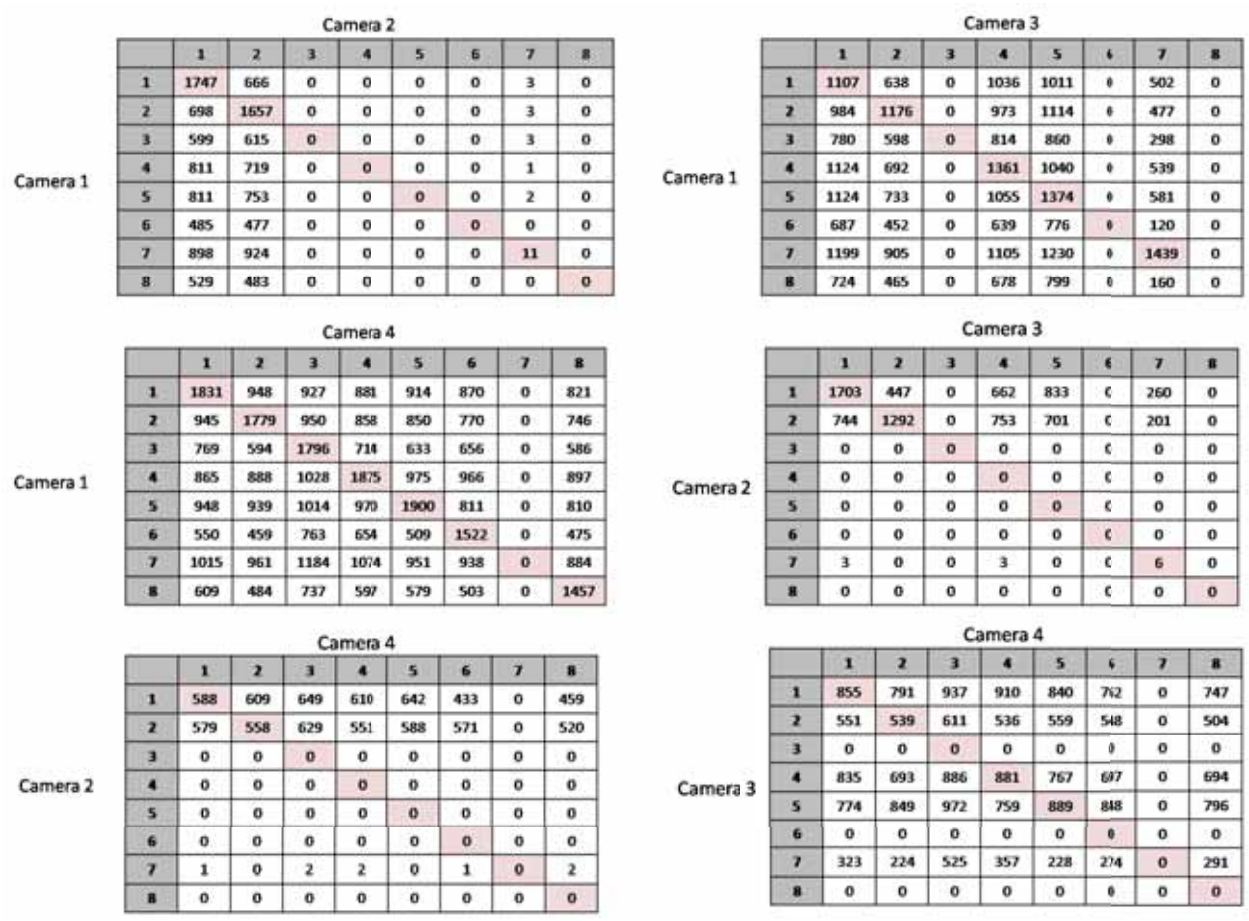
References

[1] J. W. Davis and A. F. Bobick. The representation and recognition of action using temporal templates. In *Proc. of IEEE Int'l Conf. Computer Vision and Pattern Recognition*, 1997.

[2] L. Zelnik-Manor and M. Irani. Event-based analysis of video. In *Proc. of IEEE Int'l Conf. Computer Vision and Pattern Recognition*, 2001.

[3] H. Zhong, J. Shi, and M. Visontai. Detecting unusual activity in video. In *Proc. of IEEE Int'l Conf. Computer Vision and Pattern Recognition*, 2004.

[4] T. Xiang and S. Gong. Video behaviour profiling and abnormality detection without manual labelling. In *Proc. of IEEE Int'l Conf. Computer Vision*, 2005.



Camera 1 / Camera 2

	1	2	3	4	5	6	7	8
1	1747	666	0	0	0	0	3	0
2	698	1657	0	0	0	0	3	0
3	599	615	0	0	0	0	3	0
4	811	719	0	0	0	0	1	0
5	811	753	0	0	0	0	2	0
6	485	477	0	0	0	0	0	0
7	898	924	0	0	0	0	11	0
8	529	483	0	0	0	0	0	0

Camera 1 / Camera 3

	1	2	3	4	5	6	7	8
1	1107	638	0	1036	1011	0	502	0
2	984	1176	0	973	1114	0	477	0
3	780	598	0	814	860	0	298	0
4	1124	692	0	1361	1040	0	539	0
5	1124	733	0	1055	1374	0	581	0
6	687	452	0	639	776	0	120	0
7	1199	905	0	1105	1230	0	1439	0
8	724	465	0	678	799	0	160	0

Camera 1 / Camera 4

	1	2	3	4	5	6	7	8
1	1831	948	927	881	914	870	0	821
2	945	1779	950	858	850	770	0	746
3	769	594	1796	714	633	656	0	586
4	865	888	1028	1875	975	966	0	897
5	948	939	1014	970	1900	811	0	810
6	550	459	763	654	509	1522	0	475
7	1015	961	1184	1074	951	938	0	884
8	609	484	737	597	579	503	0	1457

Camera 2 / Camera 3

	1	2	3	4	5	6	7	8
1	1703	447	0	662	833	0	260	0
2	744	1292	0	753	701	0	201	0
3	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0
7	3	0	0	3	0	0	6	0
8	0	0	0	0	0	0	0	0

Camera 2 / Camera 4

	1	2	3	4	5	6	7	8
1	588	609	649	610	642	433	0	459
2	579	558	629	551	588	571	0	520
3	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0
7	1	0	2	2	0	1	0	2
8	0	0	0	0	0	0	0	0

Camera 3 / Camera 4

	1	2	3	4	5	6	7	8
1	855	791	937	910	840	762	0	747
2	551	539	611	536	559	548	0	504
3	0	0	0	0	0	0	0	0
4	835	693	886	881	767	697	0	694
5	774	849	972	759	889	848	0	796
6	0	0	0	0	0	0	0	0
7	323	224	525	357	228	274	0	291
8	0	0	0	0	0	0	0	0

Fig. 14. The number of pairs of simulated trajectories, which are in different camera views, belong to activities i and j ($i, j = 1, \ldots, 8$), and whose temporal extents are close. Here, $\lambda = 6$ and $T = 0$.

[5] P. Smith, N. Lobo, and M. Shah. Temporalboost for event recognition. In *Proc. of IEEE Int'l Conf. Computer Vision*, 2005.

[6] Y. Wang, T. Jiang, M. S. Drew, Z. Li, and G. Mori. Unsupervised discovery of action classes. In *Proc. of IEEE Int'l Conf. Computer Vision and Pattern Recognition*, 2006.

[7] X. Wang, X. Ma, and E. Grimson. Unsupervised activity perception by hierarchical bayesian models. In *Proc. of IEEE Int'l Conf. Computer Vision and Pattern Recognition*, 2007.

[8] X. Wang, X. Ma, and W. E. L. Grimson. Unsupervised activity perception in crowded and complicated scenes using hierarchical bayesian models. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2008.

[9] N. Johnson and D. Hogg. Learning the distribution of object trajectories for event recognition. In *Proc. of British Machine Vision Conf.*, 1995.

[10] C. Stauffer and W. E. L. Grimson. Learning patterns of activity using real-time tracking. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2000.

[11] N. Oliver, B. Rosario, and A. Pentland. A bayesian computer vision system for modeling human interactions. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22:831–843, 2000.

[12] I. Haritaoglu, D. Harwood, and L. S. Davis. W4: Real-time surveillance of people and their activities. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22:809–830, 2000.

[13] M. Brand and V. Kettnaker. Discovery and segmentation of activities in video. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22:844–851, 2000.

[14] G. Medioni, I. Cohen, F. BreAmond, S. Hongeng, and R. Nevatia. Event detection and analysis from video streams. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 23:873–889, 2001.

[15] S. Honggeng and R. Nevatia. Multi-agent event recognition. In *Proc. of IEEE Int'l Conf. Computer Vision*, 2001.

[16] T. T. Truyen, D. Q. Phung, H. H. Bui, and S. Venkatesh. Adaboost.mrf: Boosted markov random forests and application to multilevel activity recognition. In *Proc. of IEEE Int'l Conf. Computer Vision and Pattern Recognition*, 2006.

[17] X. Wang, K. Tieu, and W. E. L. Grimson. Learning semantic scene models by trajectory analysis. In *Proc. of European Conf. Computer Vision*, 2006.

[18] T. Xiang and S. Gong. Beyond tracking: Modelling activity and understanding behaviour. *International Journal of Computer Vision*, 67:21–51, 2006.

[19] W. Hu, X. Xiao, Z. Fu, D. Xie, T. Tan, and S. Maybank. A system for learning statistical motion patterns. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 28:1450–1464, 2006.

[20] E. Keogh and M. Pazzani. Scaling up dynamic time. In *Proc. of SIGKDD*, 2000.

[21] D. Makris and T. Ellis. Path detection in video surveillance. *Image Vision and Computation*, 20:859–903, 2002.

[22] I. Junejo, O. Javed, and M. Shah. Multi feature path modeling for video surveillance. In *Proc. of IEEE Int'l Conf. Pattern Recognition*, 2004.

[23] F. M. Porikli and T. Haga. Event detection by eigenvector decomposition using object and frame features. In *Proc. of IEEE Conf. Computer Vision and Pattern Recognition Workshop*, 2004.

[24] Z. Fu, W. Hu, and T. Tan. Similarity based vehicle trajectory clustering and anomaly detection. In *Proc. of IEEE Int'l Conf. Image Processing*, 2005.

[25] Z. Zhang, K. Huang, and T. Tan. Comparison of similarity measures for trajectory clustering in outdoor surveillance scenes. In *Proc. of IEEE Int'l Conf. Pattern Recognition*, 2006.

[26] R. Kaucic, A. Perera, G. Brooksby, J. Kaufhold, and A. Hoogs. A unified framework for tracking through occlusions and across

sensor gaps. In *Proc. of IEEE Int'l Conf. Computer Vision and Pattern Recognition*, 2005.
[27] D. Makris and T. Ellis. Automatic learning of an activity-based semantic scene model. In *Proc. of IEEE Conf. on Advanced Video and Signal Based Surveillance*, 2003.
[28] J. Fernyhough, A. Cohn, and D. Hogg. Generation of semantic regions from image sequences. In *Proc. of European Conf. Computer Vision*, 1996.
[29] I. Junejo and H. Foroosh. Trajectory rectification and path modeling for video surveillance. In *Proc. of IEEE Int'l Conf. Computer Vision*, 2007.
[30] T. Huang and S. Russell. Object identification in a bayesian context. In *Proc. of Int'l Joint Conf. Artificial Intelligence*, 1997.
[31] Q. Cai and J. K. Aggarwal. Tracking human motion in structured environments using a distributed-camera system. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 21:1241–1247, 1996.
[32] V. Kettnaker and R. Zabih. Bayesian multi-camera surveillance. In *Proc. of IEEE Int'l Conf. Computer Vision and Pattern Recognition*, 1999.
[33] L. Lee, R. Romano, and G. Stein. Monitoring activities from multiple video streams: Establishing a common coordinate frame. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22:758–768, 2000.
[34] R. T. Collins, A. J. Lipton, H. Fujiyoshi, and T. Kanade. Algorithms for cooperative multisensor surveillance. *Proceedings of IEEE*, 89:1456–1477, 2001.
[35] S. Khan and M. Shah. Consistent labeling of tracked objects in multiple cameras with overlapping fields of view. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 25:1355–1360, 2003.
[36] J. Kang, I. Cohen, and G. Medioni. Continuous tracking within and across camera streams. In *Proc. of IEEE Int'l Conf. Computer Vision and Pattern Recognition*, 2003.
[37] O. Javed, Z. Rasheed, K. Shafique, and M. Shah. Tracking across multiple cameras with disjoint views. In *Proc. of IEEE Int'l Conf. Computer Vision*, 2003.
[38] C. Stauffer and K. Tieu. Automated multi-camera planar tracking correspondence modeling. In *Proc. of IEEE Int'l Conf. Computer Vision and Pattern Recognition*, 2003.
[39] D. Makris, T. Ellis, and J. Black. Bridging the gaps between cameras. In *Proc. of IEEE Int'l Conf. Computer Vision and Pattern Recognition*, 2004.
[40] A. Rahimi, B. Dunagan, and T. Darrell. Simultaneous calibration and tracking with a network of non-overlapping sensors. In *Proc. of IEEE Int'l Conf. Computer Vision and Pattern Recognition*, 2004.
[41] Y. Shan, H. Sawhney, and R. Kumar. Unsupervised learning of discriminative edge measures for vehicle matching between non-overlapping cameras. In *Proc. of IEEE Int'l Conf. Computer Vision and Pattern Recognition*, 2005.
[42] O. Javed, K. Shafique, and M. Shah. Appearance modeling for tracking in multiple non-overlapping cameras. In *Proc. of IEEE Int'l Conf. Computer Vision and Pattern Recognition*, 2005.
[43] Y. Shan, H. Sawhney, and R. Kumar. Vehicle identification between non-overlapping cameras without direct feature matching. In *Proc. of IEEE Int'l Conf. Computer Vision*, 2005.
[44] K. Tieu, G. Dalley, and E. Grimson. Inference of non-overlapping camera network topology by measuring statistical dependence. In *Proc. of IEEE Int'l Conf. Computer Vision*, 2005.
[45] N. Gheissari, T. B. Sebastian, J. Rittscher, and R. Hartley. Person reidentification using spatiotemporal appearance. In *Proc. of IEEE Int'l Conf. Computer Vision and Pattern Recognition*, 2006.
[46] G. Unal, A. Yezzi, S. Soatto, and G. Slabaugh. A variational approach to problems in calibration of multiple cameras. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 29:1322–1338, 2007.
[47] Y. A. Sheikh and M. Shah. Trajectory association across multiple airborne cameras. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 30:361–367, 2008.
[48] F. Fleuret, J. Berclaz, R. Lengagne, and P. Fua. Multicamera people tracking with a probabilistic occupancy map. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 30:267–282, 2008.
[49] B Triggs. Camera pose and calibration from 4 or 5 known 3d points. In *Proc. of IEEE Int'l Conf. Computer Vision*, 1999.
[50] P Gurdjos and P. Sturm. Methods and geometry for plane-based self-calibration. In *Proc. of IEEE Int'l Conf. Computer Vision and Pattern Recognition*, 2003.
[51] M. Brown and D. Lowe. Recognising panoramas. In *Proc. of IEEE Int'l Conf. Computer Vision*, 2003.
[52] X. Wang, G. Doretto, T. Sebastian, J. Rittscher, and P. Tu. Shape and appearance context modeling. In *Proc. of IEEE Int'l Conf. Computer Vision*, 2007.
[53] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.
[54] T. Hofmann. Probabilistic latent semantic analysis. In *Proc. of Uncertainty in Artificial Intelligence*, 1999.
[55] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.
[56] H. W. Kuhn. Variants of the hungarian method for assignment problems. *Naval Research Logistics Quarterly*, 3:253–258, 1956.
[57] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical dirichlet process. *Journal of the American Statistical Association*, 2006.

Xiaogang Wang received BS degree from University of Science and Technology of China in Electrical Engineering and Information Science in 2001. He received MS degree from Chinese University of Hong Kong in Information Engineering in 2000. He is currently a PhD student at the Computer Science and Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. His research interests include computer vision and machine learning. He is a student member of the IEEE.

Kinh Tieu received the PhD degree in computer science in 2006 from the Massachusetts Institute of Technology. His research interests include pattern recognition, computer vision, and machine learning. He has worked on image retrieval, visual surveillance, and medical imaging.

Eric Grimson is the Bernard Gordon Professor of Medical Engineering in MIT's Department of Electrical Engineering and Computer Science. He is a member of MIT's Computer Science and Artificial Intelligence Laboratory, and Head of its Computer Vision Group. He also holds a joint appointment as a Lecturer on Radiology at Harvard Medical School and at Brigham and Women's Hospital. He received a B.Sc. (Hons) in Mathematics and Physics from the University of Regina in 1975 and a Ph.D. in Mathematics from MIT in 1980. Prof. Grimson is currently the Head of the Department of Electrical Engineering and Computer Science at MIT. Prior to this position, he has served as Associate Department Head for Computer Science, and as Education Officer for the department. Professor Grimson is a Fellow of the IEEE and of AAAI, and a recipient of the Bose Award for Undergraduate Teaching at MIT Prof. Grimson has research interests in computer vision, and in medical image analysis. Since 1975, he and his research group have pioneered state of the art methods for activity and behavior recognition, object and person recognition, image database indexing, site modeling, stereo vision, and many other areas of computer vision. Since the early 1990's, his group has been applying vision techniques in medicine for image guided surgery, disease analysis and computational anatomy.