# Learning Directed Intention-driven Activities using Co-Clustering*

Karthik Sankaranarayanan James W. Davis
Dept. of Computer Science and Engineering
Ohio State University
Columbus, OH 43210 USA
{sankaran,jwdavis}@cse.ohio-state.edu

## Abstract

*We present a novel approach for discovering directed intention-driven pedestrian activities across large urban areas. The proposed approach is based on a mutual information co-clustering technique that simultaneously clusters trajectory start locations in the scene which have similar distributions across stop locations and vice-versa. The clustering assignments are obtained by minimizing the loss of mutual information between a trajectory start-stop association matrix and a compressed co-clustered matrix, after which the scene activities are inferred from the compressed matrix. We demonstrate our approach using a dataset of long duration trajectories from multiple PTZ cameras covering a large area and show improved results over two other popular trajectory clustering and entry-exit learning approaches.*

## 1. Introduction

Wide-area activity analysis is an important step towards the goal of high-level scene understanding in large urban settings. Typically in such scenarios, activity is a result of pedestrian movement from one location to another. It is therefore necessary to study the typical behaviors and movements of people between important areas of interest within the scene. While it may be interesting to concentrate on the actual paths taken by pedestrians between these locations, it is equally important to understand their directed intentions at a higher semantic level. For example, a high-level scene understanding task would be learn that mornings are characterized by most people going from building A to building C (without caring whether they take the left sidewalk or the right sidewalk, etc.). Here, it is the origin and the destination that are of interest.

High-level activity analysis in such wide-area settings requires low-level detection and tracking of pedestrians. Typical approaches towards activity analysis use static cameras which generally have a small field-of-view. Hence, they are limited not only in the number of visible semantic scene entry and exit locations of the trajectories, but also the duration over which pedestrians can be tracked to their final destinations. Further, the trajectory clustering approaches they employ are based on the physical locations of trajectory observations rather than being agnostic to the actual routes taken by pedestrians to their destinations. Consequently, they are restrictive in learning directed intention-driven activities of pedestrians in such wide-area settings.

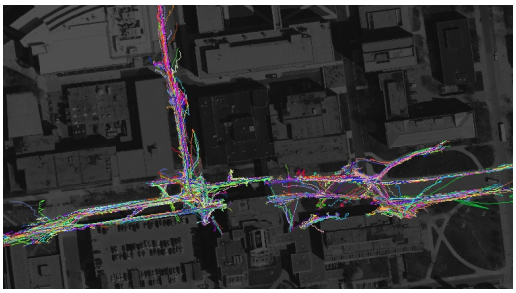


Figure 1. Some of the raw trajectories collected from multiple PTZ cameras overlaid on a shaded orthophoto of the area of interest.

In this paper, we propose a mutual information based co-clustering approach to learn directed pedestrian activity over large areas and simultaneously learn a large number of the semantic scene entry and exit locations. We employ multiple PTZ (pan-tilt-zoom) cameras and build an automatic PTZ active tracking system to detect and track pedestrians over longer durations and collect extended trajectories. We register these long distance trajectories to a common coordinate frame in the form of a UTM-based aerial orthophotograph (see Fig. 1). We then grid the orthophoto scene and use the UTM trajectory dataset to build an association matrix between every possible pair of scene locations, where the rows denote the starts and the columns denote the stops. We then employ an information theoretic

*Appears in *IEEE International Conference On Advanced Video and Signal Based Surveillance*, August 2010.

co-clustering approach to compress this association matrix such that those start locations which have similar distributions across stop locations cluster together and vice-versa. This is best achieved when the compressed matrix obtained is such that the loss of mutual information from the original association matrix is minimized.

The strength of our co-clustering approach towards this problem is that it exploits the duality and dependence of the starts on the stops (and vice-versa) and thus intertwines their clustering at all stages. It has been shown in the clustering literature [5] that a co-clustering technique is theoretically superior when a clear duality exists across rows and columns. This is because it implicitly performs an adaptive dimensionality reduction at each iteration thus resulting in an inherently "regularized" clustering. Such a clustering also gives the strength of associations between learned start and stop locations and consequently the strongest activities in the scene are inferred from these associations. Another benefit of the proposed approach is its ability to minimize the effect of trajectories arising from tracker failures with random stop locations. Such trajectories would only make weak contributions to their corresponding start and stop locations across the scene, and thus only the strongest pockets of starts and stops emerge from the co-clustering.

## 2. Related Work

Most of the work in scene activity analysis is based on analyzing pairwise similarity between trajectories and then clustering them together. In [24], a combination of spatial locations and velocities of observations is used within a modified Hausdorff distance to build trajectory similarities. Similarly in [7], a Euclidean distance metric is used for vehicle trajectory clustering. The approaches in [9, 10] also use variations of Hausdorff and Dynamic Time Warping based metrics. The problem with employing such distance measures is that they are adhoc (lacking probabilistic explanantion) and may not approximate true similarity. More recent work have moved away from such distance based methods and have focused more on modeling the spatial distribution of trajectory location and direction observations [23, 22] (see [15] for a survey). However, even these frameworks are based on individual observation level details of trajectories thus making them unsuitable for use in learning semantically higher-level intention-driven pedestrian activities over wide-areas.

Additionally, most the existing work is based on using static cameras and therefore use different background subtraction based approaches to detect and track people in the fixed view [12, 22, 23, 25]. Therefore, they are not readily extendable to wide-area settings where active tracking with PTZ cameras is necessary to obtain long duration trajectories of pedestrians. The limited views restrict the extent of the scene they are able to analyze and consequently the semantic activities of pedestrians that they can learn within them would be minimal. To the best of our knowledge, ours is the first work towards learning such wide-area pedestrian intentional activity with a semantically higher-level approach in mind than traditional trajectory clustering approaches.

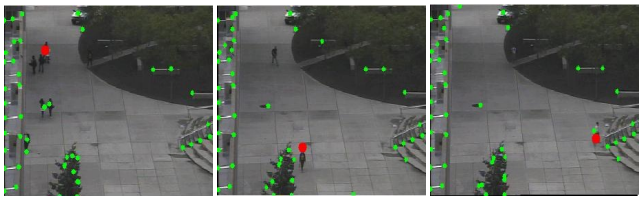


Figure 2. KLT based initialization at three different times showing the best feature point (in red) identified for tracking. Best viewed in color.

Recently, there has been much interest in using co-clustering techniques in the data clustering community [4, 6]. Two major approaches employing this idea are using spectral graph based partitioning [3] and information theoretic techniques [5]. The spectral graph formulation imposes a one-to-one association discovery between row and column clusters, which is too restrictive. On the other hand, information theoretic formulations do not impose any such restriction and learn association strengths across clusters. They have shown improvement in document clustering by simultaneously clustering words into topics and documents into document clusters. The key is to simultaneously maximize the mutual information across documents and words at all stages during clustering, which is better than standard one-sided clustering approaches such as spectral clustering [5]. Co-clustering has also been recently adopted in computer vision for learning intermediate image concepts by clustering local interest-point descriptors for scene classification [13].

## 3. UTM-based Trajectory Extraction

Analyzing people's activities driven by directional intentions requires pedestrian trajectories across large areas. In this section, we present a system to collect long duration pedestrian trajectories using multiple PTZ cameras and describe how to register the tracks to a UTM-based aerial orthophoto. The proposed system consists of an KLT-based technique to automatically initialize tracking, a covariance-based PTZ tracker to actively follow a selected target, and a registration framework to map the pan-tilt camera orientations of trajectories to a common UTM-based reference frame. We focus on tracking pedestrians, though the method could be applied to other moving targets (e.g., vehicles).

Figure 3. Active tracking of a target over a large area showing considerable changes in views.

### 3.1. KLT-based Automatic Initialization

Pedestrian detection techniques are well studied in the literature [2, 11], however most of the learning-based techniques are view specific which is unsuitable for our domain since a PTZ camera overlooking a large area can have a wide range of pedestrian views. We employ the KLT-based feature tracker [20] to automatically detect "good features to track" in the scene. Since such features could include points on the background, we employ a simple motion model to eliminate background feature points and select the feature point with the largest range of motion over a few frames as the best feature (see Fig. 2)

$$f_{\text{best}} = \arg\max_{f} ||L_{t+k}(f) - L_t(f)|| \qquad (1)$$

where $L_t(f) \in \mathbb{R}^2$ denotes the pixel location of a KLT feature $f$ at time $t$ (the delay $k$ can be picked according to the frame rate). We then apply a frame differencing technique locally around the selected feature point to detect the target blob and calculate its centroid. The target centroid is then handed over to the appearance-based PTZ tracker for active tracking of the target across the entire scene.

### 3.2. Wide-Area PTZ Active Tracking

Once the initialization framework detects the centroid of the target to track, the active camera system needs to continually follow the target as it moves throughout the scene. We use the appearance-based tracking algorithm of [16] and build a covariance descriptor of the target using a combination of position, color, and gradient features $f_k = [x\ y\ R\ G\ B\ I_x\ I_y]$. This covariance descriptor is then matched across successive frames in a small spatial window to track the target. The pan-tilt of the PTZ camera is continually moved to keep the target centered in its view using the active camera model of [17]. In addition, the camera's zoom is continually adjusted so that the target being tracked is always of constant size irrespective of the target's distance from the camera. Since we wish to track targets over long distances across the scene, the appearance of targets undergo considerable change. To adapt to this and to overcome noise, a model update method [16] is employed by keeping a set of the most recent covariance matrices and computing their mean on the Riemannian manifold. Figure 3 shows a few frames from a tracking sequence.

Typical reasons for a target to leave the scene would be to enter a building or car, walk behind large occlusions, leave the field-of-coverage of the camera, etc. In order for the system to detect such an event and stop tracking the target, we analyze the evolution of the covariance matching distance over time. Since the tracker is based on matching the covariance matrix descriptor of the target with the model matrix, this matching distance (lower means a better match) is used to determine if the target is lost or not.

A simple technique would be to have a fixed threshold value for the matching distance. However the matching distance is sensitive to the size of the target, illumination changes, frame rate, etc. In order to take care of these variations, our algorithm keeps a list of previous match distances from a fixed-size (yet floating) time window in the past. It then computes a mean of these values to obtain an average matching distance which is sensitive to the current context. A matching threshold is computed as a scaled value of this average matching distance and a bad match is marked as being found if this threshold is exceeded. However, the algorithm does not immediately mark the target as lost because this bad match could have been due to noisy frames or a temporary occlusion (such as from a tree, passing car, etc.). In such cases, the algorithm is able to pick up the target as soon as it reappears (see Fig. 4). If the matching distance exceeds the threshold for a finite number of successive frames, then the target is classified as "lost". In such an event, the system terminates the tracking and moves to new PTZ home location (randomly picked from a manually selected set of 15 home locations) and waits for an automatically detected pedestrian to track (and this process repeats).

### 3.3. Trajectory Registration

While a target is being tracked, its location is recorded in the form of pan-tilt orientations of the PTZ camera. However, in order to incorporate trajectory information from multiple PTZ cameras and perform activity analysis over a large area, we need to map this pan-tilt trajectory data to a common reference frame. Several techniques have been proposed to calibrate correspondence information across views from different cameras [21, 8, 1]. When these views overlap, static features are selected to compute an assumed homography between the two views and calibrate them to a single ground plane. However, since we require mapping pan-tilt coordinates of a PTZ camera to a common ground

visible occluded visible

Figure 4. Tracking is continued as the target reappears from behind a temporary occlusion.

plane, we employ the registration framework of [18]. The pan-tilt coordinates of each trajectory observation are converted to their corresponding UTM location using

$$\begin{bmatrix} x_g \\ y_g \\ 1 \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & t_x \\ a_3 & a_4 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \tan\phi \cdot \cos\theta \\ \tan\phi \cdot \sin\theta \\ 1 \end{bmatrix} \quad (2)$$

where ($\theta$,$\phi$) denotes the (pan,tilt) and ($x_g$,$y_g$) the UTM ground point location of each trajectory observation. The registration parameters $a_1$, $a_2$, $a_3$, $a_4$, $t_x$, and $t_y$, are learned using the technique described in [18]. Once we register these trajectories with the orthophoto, we have a common reference coordinate system in which all trajectory information from multiple cameras can be analyzed in an integrated manner.

# 4. Intentional Activity Discovery

In this section, we describe our mutual information based co-clustering technique to automatically discover semantic scene entry and exit locations from the trajectory dataset and infer activity associations between them.

We start by gridding the space of UTM locations for the area of interest into cells of size 1 x 1 meters. Each cell is a potential start or a stop location. Let us denote the starts and stops by discrete jointly distributed random variables $\mathbf{X}$ (starts) and $\mathbf{Y}$ (stops), where $\mathbf{X} \in \mathbf{\Omega}$ and $\mathbf{Y} \in \mathbf{\Omega}$ and the set $\mathbf{\Omega}$ corresponds to all possible scene grid locations. We wish to cluster those start states together which exhibit similar distributions across stop states in the trajectory data and simultaneously cluster those stop states together which exhibit similar distributions across start states. In other words, we wish to find compact representations of $\mathbf{X}$ and $\mathbf{Y}$, say $\hat{\mathbf{X}}$ and $\hat{\mathbf{Y}}$, which capture the "similarity" in distributions within them (they would take values only in a subset of $\mathbf{X}$ and $\mathbf{Y}$). This can be achieved by obtaining $\hat{\mathbf{X}}$ and $\hat{\mathbf{Y}}$, such that the mutual information between them $I(\hat{\mathbf{X}};\hat{\mathbf{Y}})$ is maximized, where the mutual information between random variables $\mathbf{X}$ and $\mathbf{Y}$ is given by

$$I(\mathbf{X};\mathbf{Y}) = \sum_{\mathbf{x}\in\mathbf{X}} \sum_{\mathbf{y}\in\mathbf{Y}} p(\mathbf{x},\mathbf{y}) \log \frac{p(\mathbf{x},\mathbf{y})}{p(\mathbf{x})p(\mathbf{y})} \quad (3)$$

## 4.1. Start-Stop Association Matrix

We next use the UTM trajectory dataset to build the association matrix which captures the joint distribution between the start states $\mathbf{X}$ and stop states $\mathbf{Y}$ in the scene. For each trajectory, we use a Gaussian likelihood model to calculate its probability of starting from any particular start location in the scene ($\mathbf{x_i} =< x_i, y_i >$) with

$$p_{\mathbf{x_i}}(\mathbf{x}) = \exp(-\frac{||\mathbf{x} - \mathbf{x_i}||^2}{\sigma}) \quad (4)$$

where $\mathbf{x} =< x, y >$ is the start location of that trajectory. Similarly, for that trajectory, we calculate its probability of stopping at any particular stop location in the scene ($\mathbf{y_i}$) with

$$p_{\mathbf{y_i}}(\mathbf{y}) = \exp(-\frac{||\mathbf{y} - \mathbf{y_i}||^2}{\sigma}) \quad (5)$$

where $\mathbf{y}$ is the stop location of that trajectory. Assuming independence, we multiply $p_{\mathbf{x_i}}(\mathbf{x})$ and $p_{\mathbf{y_i}}(\mathbf{y})$ to calculate the joint probability of starting at $\mathbf{x_i}$ and stopping at $\mathbf{y_i}$ for that trajectory. By repeating this for all trajectories from the dataset and summing up all the joint probabilities, we obtain the association matrix between starts $\mathbf{X}$ and stops $\mathbf{Y}$ and normalize it to make it a joint probability distribution $p(\mathbf{x},\mathbf{y})$ (with new data, the matrix can be updated online and then re-normalized). Also note that with this modeling of start-stop associations, the noise trajectories (arising from tracker failures, etc.) end up making weak contributions to their corresponding start and stop locations across the scene. Therefore the model is able to minimize their effect on the association matrix ensuring that only the strongest pockets of starts and stops emerge from the co-clustering.

## 4.2. Mutual Information based Co-Clustering

Formally, we can express the clustering/compression of $\mathbf{X}$ and $\mathbf{Y}$ (to $\hat{\mathbf{X}}$ and $\hat{\mathbf{Y}}$) in terms of two mapping functions $\hat{\mathbf{X}} = C_{\mathbf{X}}(\mathbf{X})$ and $\hat{\mathbf{Y}} = C_{\mathbf{Y}}(\mathbf{Y})$. Therefore, our criteria is to find mapping functions $C_{\mathbf{X}}$ and $C_{\mathbf{Y}}$ such that the mutual information between the resulting clusters $I(\hat{\mathbf{X}};\hat{\mathbf{Y}})$ is maximized. Given the mutual information in the original dataset $I(\mathbf{X};\mathbf{Y})$, this criteria is equivalent to minimizing the loss of mutual information as given by

$$\Delta MI = I(\mathbf{X};\mathbf{Y}) - I(\hat{\mathbf{X}};\hat{\mathbf{Y}}) \quad (6)$$

Based on [5], this loss of mutual information ($\Delta MI$) is equivalent to calculating the Kullback-Leibler (KL) divergence between the original distribution $p(\mathbf{x},\mathbf{y})$ and its "approximation" distribution $q(\mathbf{x},\mathbf{y})$ as given by

$$\Delta MI = D(p(\mathbf{X},\mathbf{Y})||q(\mathbf{X},\mathbf{Y})) \quad (7)$$

where $D(\cdot||\cdot)$ denotes the KL-divergence, and the "approximation" distribution $q(\mathbf{X}, \mathbf{Y})$ corresponds to mapping functions $C_\mathbf{X}$ and $C_\mathbf{Y}$ as given by

$$q(\mathbf{x}, \mathbf{y}) = p(\hat{\mathbf{x}}, \hat{\mathbf{y}})p(\mathbf{x}|\hat{\mathbf{x}})p(\mathbf{y}|\hat{\mathbf{y}}) \tag{8}$$

where $\mathbf{x} \in \hat{\mathbf{x}}$ and $\mathbf{y} \in \hat{\mathbf{y}}$.

The clustering algorithm is initialized by randomly picking initial mapping functions $C^0_\mathbf{X}$ and $C^0_\mathbf{Y}$, and by specifying the desired number of row and column clusters. At each iteration, we wish to compute the "approximation" distribution $q(\mathbf{x}, \mathbf{y})$. For this, we first calculate the joint distribution $p(\hat{\mathbf{x}}, \hat{\mathbf{y}})$ based on the mapping functions

$$p(\hat{\mathbf{x}}, \hat{\mathbf{y}}) = \sum_{\mathbf{x}\in\hat{\mathbf{x}}} \sum_{\mathbf{y}\in\hat{\mathbf{y}}} p(\mathbf{x}, \mathbf{y}) \tag{9}$$

The conditional distributions $p(\mathbf{x}|\hat{\mathbf{x}})$ and $p(\mathbf{y}|\hat{\mathbf{y}})$ for Eqn. 8 are calculated using

$$p(\mathbf{x}|\hat{\mathbf{x}}) = \frac{p(\mathbf{x})}{p(\hat{\mathbf{x}})}, \quad p(\mathbf{y}|\hat{\mathbf{y}}) = \frac{p(\mathbf{y})}{p(\hat{\mathbf{y}})} \tag{10}$$

where the marginal distributions for $\mathbf{x}$ and $\hat{\mathbf{x}}$ are given as $p(\mathbf{x}) = \sum_{\mathbf{y}\in\mathbf{Y}} p(\mathbf{x}, \mathbf{y})$ and $p(\hat{\mathbf{x}}) = \sum_{\mathbf{x}\in\hat{\mathbf{x}}} p(\mathbf{x})$ and similarly for $\mathbf{y}$ and $\hat{\mathbf{y}}$.

At each iteration $t$ of the algorithm, we update the mapping functions back-and-forth as follows (as described in [5]). First, for each row $\mathbf{x}$, we find its new cluster assignment $C^{t+1}_\mathbf{X}(\mathbf{x})$ using

$$C^{t+1}_\mathbf{X}(\mathbf{x}) = \arg\min_{\hat{\mathbf{x}}} D(p(\mathbf{y}|\mathbf{x})||q^t(\mathbf{y}|\hat{\mathbf{x}})) \tag{11}$$

while keeping $C^{t+1}_\mathbf{Y}(\mathbf{y}) = C^t_\mathbf{Y}(\mathbf{y})$. Here, the conditional is obtained as $q^t(\mathbf{y}|\hat{\mathbf{x}}) = q^t(\mathbf{y}|\hat{\mathbf{y}})q^t(\hat{\mathbf{y}}|\hat{\mathbf{x}})$. We now update distributions from Eqns. 10, 9, and 8 with this new row mapping $C^{t+1}_\mathbf{X}(\mathbf{x})$. After this, for each column $\mathbf{y}$, we find its new cluster assignment $C^{t+2}_\mathbf{Y}(\mathbf{y})$ using

$$C^{t+2}_\mathbf{Y}(\mathbf{y}) = \arg\min_{\hat{\mathbf{y}}} D(p(\mathbf{x}|\mathbf{y})||q^{t+1}(\mathbf{x}|\hat{\mathbf{y}})) \tag{12}$$

while keeping $C^{t+2}_\mathbf{X}(\mathbf{x}) = C^{t+1}_\mathbf{X}(\mathbf{x})$. Also, the conditional is given as $q^{t+1}(\mathbf{x}|\hat{\mathbf{y}}) = q^{t+1}(\mathbf{x}|\hat{\mathbf{x}})q^{t+1}(\hat{\mathbf{x}}|\hat{\mathbf{y}})$. Again, we update distributions from Eqns. 10, 9, and 8 using this new column mapping $C^{t+2}_\mathbf{Y}(\mathbf{y})$.

We now calculate the KL divergence from Eqn. 7 to measure the loss in mutual information ($D^{t+2}$) and repeat the co-clustering iterations until the loss in mutual information converges ($D^t - D^{t+2} < \epsilon$). We set $\epsilon = 10^{-3}$.

Once the co-clustering converges, we get the final start and stop mapping functions $C_\mathbf{X}$ and $C_\mathbf{Y}$ along with the corresponding compressed matrix $p(\hat{\mathbf{x}}, \hat{\mathbf{y}})$ which indicates the strength of associations across start and stop clusters. We pick a threshold and remove start/stop clusters with very low probability values. (Hence we can choose a large number of expected positive clusters during initialization).

# 5. Experiments

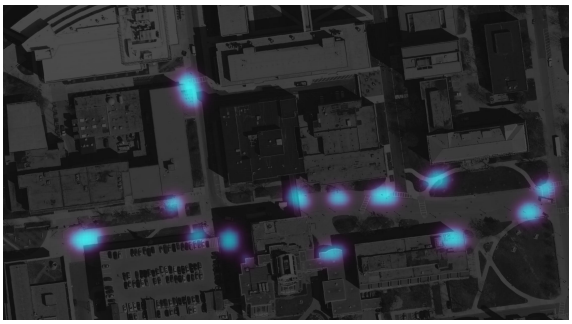
In this section, we describe different experiments that we performed in order to evaluate the proposed approach. We collected trajectory data using the system described in Sect. 3, built the association matrix using the dataset, and automatically learned semantic entry and exit locations in the scene using the described co-clustering approach. We further used the compressed matrix to infer strong direction-intended activities (entry-exit pairs) in the scene. We then mapped trajectories back to these strong activities to obtain the popular trajectory clusters in the scene.

In our system, we used two PTZ surveillance cameras (Pelco Spectra III SE and IV) mounted atop a building eight stories high overlooking different areas such that they cover a wide field-of-coverage (approximately 400 x 200 meters). This area includes 8 buildings, a parking garage, and numerous streets and sidewalks. Using the system described in Sect. 3, we collected approximately 1500 long duration pedestrian trajectories across the entire region (see Fig. 1).

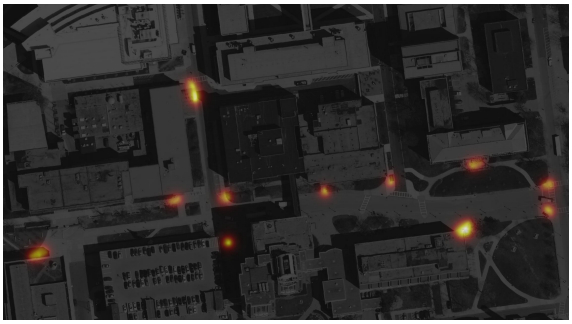
## 5.1. Semantic Entries and Exits

Using the collected trajectory dataset, we built the association matrix using the technique described in Sect. 4.1 with $\sigma$=3. We then used this matrix $p(\mathbf{x}, \mathbf{y})$ as the input to the co-clustering algorithm. Based on our expectation of the scene, we manually initialized the number of start and stop clusters to 15 each. We then ran the co-clustering algorithm to obtain the clustering label assignment mapping functions $C_\mathbf{X}$ and $C_\mathbf{Y}$. Since the algorithm can run into a local minima [5], we performed 5 repetitions of the algorithm and picked the one with the least loss in mutual information. We picked a threshold of $10^{-5}$ and set all start-stop cluster associations below it to be zero. Using the row cluster label assignments and the marginal distribution of the start states, we plotted a probability map overlaid onto the orthophoto of the scene to display the start state clusters that the algorithm discovered. As seen in Fig. 5(a), the algorithm discovered 14 strong start state clusters (the dataset had 15 actual start home locations but one of them had very few trajectories originating from it). Similarly, using the column cluster label assignments, we learned the stop states in the scene as shown in Fig. 5(b). In this case, the algorithm discovered 11 stop state clusters. The remaining noisy start-stop locations (from tracker failures, etc.) had very weak associations and were relegated to the remaining non-acceptable clusters.

To give the reader a sense of the discovered clusters, we point out the semantic meaning for some of these locations. The stop states A, J, and K seen in Fig. 6 correspond to areas where the field-of-coverage of the cameras end. States B and H correspond to locations where there is a building entry and people walking behind the building thus going out of view of the camera. Locations C, F, and I are entries



(a)



(b)

Figure 5. Semantic scene (a) entries and (b) exits discovered in the scene (best viewed in color).

into different buildings. Locations D and G are areas where people leave the scene due to large occlusions (buildings). Similarly, the start clusters are either locations where people enter the field-of-coverage from behind buildings or where there are building exits in the scene.

### 5.2. Activities

The compressed matrix $p(\hat{\mathbf{x}}, \hat{\mathbf{y}})$ from the co-clustering algorithm gives the strength of association between the start state and stop state clusters. This compressed matrix showing the learned activities is shown in Table 1.

Each activity value in Table 1 indicates the strength of association between the pair of corresponding entry X and exit Y locations. Therefore, this indicates the popularity of the activity of intending to go from location X to location Y irrespective of actual path taken by the person between X and Y. The strongest activities in the scene corresponding to each entry location are shown in Fig. 6. These associations are indicated in boldface in Table 1.

### 5.3. Trajectory Clusters

Next, we used the strongest activity associations learned from the compressed matrix $p(\hat{\mathbf{x}}, \hat{\mathbf{y}})$ to extract pathway clusters of trajectories from the original dataset. For display, we picked the top 30 trajectories which had the highest probability of starting and stopping at an activity's corresponding start and stop location respectively. We repeated this for each activity in decreasing order of activity strength. The resulting trajectory clusters are shown in Fig. 7.

Table 1. Activity strength between starts (rows) and stops (columns). Associations in boldface are shown in Fig. 6 (all values are x$10^{-2}$).

| | A | B | C | D | E | F | G | H | I | J | K |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | **8.87** | 1.76 | 0 | 1.91 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 0 | 0 | **7.23** | 0.66 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0.15 | 0 | 2.36 | **5.09** | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | **1.12** | 0 | 0.65 | 0 | 0.65 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0.29 | 5.13 | **6.96** | 0 | 0.73 | 0 | 0 | 0 | 0 | 0 | 0 |
| 6 | 0 | **0.75** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 7 | **8.65** | 0.13 | 3.99 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 8 | 0 | 0 | 0 | 0 | 0 | **5.06** | 0 | 0 | 0 | 0 | 0 |
| 9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 5.37 | 0 | **7.55** |
| 10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **4.14** | 0 | 3.78 |
| 11 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **4.77** | 0 | 0 | 2.58 |
| 12 | 0 | 0 | 0 | 0 | 0 | 0 | 0.18 | 0 | 0 | **2.54** | 0 |
| 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **2.52** | 0 | 0 | 0 |
| 14 | 0 | 0 | 0 | 0 | 0 | 0.41 | **3.84** | 0 | 0 | 0 | 0 |

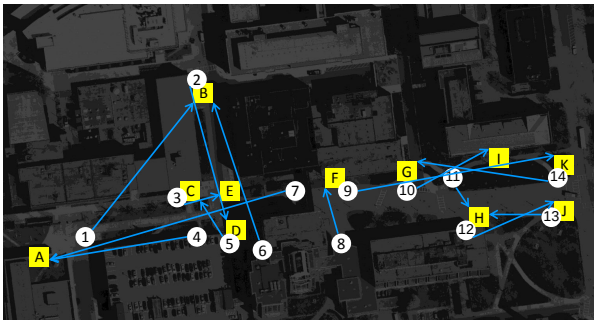



Figure 6. Activity map showing the strongest activity for each scene entry location. White circles - starts. Yellow squares - stops.

An important aspect of the proposed approach towards learning directed intention-driven activities in the scene is that trajectories which originate and terminate at the same pair of starts and stops should cluster together irrespective of the actual physical proximities of their observations along the trajectory (because targets could take different routes). An example of this is discovered in one of the trajectory clusters (see Fig. 8). All the tracks go from location 5 (bottom-right) to location C (top-left) and pedestrians either go to the left or right of the large grass patch. In either case, the pedestrian intention is to get to location C and consequently both paths correspond to the same semantic activity. Typical trajectory clustering algorithms based on spatial location or spatial distribution of observations would not be able to learn such activities in the scene.

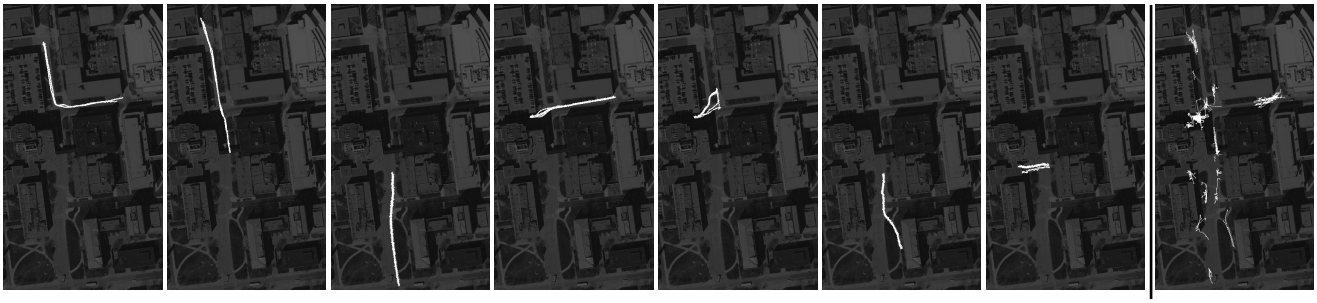


Figure 7. Trajectory clusters corresponding to the 7 strongest scene activities (scene rotated). The last image shows the weakest cluster.

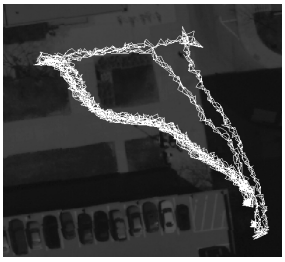


Figure 8. Activity involving different possible paths taken by pedestrians but with same origin/destination intention.

### 5.4. Comparison with Existing Approaches

In order to compare our proposed technique with current trajectory clustering approaches, we tested the trajectory clustering algorithm of [24]. This algorithm uses a pairwise similarity metric between trajectories based on spatial proximity and velocities of observations and uses a modified Hausdorff distance. It then uses this similarity matrix in a spectral clustering framework to extract trajectory clusters. Figure 9 shows two of the strongest clusters learned by this algorithm. Since the algorithm uses individual trajectory observations, the similarity of observations causes bleeding of pairwise similarity values across trajectories consequently resulting in many "true" underlying clusters being clustered together (under-segmented).

Since the above approach incorporates individual trajectory observations, we also tested a modified version of the above approach by building a pairwise similarity matrix across trajectories based only on their start and stop locations. The strongest two clusters from the modified approach are shown in Fig 10. Even in this case, we observe that there is under-segmentation since the pairwise start-stop similarity measure between trajectories bleeds across trajectories of different underlying activities resulting in fewer and less meaningful clusters. (Notice how each cluster still spans multiple underlying starts and stop locations.)

We also tested the entry-exit learning approach of [14] to see if it is able to detect semantically meaningful start and stop locations in the scene, and compared it to our results.

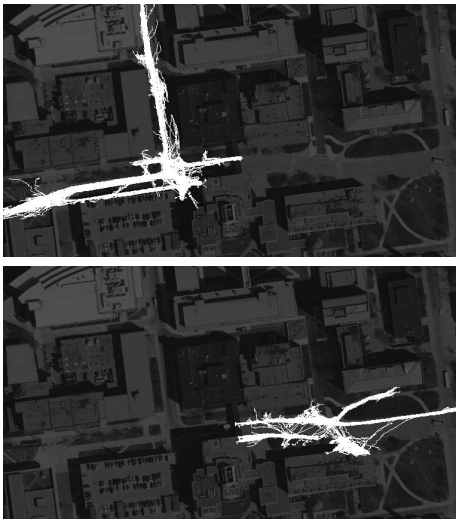


Figure 9. Top 2 strongest clusters obtained using [24].

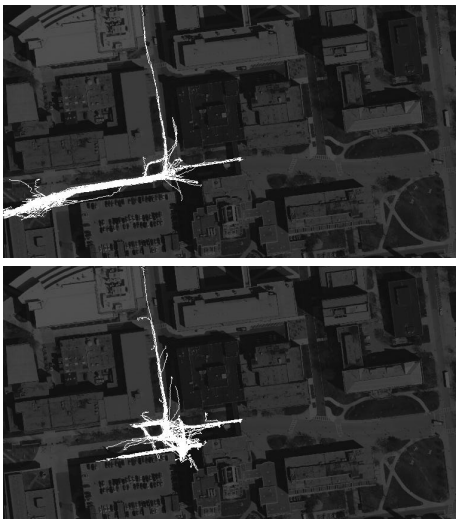


Figure 10. Top 2 strongest clusters obtained with the modified approach of [24] using start-stop based pairwise trajectory similarity.

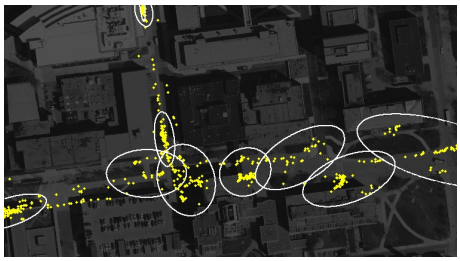


Figure 11. Stop state clusters obtained using the approach of [14].

This approach is based on an Expectation-Maximization algorithm to fit a mixture of Gaussians to the trajectory start and stop locations. Once a mixture of Gaussian model is learned, a threshold mechanism based on the density of points in each distribution is used to eliminate "sparsely" populated clusters (considering them to be noise). This density is calculated as the ratio of number of points classified as belonging to a particular cluster and the area of the ellipse based on the eigenvalues of the covariance matrix.

We ran the above algorithm on the stop locations in the trajectory dataset and picked the number of clusters automatically using the Bayesian Information Criterion (BIC) [19]. The maximum value for BIC criterion was obtained for 12 stop clusters. We then used the density based thresholding technique and picked $5\%$ of the maximum density to be the threshold below which the clusters are considered as noise. Using this approach we obtained 10 clusters whose distributions are shown in Fig 11. As seen in the figure, this approach clusters together locations based on their spatial proximity without respect for the manifold of the actual trajectories and therefore multiple underlying stop locations end up being clustered together. Also notice that some useful stop locations are incorrectly classified as noise.

## 6. Summary and Future Work

We proposed a novel co-clustering approach based on mutual information to learn directed intention-driven pedestrian activity over large areas. The proposed approach also simultaneously learns a large number of the semantic scene entry and exit locations. To do this, we employed an information theoretic co-clustering approach to compress the start-stop association matrix such that those start locations which have similar distributions across stop locations cluster together and vice-versa. This minimizes the loss of mutual information between the original association matrix and the compressed matrix. The strength of our co-clustering approach towards this problem is that it exploits the duality between starts and stops. The co-clustering labels are also used to infer the strongest activities in the scene. We demonstrated our approach using a dataset of long duration trajectories from multiple cameras covering a large area. We also demonstrated better results than alternate trajectory clustering and start-stop learning approaches. In the future, we plan to use the learned activities for scene modeling and anomaly detection.

**Acknowledgement:** We gratefully acknowledge the support of the U.S. Department of Energy through the LANL/LDRD Program under LDRD-DR project 'RADIUS' for this work.

## References

[1] M. Brown and D. Lowe. Recognising panoramas. In *ICCV*, 2003. 3
[2] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. 2
[3] I. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *ACM SIGKDD*, 2001. 2
[4] I. Dhillon, S. Mallela, and R. Kumar. A divisive information theoretic feature clustering algorithm for text classification. In *JMLR*, 2003. 2
[5] I. Dhillon, S. Mallela, and D. Modha. Information-theoretic co-clustering. In *ACM SIGKDD*, 2003. 2, 4, 5
[6] R. El-Yaniv and O. Souroujon. Iterative double clustering for unsupervised and semi-supervised learning. In *ECML*, 2001. 2
[7] Z. Fu, W. Hu, and T. Tan. Similarity based vehicle trajectory clustering and anomaly detection. In *ICIP*, 2005. 2
[8] P. Gurdjos and P. Sturm. Methods and geometry for plane-based self calibration. In *CVPR*, 2003. 3
[9] I. Junejo, O. Javed, and M. Shah. Multi-feature path modeling for video surveillance. In *ICPR*, 2004. 2
[10] E. Keogh and M. Pazzani. Scaling up dynamic time. In *ACM SIGKDD*, 2000. 2
[11] B. Leibe, E. Seemann, and B. Schiele. Pedestrian detection in crowded scenes. In *CVPR*, 2005. 2
[12] J. Li, S. Gong, and T. Xiang. Scene segmentation for behaviour correlation. In *CVPR*, 2008. 2
[13] J. Liu and M. Shah. Scene modeling using co-clustering. In *ICCV*, 2007. 2
[14] D. Makris and T. Ellis. Automatic learning of activity-based semantic scene model. In *AVSS*, 2003. 7, 8
[15] B. Morris and M. Trivedi. Trajectory based primitive events for learning and recognizing activity. *IEEE Transactions on Circuits and Systems for Video Technology*. 2
[16] F. Porikli, O. Tuzel, and P. Meer. Covariance tracking using model update based on means on Riemannian manifolds. In *CVPR*, 2006. 3
[17] K. Sankaranarayanan and J. Davis. An efficient active camera model for video surveillance. In *WACV*, 2008. 3
[18] K. Sankaranarayanan and J. Davis. A fast linear registration framework for multi-camera GIS coordination. In *AVSS*, 2008. 4
[19] G. Schwarz. Estimating the dimension of a model. *Annals of Statistics*. 7
[20] J. Shi and C. Tomasi. Good features to track. In *CVPR*, 1994. 2
[21] B. Triggs. Camera pose and calibration from 4 or 5 known 3d points. In *ICCV*, 1999. 3
[22] X. Wang, K. Ma, G. Ng, and E. Grimson. Trajectory analysis and semantic region modeling using a nonparametric bayesian model. In *CVPR*, 2008. 2
[23] X. Wang, X. Ma, and E. Grimson. Unsupervised activity perception by hierarchical bayesian models. In *CVPR*, 2007. 2
[24] X. Wang, K. Tieu, and E. Grimson. Learning semantic scene models by trajectory analysis. In *ECCV*, 2006. 2, 7, 8
[25] T. Zhang, H. Lu, and S. Li. Learning semantic scene models by object classification and trajectory clustering. In *CVPR*, 2009. 2