Extracting Pathlets From Weak Tracking Data*

Kevin Streib James W. Davis
Dept. of Computer Science and Engineering
Ohio State University, Columbus, OH 43210
{streib,jwdavis}@cse.ohio-state.edu

Abstract

We present a novel framework for extracting "pathlets" from tracking data. A pathlet is defined as a motion region that contains tracks having the same origin and destination in the scene and that are temporally correlated. The proposed method requires only weak tracking data (multiple fragmented tracks per target). We employ a probabilistic state space representation to construct a Markovian transition model and estimate the scene entry/exit locations. The resulting model is treated as a set of vertices in a graph and a similarity matrix is built which describes broader non-local relationships between states. A Spectral Clustering approach is then used to automatically extract the pathlets of the scene. We present experimental results from scenes of varying difficulty and compare against other approaches.

1. Introduction

One important task in video surveillance (of many) is to observe/model an active environment and judge the scene normalcy. Often the modeling is performed by collecting data over long periods of time and extracting trajectory clusters or semantic regions using trajectory-based [3, 7, 8, 13, 14, 20] or feature-based [5, 6, 16, 17, 18] approaches. Unfortunately, many of these algorithms are problematic when there are a large number of trajectories to process. Furthermore, many of the extracted semantic regions suffer from one of two limitations. Namely, a region may contain several different paths, or regions contain single paths but overlap in areas where multiple paths intersect spatially and travel in the same direction within the intersection.

In this work, our goal is to carve the space into "pathlets", which we define as contiguous areas with a similar origin and destination for tracks passing through the region and that are temporally correlated. Our approach begins by tracking objects using a real-time multi-object tracker based on the well-known Kanade-Lucas-Tomasi (KLT) feature tracker [11]. Using the KLT tracker typically results in "weak tracking", where multiple, and frequently broken, tracks are found for each target object. We choose such a tracker because the resulting tracklets are more informative than pixel-based optical flow and the approach is capable of tracking large numbers of features on-line in both sparse and crowded scenes (unlike slower, stronger trackers), a necessity when dealing with complex urban environments. Furthermore, by tracking multiple features on-line, the true temporal correlation of motion regions can be computed. Additionally, a method capable of extracting pathlets from short fragmented tracks will also be applicable to the output of stronger trackers (the reverse is typically not the case for many algorithms).

From an imposed grid on the image, we quantize the tracks into a set of location-orientation states. We then produce a Markovian state transition model and probabilistically determine the entry/exit states in the scene. Our spatial Markovian assumption holds for weak tracks as long as there are enough overlapping tracks produced over time on a path (which holds in typical scenes). Finally, we build a similarity matrix to define the non-local relationship between states (based on entry/exit distributions for tracks in each state and temporal correlation) and use a version of Spectral Clustering [19] to partition the state space into clusters (pathlets). By clustering a *state space* rather than the *trajectories* themselves, our approach is not adversely affected by large numbers of trajectory data (in fact, the state space is enhanced by more data).

We evaluate our method on five different active urban scenes with varying degrees of difficulty and compare our results with trajectory clusters extracted from a trajectory-based approach [17] and semantic regions extracted from a feature-based method [5].

2. Related Work

Scene-based activity analysis is typically approached via algorithms that utilize either motion or appearance features

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

[4, 5, 6, 9, 16, 18] or trajectories [1, 3, 7, 8, 10, 13, 14, 15, 17, 20]. The approaches using motion or appearance features analyze activities without relying on tracking. Among the features that have been used are optical flow [9, 16, 18], combinations of optical flow and appearance metrics [5], spatiotemporal gradients [4], and static and moving pixel comparisons [6]. Furthermore, instead of working at a pixel level, several of these approaches [4, 6, 9, 18] use features extracted from small cells or spatiotemporal volumes where behavior patterns are generally more consistent.

A variety of approaches have been developed utilizing trajectories to perform scene activity analysis. In [3, 17] Spectral Clustering is performed on pairwise trajectory similarity matrices. An envelope approach is used in [7, 8] to determine if tracks should be assigned to existing routes or formed into new routes. In [1] a GMM was used to model the pdf of transition vectors up to twenty time steps in advance for each pixel location. A similar approach is presented in [10] where they use Kernel Density Estimation to learn a model for the joint probability of a transition between any two image points and the time taken to complete the transition. Vector quantization is used in [13] to reduce trajectories to a set of prototypes. In [15] observations are treated as words and trajectories as documents which are clustered via language processing algorithms. Hierarchical clustering is employed in [14] to combine similar tracks into HMMs. In [20] scenes are divided into multiple cell blocks, each potentially containing several motion patterns of trajectories that are parameterized by a quadratic model, and a GMM is used to extract the primary motion patterns for a graph-cut algorithm used to group the motion patterns for each mode.

Several algorithms [3, 5, 6, 7, 8, 13, 14, 16, 17, 18, 20] extract either paths or semantic regions from the scene. However, the extracted regions generally have one of two limitations. Algorithms such as [5, 6] segment the scene into non-overlapping regions which may cause each region to contain several different motion patterns. While algorithms such as [3, 7, 8, 13, 14, 16, 17, 18, 20] are capable of handling multi-directionality by extracting regions for entire paths, the resulting regions may overlap and contain a similar motion direction since sections of paths may intersect spatially and travel in the same direction throughout the intersection. Thus, mapping a target's instantaneous location and direction to a unique region is not always possible. The algorithm presented in [15] does not suffer from either limitation, but its clustering complexity is related to the number of tracks. Since we cluster a state space, our approach has a fixed clustering complexity. Furthermore, the resulting pathlets capture distinct motion patterns and can be mapped to uniquely given a target's instantaneous location and direction.

3. Probabilistic Scene Model

In our approach, modeling the scene is a three-step process consisting of gathering tracklets, generating a probabilistic state space, and estimating the track entry and exit locations of the scene. Pathlets are extracted directly from this model.

3.1. Weak Tracking

The "weak tracker" employed in this paper utilizes the OpenCV implementation of the Kanade-Lucas-Tomasi (KLT) feature tracker [11]. To limit features/tracks to only moving objects, we created a motion mask between frames and employed selected features within this area. The motion mask is created using a variant of image differencing based on Weber's Law (a model of perceptual just-noticeable-difference [2]), which tends to better handle low-contrast image differences in shadow regions. To overcome any erroneous drift or feature matching during tracking, only those features that move in a continuous direction and lie in the motion mask are accepted. We refer to this particular tracker as "weak" because multiple short/broken tracklets per target are typically produced. Figure 1 shows example weak tracks. Again, our goal is to be able to employ such a weak tracker because it is able to track large numbers of features in real-time, a necessity for online analysis of complex scenes such as a busy and crowded urban environment.

Figure 1. Example tracks for multiple moving objects in an urban environment.

3.2. Markovian State Space

Motivated by the grid-based approaches of [4, 6, 9, 15, 18, 20], we divide the scene into disjoint cells of size $L \times L$ pixels to reduce the problem dimensionality. The collected tracks are then mapped into local sets of [grid-location, motion-direction] states. For each cell a track passes through, the coordinates of the track within the cell are approximated by a least-squares line fit whose orientation is quantized into 45° orientations. Thus, a state $s_i = [(x, y), \theta]$ is defined as an $L \times L$ cell location centered at (x, y) with angle quantization θ. For each track

component to be mapped to a cell, it must traverse the cell sufficiently (relative to cell size L) and exit from a different edge than it enters.

Once all of the tracks are quantized into sets of states, a state transition model is constructed. From the track data, the count of Markovian transitions, $C\left(s_j|s_i\right)$, from state s_i to a neighboring 8-connected state s_j, is computed by counting the number of actual s_i-to-s_j track connections. States with insufficient statistics from real data (less than 5 transitions in or out for our experiments) are removed from the state space.

3.3. Entry/Exit Locations

We next probabilistically estimate the start states (entry locations) and terminal states (exit locations) of the scene from the tracks. Since we are using weak tracking (where fragmented tracks start and stop across the entire scene), typical algorithms for finding entry/exit locations such as those presented in [8, 12] will perform poorly. Our approach is specifically designed to handle weak tracking data. We start by constructing an image mask of where tracks *can* exist by counting the number of trajectories passing through each pixel in the image. A binary presence image is then created by thresholding low presence pixels. A median filter is used to remove salt-and-pepper noise and the remaining regions are dilated. Finally, the pixel coordinates defining the boundary of each substantial region are found. Ideally, tracks should truly begin and end at locations on these boundaries. We soften this constraint by allowing those states within a specified distance away from the boundary to be feasible entry or exit states (we use a Euclidean distance of 40 pixels in our experiments).

The entry weight $\mathcal{W}_E$ for a feasible entry state s_i is calculated as $\mathcal{W}_E = N_{start} \cdot \left(\max\left(0, 1 - {}^{N_{in}}/_{N_{start}}\right)\right)$, where N_{start} is the number of tracks that *start* in s_i and transition *out* to another state, and N_{in} is the number of tracks that transition *into* s_i. For an ideal entry state, $N_{in} \ll N_{start}$, while $N_{in} \gg N_{start}$ for any internal non-start state. The probability of starting in a state, $P_E\left(s_i\right)$, is calculated by normalizing by the sum of entry weights across all feasible start states.

The procedure for determining the exit weight for each state is analogous to determining the entry weight. Namely, the weighted number of stops $\mathcal{W}_X$ for a feasible exit state s_j is calculated as $\mathcal{W}_X = N_{stop} \cdot \left(\max\left(0, 1 - {}^{N_{out}}/_{N_{stop}}\right)\right)$, where N_{stop} is the number of tracks that transition *into* s_j and *stop*, and N_{out} is the number of tracks that transition *out* of s_j. After removing any state transitions accounting for less than 5% of the total transitions out of state s_i, we determine the probability of stopping at s_i as $P\left(s_j = \emptyset|s_i\right) = \mathcal{W}_X / \left(\mathcal{W}_X + \sum_l C\left(s_l|s_i\right)\right)$. If $P\left(s_j = \emptyset|s_i\right) < 0.5$ (more likely to transition than exit) we set $\mathcal{W}_X = P\left(s_j = \emptyset|s_i\right) = 0$. Finally, we calculate the probability of transitioning from s_i to s_j as $P\left(s_j|s_i\right) = C\left(s_j|s_i\right) / \left(\mathcal{W}_X + \sum_l C\left(s_l|s_i\right)\right)$.

The result is a probabilistic state model for entries, transitions, and exits. The model could be thresholded to make firm assignments, but we retain the softness for the pathlet identification process.

4. Pathlet Extraction

Our goal is to extract meaningful pathlets from the scene to represent detailed and localized path behaviors. If a similarity measure can be formed between pairs of states $\left(s_i, s_j\right)$, a natural representation of the relationship in our state space is an undirected graph $G = (V, E)$, where V and E are the sets of vertices and edges in the graph, respectively. The relationships between the set of vertices $V = \{v_1, v_2, \ldots, v_n\}$ can be defined by a similarity matrix W where $W_{ij} = W_{ji}$ and $W_{ij} \geq 0$. In our case, pathlets can be thought of as disjoint subgraphs, or clusters, of vertices where each vertex represents a state in our model.

4.1. State Similarity Matrix

A similarity metric between states should be high when the states belong to the same pathlet and low when they do not. Consequently, we define the similarity metric between two states (s_i, s_j) as

$$W_{ij} = \Psi\left(s_i, s_j\right) \cdot R\left(s_i, s_j\right) \tag{1}$$

which is a combination of component similarity weights for common origin/destination locations (Ψ) and temporal cross-correlation (R). Both of the similarity components have values between 0 and 1. As our probabilistic state model is a "local" representation (Markovian), we incorporate broader "non-local" similarity properties here to better capture the global context of the scene.

Origin/Destination Similarity Ψ
We compare the origin and destination state locations of tracks passing through two states as one measure of non-local similarity. Using probabilistic sampling of the state space model (entries, transitions, exits), we generate extended tracks (much longer than the tracks typically present in the original weak tracking data) and determine the starting/ending state locations for those tracks passing through each state. To sample a trajectory from the model, we probabilistically choose an entry state s_i based on sampling $P_E\left(\cdot\right)$ followed by repeatedly Monte Carlo sampling the transition model (inverse CDF method) until an exit state is selected. We do this repeatedly to populate the space with thousands of tracks. For each dataset in the experiment section the average track length of the sampled tracks was between 1.77 and 3.13 times longer than the average tracklet length received from the weak tracker.

The probability of a track through s_i *originating* from start/entry state s_j^E, $P_E\left(s_j^E \rightarrow s_i\right)$, is found by normalizing the number of sampled tracks passing through s_i originating from s_j^E by the total number of sampled tracks through s_i. To reduce the effects of multiple close-proximity entry states, this distribution is softened by smoothing across all entry states. This softened probability of a track through state s_i originating from an entry state s_j^E is calculated as

$$\hat{P}_E\left(s_j^E \rightarrow s_i\right) = \rho \sum_{l=1}^{k} P\left(s_l^E \rightarrow s_i\right) \cdot D\left(s_j^E, s_l^E\right) \cdot \Theta\left(s_j^E, s_l^E\right) \quad (2)$$

for all k feasible entry states, where $D(\cdot)$ and $\Theta(\cdot)$ are used to measure the proximity and orientation similarity of the two entry states (defined below) and ρ is a normalizing constant (the sum of $\hat{P}_E$ over all feasible entry states).

Letting ε_{ij} be the Euclidean distance between two states, the proximity weight is defined as $D\left(s_i, s_j\right) = \exp\left(-\alpha \cdot \varepsilon_{ij}\right)$ if $\varepsilon_{ij} \leq T_{dist}$ and 0 otherwise. We set $\alpha = 0.1$ in our experiments and vary T_{dist} (between 3 and 5) depending on the scene and cell size. The orientation weight $\Theta(s_i, s_j) = \max\left(0, \cos\left(\left|s_i\left(\theta\right) - s_j\left(\theta\right)\right|\right)\right)$ measures the difference in the local motion direction of two states, limiting the state connection to those traveling in a similar direction ($< 90^\circ$ difference).

A probabilistic start/entry distribution metric between s_i and s_j (measuring their similarity of start distributions) is then defined as

$$\Psi_E\left(s_i, s_j\right) = \sum_{l=1}^{k} \min\left(\hat{P}_E(s_l^E \rightarrow s_i), \hat{P}_E(s_l^E \rightarrow s_j)\right) \quad (3)$$

across all k possible start states. Conceptually, this metric represents the intersection of the two distributions. Note, other metrics (e.g., the Bhattacharyya coefficient or Hellinger distance) could also be used. To compute the probability of a track through s_i *terminating* at destination/exit state s_j^X, $\hat{P}_X\left(s_j^X \leftarrow s_i\right)$, an analogous procedure is performed. The similarity between exit location distributions of two states is given as $\Psi_X\left(s_i, s_j\right)$, computed as in Eqn. 3. Finally, we take the minimum of $\Psi_E\left(s_i, s_j\right)$ and $\Psi_X\left(s_i, s_j\right)$

$$\Psi(s_i, s_j) = \min\left(\Psi_E(s_i, s_j), \Psi_X(s_i, s_j)\right) \quad (4)$$

to measure the similarity of the origin/destination distribution between states.

Temporal Cross-Correlation Similarity R

We also measure the temporal similarity between states to separate paths that are spatially similar but traversed at different times/rates (e.g., road vs. sidewalk). We form the temporal trace ν for each state s_i where ν_k represents the number of real tracks (not sampled) mapped to the state during the k^{th} time interval. To handle the delay between distant states on a path, the temporal similarity R of states s_i and s_j is defined as the maximum cross-correlation $(xcorr\left(a(t), b(t)\right) = corr\left(a(t), b(t-\tau)\right))$ value of their temporal traces. We use a time interval duration of 5 sec and maximum time shift window of ± 20 sec in our experiments.

4.2. Spectral Clustering

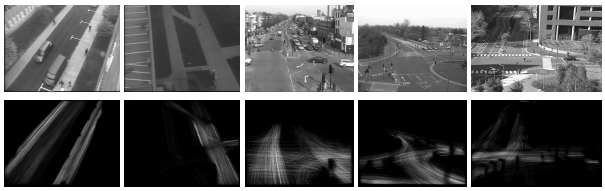
After the state similarity matrix W is constructed for the state space, the Spectral Clustering algorithm presented in [19] is used to automatically segment the states into pathlets. The normalized Laplacian of the similarity matrix W is used and defined as $\mathcal{L}_N = D^{-1/2} W D^{-1/2}$, where D is the diagonal degree matrix with $D_{ii} = \sum_{j=1}^{n} W_{ij}$. The algorithm uses $Z \in \mathbb{R}^{n \times c}$, a matrix of the largest c eigenvectors of $\mathcal{L}_N$ that are rotated to align with the canonical coordinate system, to define a quality metric for c clusters

$$q_c = 1 - \frac{\frac{1}{n}\left(\sum_{i=1}^{n}\sum_{j=1}^{c} \frac{Z_{ij}^2}{M_i}\right) - 1}{c} \quad (5)$$

where $M_i = \max_j Z_{ij}^2$. Starting at $c^* = \arg\max_c q$, the algorithm uses this quality metric to automatically determine the number of clusters $\bar{c} = \tilde{c} - 1$ where $\tilde{c}$ is the smallest value for c such that $q_{\tilde{c}} > q_{c^*} - \epsilon$ for some small value ϵ. Each data point (state) i is then assigned to cluster j where $j = \arg\max Z_{ij}^2$. See [19] for additional details.

5. Experimental Results

We tested our scene modeling and pathlet segmentation approach with five urban scenes of varying complexity. An image depicting each scene is shown in the top row of Fig. 2. Scenes I and II are taken from our camera network and the tracks were captured in real-time. The remaining video scenes were from published datasets (Scenes III and IV are from [5] and Scene V is from [16]). Scene I is a fairly simple scene consisting of two sidewalks and a one-way road. Scene II consists of a primary walkway with multiple secondary branching walkways creating 'T' and 'Y' splits and merges. Scenes III and V contain bi-directional traffic moving vertically and horizontally along with several pedestrians crossing the roads and walking on the adjacent sidewalks. Scene IV depicts a roundabout where traffic enters and exits from the left, right, and top of the image. Below each scene image (bottom row) we show a pixel-wise histogram (presence image) of the tracks collected from the weak tracker (brighter pixels correspond to more tracks present). The tracks were then quantized/mapped to the state space and the state transitions and probabilistic start/stop locations were computed.



Scene I Scene II Scene III Scene IV Scene V

Figure 2. Five urban scenes. Top row: scene image. Bottom row: presence image from raw track data.

Scene	Image Size	Video Length	# of Tracks	L	T_{dist}
I	640 × 480	15:00	7079	20	3
II	640 × 480	15:00	6779	20	3
III	360 × 288	3:12	5000	10	5
IV	360 × 288	7:38	5000	10	5
V	720 × 480	4:36	4778	20	3

Table 1. Image size, length of video (min:sec), and number of tracks extracted from the five scenes shown in Fig. 2 along with the cell size and proximity threshold used to extract pathlets via our proposed algorithm.

5.1. Comparative Results

We compare the results of our pathlet approach with two other methods. The first method is the feature-based approach of [5] which performs background subtraction to extract image events (objects), describes the image events via a 10-D combination of appearance and motion-based features, clusters the image events into atomic video events using K-means, groups the atomic video events into behavior labels via EM, and extracts semantic regions using Spectral Clustering on a similarity matrix built by combining the behavioral histogram and spatial differences between pixels. To modify their approach for weak tracking data, we describe image events by a 4-D feature vector $[x\ y\ dx\ dy]^T$ containing the spatial coordinates and instantaneous displacement of a tracked feature. Since no object bounding boxes are available, we populate the behavioral histograms (at each pixel) by centering a disk on each image event and updating the corresponding behavior count for all pixels within the inscribed neighborhood with a value ≤ 1 based on their proportional radial pixel distance from the disk's center (forming weighted histograms). To limit the clustering complexity on large images, we grid the resulting histogram image into 4x4 cells (much smaller than our grid size L) and represent each cell via the sum of the pixel histograms within the cell, removing low activity locations as described in [5]. Note that in [5] a second pass of the clustering algorithms is performed in each region to extract a final set of spatially overlapping Gaussian-shaped behavior labels that are used for scene analysis which are not used in this paper (we compare to the spatial region segmentation).

The second comparison method is the trajectory clustering method of [17] which performs Spectral Clustering on a pair-wise trajectory similarity matrix based on a modified Hausdorff distance of the tracks. As referenced in [17], the spatial density distribution of each cluster is thresholded to provide masks of each cluster region.

Unlike our approach, which has a fixed clustering complexity, the multiple clustering steps (K-means and EM) in [5] and the clustering of a pairwise track similarity matrix in [17] are both adversely affected by the number of tracks (considering that hundreds of thousands of weak tracks can be accumulated over one day of tracking). To give a fair algorithm comparison in spite of these limitations in the alternate approaches, we limit the number of tracks used in all three algorithms for each scene to the values shown in Table 1 (extremely small tracks were already removed from the datasets). For convenience, we will refer to our proposed approach as Method 1, the feature-based approach [5] as Method 2, and the trajectory-based approach [17] as Method 3 throughout the results section.

For each scene the same track dataset is used to generate the corresponding similarity matrices for each method. Spectral Clustering [19] is performed on each similarity matrix with $\epsilon = 0.006$ (Sec. 4.2) for all experiments. We additionally run a connected components algorithm to separate spatially disjoint clusters into multiple regions and remove all regions containing low presence ($<$ 1-2% coverage of the total scene presence image). Figure 3 shows the resulting pathlets from Method 1, semantic regions from Method 2, and trajectory clusters from Method 3 for all five scenes.

Qualitative Analysis

For Scene I, Methods 1 and 3 capture the bi-directionality of the sidewalks and the vehicular traffic down the road. Both algorithms also capture movement (cyclists) traveling the wrong way up the one-way road. Method 3 also generates a non-coherent cluster containing a few tracks traveling up the road and crossing the road. Method 2 is able to capture the main motion areas, but under-segments them into two regions by combining the left sidewalk with the road (we would want this algorithm to produce three regions for this scene). The blocky artifacts in the pathlets from Method 1 are a result of the grid technique and could be reduced with a smaller grid size.

In Scene II, the long walkway on the left was rarely traversed due to a temporary fence for construction and hence is only covered by a pathlet in one direction from Method 1 and is not covered by any semantic region using Method 2. Since Method 3 sometimes generates clusters containing just a few tracks, it captures the bi-directional nature of this sidewalk. Method 2 is the only algorithm which creates a separate region for the branch at the top of the image; however it splits the bottom section of the main walkway.

Ideally Method 1 would have generated a separate pathlet for the branch at the top of the image (second row). Unfortunately, the tracker fails frequently at the top of the scene (see presence image in Fig. 2) causing multiple probabilistic entry locations along the entire branch. These multiple entry locations result in a bleeding artifact in the origin/destination similarity Ψ, reducing the expected separation between the origin distribution of states on the branch with those on the main walkway.

For Scene III, Method 2 does well at extracting regions for the right half of the scene and the vertical traffic in the upper left portion of the scene, but over-segments the lower left portion of the scene into four regions instead of two. While Method 3 extracts clusters of vertical traffic, there are three clusters that group trajectories from distinct paths together. An occlusion (pole) causes both Methods 1 and 3 to model leftward traffic using two regions. Method 1 extracts pathlets for all four directions of traffic crossing the main intersection, a pathlet for the continuous turn lane entering the scene on the left from which vehicles travel up the scene, and a pathlet describing pedestrian crossings.

On Scene IV, Method 1 performs very well, extracting pathlets for traffic entering at the bottom and traveling both leftward and upward. Furthermore, there are two pathlets to describe traffic entering from the top and left, a pathlet for the area where traffic from these two pathlets merge, and a final pathlet for when traffic on this pathlet merges with traffic circling the roundabout. Both Methods 2 and 3 under-segment the scene, resulting in large regions containing several different motion patterns.

In Scene V, Method 1 is able to extract pathlets for traffic entering at the bottom of the image, traveling horizontally across the image, and traveling vertically at the top of the image, as well as pedestrians moving toward the right of the image. Method 3 extracts regions for pedestrians from a small number of tracks where the other two methods require behavior to be more consistent to be modeled. Method 3 fails to separate rightward trajectories from those turning left and traveling vertically up the scene. Method 2 over-segments all of the roadways.

Quantitative Analysis

In addition to the visual qualitative analysis showing the advantage of our pathlet approach, we use three metrics to quantitatively compare the results of the three algorithms and provide the results in Table 2.

The number of regions N_R used to describe the scene (first column of Table 2 for each method) provides a rough comparative idea if the scene is under or over-segmented. Values of N_R vary depending on the scene, but can be compared across algorithms. Due to the nature of the similarity matrix constructed in Method 2, the extracted regions will be non-overlapping (a limitation). Thus, Method 2 should result in fewer regions than Methods 1 and 3 for bi-directional areas. Methods 1 and 3 typically result in a similar number of regions N_R (i.e., ± 1). Looking at Fig. 3, Method 2 over-segments Scenes II, III, and V, and under-segments Scenes I and IV.

The scene presence coverage Ω (second column of Table 2 for each method) measures the percentage of the total presence from Fig. 2 (bottom row) encompassed by the extracted region masks. This metric could produce high values with regions which are qualitatively poor (e.g., a few undescriptive large regions containing several paths or several over-segmented regions which combine to cover a single path), but provides a measure for how much of the scene activity is being captured. Ideally this metric should be high, but not too high that rare traffic areas are modeled. Based on Ω, Methods 1 and 3 cover roughly the same amount of activity for Scenes I-III, while Method 1 covers more activity for Scene IV and Method 3 does so for Scene V. The regions from Method 2 cover the least amount activity in Scenes I and II, which is likely a result of removing low activity pixels, but have comparable coverage for the remaining scenes.

Given the viewpoint and structure of the scenes tested, we also desire the regions to contain one main segment with no branches (a single path). In our scenes there are few pathway regions with complex shapes. As a result, the horizontal and vertical signatures (row and column projections) of the region masks should be relatively uniform. We created histograms of the two signatures for each region using bins of width L and computed the median and maximum count of the non-zeros bins. The shape consistency of each region, S_C, is measured as the minimum of the two signature's median/maximum ratio. Thus, $S_C \in [0, 1]$ where more consistent regions have values closer to 1. The third column of Table 2 for each method shows the minimum and mean S_C value of the extracted regions.

While Methods 1 and 3 are comparable in the amount of activity they cover (Ω), the shape consistencies S_C of the two algorithms score much differently. The minimum S_C is always higher from Method 1 than it is from Method 3 (i.e., the most inconsistent region from Method 1 is always more consistent than the most inconsistent region from Method 3) and the mean S_C from Method 1 is higher than Method 3 in every scene but Scene I. The undesirable regions from Method 3 containing several different paths with slight overlaps (Scene I row 3, Scene III rows 2-4, and Scenes IV and V row 1 in Fig. 3) are the regions which receive the lowest values for S_C. While Method 2 produces consistent regions (without branches), it is obvious from Fig. 3 that the regions are either over-segmented (Scenes II, III, and V) or incorporate multiple motion patterns (Scenes I and IV).

	Method 1 (Proposed)			Method 2 [5]			Method 3 [17]		
Scene	N_R	Ω	$\mathcal{S}_C$	N_R	Ω	$\mathcal{S}_C$	N_R	Ω	$\mathcal{S}_C$
I	7	97.4	0.600/0.694	2	93.12	0.633/0.714	7	98.94	0.217/0.736
II	7	94.86	0.333/0.649	6	92.73	0.452/0.689	9	95.46	0.243/0.516
III	7	97.56	0.500/0.666	9	96.58	0.450/0.681	6	96.02	0.125/0.504
IV	6	96.27	0.500/0.662	5	96.73	0.600/0.689	6	89.16	0.369/0.579
V	9	87.5	0.370/0.698	15	95.46	0.375/0.635	8	95.74	0.192/0.525

Table 2. Number of regions N_R, scene activity coverage percentage Ω, and region shape consistency $\mathcal{S}_C$ (min/mean) for the five scenes shown in Fig. 3.

5.2. Discussion

In an overall sense, our pathlets are qualitatively and quantitatively more accurate at modeling motion patterns than the regions from the other two methods. Our approach to extract these pathlets (Method 1) could be considered a mid-level approach between the lower-level pixel/cell-based approach of Method 2 and the higher-level trajectory-based approach of Method 3. Since our method clusters states, it is not adversely affected by the number of tracks as are Methods 2 and 3. Furthermore, our method is capable of working well on weak tracking data. Unlike Method 3, which sometimes generates clusters containing few trajectories, our approach will not model and emphasize rarely used paths. Finally, the extracted pathlets from our method typically only contain one motion pattern to better enable unique mappings for instantaneous object location and velocity. These characteristics are not guaranteed with Methods 2 and 3.

6. Summary

We presented a novel framework to extract pathlets from weak tracking data. Tracks are mapped into a Markovian state space and entry/exit states are estimated. Pathlets are generated by applying Spectral Clustering to the state space. Finally, we compared our approach against a feature-based and trajectory-based approach using weak tracking data on five urban scenes and showed that our pathlets provide a better overall representation of the scene activity.

7. Acknowledgments

This research was supported in part by the US Air Force Research Laboratory Human Effectiveness Directorate (WPAFB) under contract No. FA8650-07-D-1220.

References

[1] A. Basharat, A. Gritai, and M. Shah. Learning object motion patterns for anomaly detection and improved object detection. In *Proc. CVPR*, 2008. 1, 2

[2] V. Bruce, P. Green, and M. Georgeson. *Visual Perception Physiology, Pyschology, and Ecology*. Psychology Press Ltd, 3rd edition, 1996. 2

[3] A. Fu, W. Hu, and T. Tan. Similarity based vehicle trajectory clustering and anomaly detection. In *Proc. ICIP*, 2005. 1, 2

[4] L. Kratz and K. Nishino. Anomaly detection in extremely corwded scenes using spatio-temporal motion pattern models. In *Proc. CVPR*, 2009. 1, 2

[5] J. Li, S. Gong, and T. Xiang. Scene segmentation for behaviour correlation. In *In Proc. ECCV*, 2008. 1, 2, 4, 5, 7, 8

[6] C. Loy, T. Xiang, and S. Gong. Multi-camera activity correlation analysis. In *Proc. CVPR*, 2009. 1, 2

[7] D. Makris and T. Ellis. Path detection in video surveillance. *Image and Vision Computing*, 20:895–903, 2002. 1, 2

[8] D. Makris and T. Ellis. Automatic learning of an activity-based semantic scene model. In *Proc. AVSS*, 2003. 1, 2, 3

[9] I. Pop, M. Scuturici, and S. Miguet. Common motion map based on codebooks. In *Int. Sym. Vis. Comp.*, 2009. 1, 2

[10] I. Saleemi, K. Shafique, and M. Shah. Probabilistic modeling of scene dynamics for applications in visual surveillance. *IEEE TPAMI*, 31(8):1472–1484, 2009. 1, 2

[11] J. Shi and C. Tomasi. Good features to track. In *Proc. CVPR*, 1994. 1, 2

[12] C. Stauffer. Estimating tracking sources and sinks. In *Proc. Second IEEE Event Mining Workshop*, 2003. 3

[13] C. Stauffer and W. Grimson. Learning patterns of activity using real-time tracking. *IEEE TPAMI*, 22(8):747–767, 2000. 1, 2

[14] E. Swears, A. Hoogs, and A. Perera. Learning motion patterns in surveillance video using hmm clustering. In *Proc. WMVC*, 2008. 1, 2

[15] X. Wang, K. Ma, W. Ng, and W. Grimson. Trajectory analysis and semantic region modeling using a nonparametric bayesian model. In *Proc. CVPR*, 2008. 1, 2

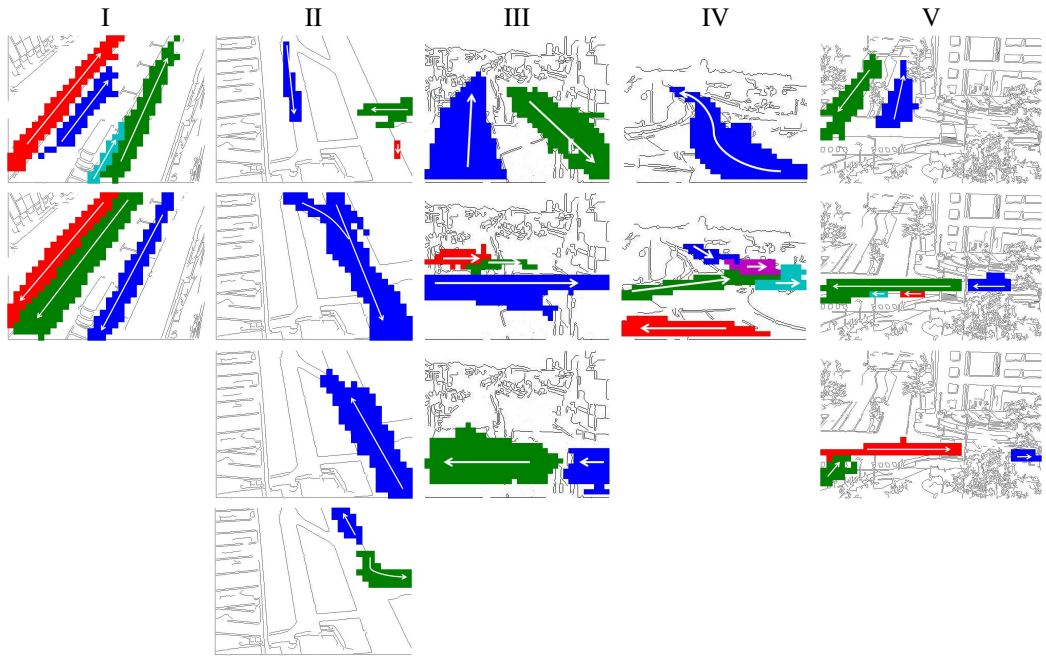
[16] X. Wang, X. Ma, and E. Grimson. Unsupervised activity perception in crowded and complicated scenes using hierarchical bayesian models. *IEEE TPAMI*, 31(3):539–555, 2009. 1, 2, 4

[17] X. Wang, K. Tieu, and W. Grimson. Learning semantic scene models by trajectory analysis. In *Proc. ECCV*, 2006. 1, 2, 5, 7, 8

[18] Y. Yang, J. Liu, and M. Shah. Video scene understanding using multi-scale analysis. In *Proc. ICCV*, 2009. 1, 2

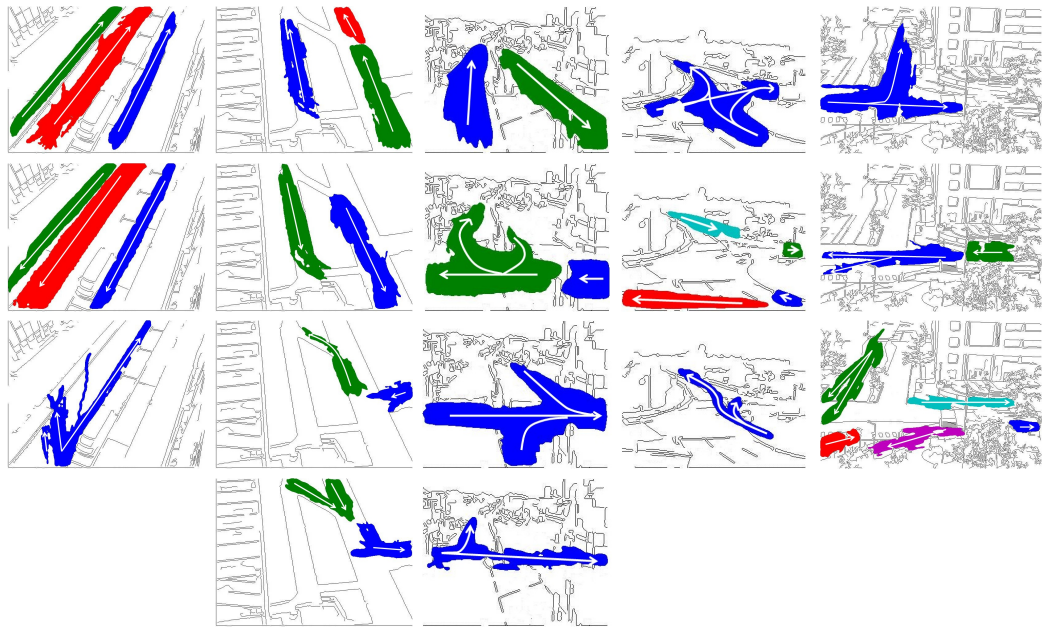
[19] L. Zelnik-Manor and P. Perona. Self-tuning spectral clustering. In *NIPS*, 2004. 1, 4, 5

[20] T. Zhang, H. Lu, and S. Li. Learning semantic scene models by object classification and trajectory clustering. In *Proc. CVPR*, 2009. 1, 2



Pathlets from Method 1 (Proposed)

Semantic regions from Method 2 [5]



Trajectory clusters from Method 3 [17]

Figure 3. Extracted pathlets, semantic regions using approach based on [5], and trajectory clusters using [17] for five urban scenes. (Best viewed in color.)