Exploiting Multiple Cameras for Environmental Pathlets*

Kevin Streib and 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 multi-camera framework to extract reliable pathlets [1] from tracking data. The proposed approach weights tracks based on their spatial and orientation similarity to simultaneous tracks observed in other camera views. The weighted tracks are used to build a Markovian state space of the environment and Spectral Clustering is employed to extract pathlets from a state-wise similarity matrix. We present experimental results on five multi-camera datasets collected under varying weather conditions and compare with pathlets extracted from individual camera views and three other multi-camera algorithms.

1 Introduction

An important task in video surveillance is to observe/model an environment and extract behaviorial trends. This is typically done by collecting data for extended periods of time and extracting pathway regions (trajectory clusters [2–4], semantic regions [5, 6], pathlets [1]) using either trajectory or feature-based approaches. Generally these algorithms exploit data from single cameras and the extracted regions typically correspond to locations in the image where motion occurs. If these regions were projected to an orthophoto (i.e., aerial top-down image as seen in Google maps), locations corresponding to non-ground plane motion (e.g., head of a pedestrian) would project to locations other than the ground plane area where the object was actually moving. Thus, many of the extracted regions describe only 2D sensor-view patterns. What is really desired are pathway regions corresponding to ground-plane areas of traffic.

One solution to the erroneous region projection could be to use tracks solely from the area of objects in contact with the ground. However, finding such locations is generally not straightforward, nor would it be feasible to perform in real-time with environments containing large amounts of moving objects. As many surveillance systems contain multiple cameras viewing the same area from different vantage points, we exploit the data collected from multiple cameras to generate refined pathway regions to better capture the actual pathway regions. We choose to map the cameras to an orthophoto, instead of to a single camera view, because they do not suffer from projective distortions which causes image pixels to represent varying amounts of spatial data.

The camera network we utilize encompasses a crowded urban environment where receiving a single long track per object for its duration through the scene is infeasible in real-time. Consequently, we employ the Kanade-Lucas-Tomasi (KLT) tracker

* Appears in International Symposium on Visual Computing, November 2010.

[7], which results in multiple fragmented tracks per object but is capable of tracking hundreds of features simultaneously in real-time. A pathway analysis method suited to this type of tracking data is the approach of [1], which extracts "pathlets" (coherent motion regions containing tracks with similar origin and destination) of a scene from KLT tracks. In this paper, we present a novel extension to [1] which combines information from multiple cameras to extract the pathlets. The approach is based on using tracks from other cameras to vote for or remove tracks from a given camera view. We evaluate our proposed approach on multiple datasets, and compare the results with pathlets extracted using the tracking data from the individual camera views and three other multi-camera approaches suited to the task.

2 Related Work

The majority of surveillance-related research on modeling scene behavior has focused on using a single camera. These works often use either trajectory [1–4] or feature-based [5, 6] approaches to extract trajectory clusters or semantic motion regions of a scene. Trajectory methods have included envelope approaches to determine if tracks should be assigned to existing routes or formed into new routes [3], vector quantization to reduce trajectories to a set of prototypes [2], and Spectral Clustering on pairwise trajectory similarity matrices [4]. Alternatively, optical flow [6] and combinations of optical flow and appearance metrics [5] can be used to analyze scene activity without relying on tracking.

Recently, research has begun focusing on utilizing entire camera networks rather than single cameras. Data is fused from 2D image and 3D world coordinates in [8] to track objects between cameras. In [9] a camera network topology is estimated and targets are tracked across blind areas within the camera network by associating activities across camera views. In [10] objects are tracked in partially overlapping cameras, and the extracted features are mapped to a ground plane and associated across views to track targets through the camera network. Information is combined across all cameras in [11] and the ground plane location corresponding to feet are found to track people in crowded scenes. In [12, 13] correlation of activity regions from different cameras are modeled and used to detect global activity anomalies. Features are extracted from objects tracked independently in each camera view in [14] and the distribution of activities across camera view feature spaces are learned to group trajectories belonging to the same activity and model paths taken across camera views.

While several methods have been proposed to utilize multiple cameras to either track objects for extended durations or correlate activities across camera views, to the best of our knowledge the work presented in this paper is the first approach which exploits data from multiple cameras to extract meaningful pathway regions that more accurately describe where objects are moving than regions derived using a single camera.

3 Multi-Camera Pathlet Extraction

We based our approach on the pathlet method of [1], that is specifically designed to handle weak tracking data. The algorithm in [1] overlays an L x L grid onto a scene and quantizes tracks into states, where each state $s_i = [(x, y), \theta]$ is defined as a grid cell location and the quantized angle of the track through the cell. Once quantized, the tracks are used to count the transitions from each state to states in their 8-connected neighboring cells. These counts are then normalized to generate a Markovian state transition model for each state. Next, the ratio of number of tracks that enter/leave a state versus start/stop at the state are used to find the entry/exit probabilities for each state close to the border of regions where tracks exist. The result is a probabilistic scene model of where tracks enter the scene, how they transition throughout the scene, and where they exit the scene. This model is employed to sample tracks (longer than the original weak tracks) which are used to determine where tracks through each state typically originate and terminate. Finally, the original weak tracks are used to produce a trace of the count of tracks in each state across time. A state-wise similarity matrix is constructed based on the origin/destination and temporal cross-correlation of tracks through the states, and the Spectral Clustering algorithm of [15] is employed to extract pathlets from the scene. See [1] for additional details

We extended the algorithm in three ways to enable multi-camera fusion. First, the Markovian transitions between states are adapted to handle confidence weights assigned to the tracks to enable scene modeling using tracks with varying believability. Second, we removed the constraint that entry and exit states be close to the border of regions where tracks exist to eliminate the necessity of finding the border around motion regions, and instead use the sum of track weights entering a state (N_{in}) and exiting a state (N_{out}) to calculate the entry and exit weights ($\mathcal{W}_E$ and $\mathcal{W}_X$) for *any* state as

$$\mathcal{W}_E = \frac{N_{out}}{\left(1 + \exp\left(-\frac{\frac{N_{out}}{N_{out}+N_{in}} - \mu}{\sigma}\right)\right)}, \quad \mathcal{W}_X = \frac{N_{in}}{\left(1 + \exp\left(-\frac{\frac{N_{in}}{N_{out}+N_{in}} - \mu}{\sigma}\right)\right)}, \quad (1)$$

where μ and σ are the mean and standard deviation of a CDF of a logistic distribution (set to 0.75 and 0.05 in our experiments), respectively. We use the above formulation as it is a more flexible method than presented in [1]. Finally, since the temporal cross-correlation used in [1] was empirically found to have little affect on the extracted pathlets, we define the state similarity matrix using solely the origin and destination similarity of tracks through the states.

Our proposed multi-camera method projects the tracks from all cameras to an orthophoto (via homography) and computes a weight for each track based on its spatial and orientation similarity to temporally overlapping tracks from the remaining cameras. The intuition for this approach can be explained by considering the ideal case of tracking an individual with two cameras that are 180° apart and registered to the orthophoto with a planar homography. Intuitively, KLT tracks from the two cameras corresponding to features close to the feet of the person will result in projected tracks on the ground plane location [11]. Feature points near the head of the person will project incorrectly (they violate the planar assumption). Thus, if projected tracks in one camera view are

close in proximity and travel in the same direction as temporally overlapping tracks from other cameras, they are more likely to correspond to tracks closer to the ground, thereby more accurately describing the true paths in the environment.

The procedure for determining $w\left(t_i^1\right)$, the weight of track i in Camera-1 from all other cameras, is as follows. First, $\alpha\left(t_i^1, t_j^2\right)$, the pairwise track weight for each temporally overlapping track t_j^2 in Camera-2, is computed as

$$\alpha\left(t_i^1, t_j^2\right) = \frac{1}{\tau_2 - \tau_1 + 1} \sum_{k=\tau_1}^{\tau_2} v\left(t_i^1[k], t_j^2[k]\right) \cdot \exp\left(-\frac{\|t_i^1[k] - t_j^2[k]\|}{\sigma}\right), \quad (2)$$

within the temporal overlap τ_1 to τ_2, and where the binary orientation similarity

$$v\left(a, b\right) = \begin{cases} 1 & |\angle(a) - \angle(b)| \leq 15^\circ \\ 0 & else \end{cases} \quad (3)$$

considers two tracks to be traveling in the same direction at time k if the direction of their instantaneous velocities are within 15°. We use a value of $\sigma = 4$ for the exponential weighting in Eq. 2 in all of our experiments.

Next, we use a greedy algorithm to determine $\beta_2\left(t_i^1\right)$, the weight of t_i^1 from *all* tracks in Camera-2. First, all elements in an indicator vector x used to keep track of matched observations in t_i^1 are initialized to zero. Then, while there are unmatched observations $\left(\text{to } t_i^1\right)$ and unmatched tracks from Camera-2 that have temporally co-occurring observations, $\beta_2\left(t_i^1\right)$ is incremented by $\frac{m}{M} \cdot \alpha\left(t_i^1, t_{jj}^2\right)$, where t_{jj}^2 is the unmatched track from Camera-2 resulting in the maximum track weight $\alpha\left(t_i^1, t_{jj}^2\right)$, m is the number of matched observations between the two tracks (i.e., temporally co-occurring and previously unmatched in x), and M is number of observations in t_i^1. The indicator vector x is then updated to reflect the observations that were just matched. Thus, $\beta_2\left(t_i^1\right)$ is a weighted average of the tracks from Camera-2 that get matched to t_i^1 based on the number of elements they match to in x.

The final weight of track i in Camera-1, $w\left(t_i^1\right)$, is calculated as the product of the camera weights for t_i^1 across all cameras

$$w\left(t_i^1\right) = \prod_{z \neq 1} \beta_z\left(t_i^1\right). \quad (4)$$

Thus, the weight of a track will increase as the support from tracks in other cameras increases.

4 Experimental Results

We tested the proposed multi-camera pathlet extraction method on four two-camera datasets and one three-camera dataset from the camera locations shown in Fig. 1(a), whose corresponding camera views are shown in Fig. 1(c). The three-camera dataset, Dataset V, is composed of track data from the three camera views used in Datasets II-IV. Track information was collected on different days with varying weather conditions.

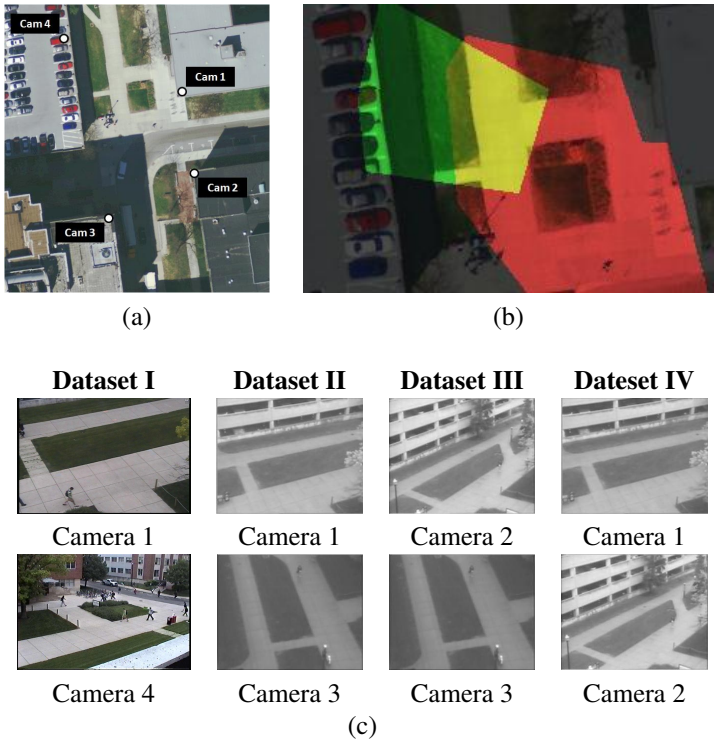

Fig. 1. (a) Camera locations of four cameras used for the datasets. (b) Individual and overlapping coverage of Cameras 1 and 4. (c) Camera views for Datasets I-IV. (Best viewed in color.)

Dataset I consists of tracks collected from Cameras 1 and 4 for approximately fifteen minutes on a sunny day. Datasets II-V consist of tracks collected from Cameras 1, 2, and 3 for six hours on a rainy day.

4.1 Alternate Approaches

We present three alternate methods in addition to our proposed approach to extract pathlets using data from multiple cameras and compare the results. In all of the methods, projections between the camera and ortho-space are performed using a planar homography [16] (i.e., a plane-to-plane mapping) from manual correspondence, which results in a projection matrix to transfer between coordinate systems. Using the homography, the camera views are mapped to a common orthophoto for each dataset to determine the overlapping ground region.

The first method (Method A) is a naive approach which projects the tracks from all cameras to the orthophoto. Tracks are given equal unitary weight and pathlets are

extracted from a state space model built in the ortho-space. This method is the same as the proposed method but with no track weighting scheme.

The second method (Method B) projects the tracks from each camera separately to the orthophoto (again assigning each track with a unitary weight), and extracts pathlets using the tracking data from each individual camera alone. The pathlets extracted from each individual camera are then intersected together, resulting in a set of combined pathlets. The intersection algorithm generates a label matrix (number of states $\times$ number of cameras) where each element $[i, j]$ corresponds to the pathlet ID that state i belongs to from the pathlets extracted using the tracking data from camera j. Each combined pathlet consists of all states having the same label combination across the individual camera pathlets, where a state must be labeled in a pathlet from each camera to persist. Thus, the number of combined pathlets is equivalent to the number of unique rows of the label matrix containing no zeros.

The third method (Method C) extracts the pathlets for the tracking data from each camera (assuming unitary track weight) in the camera-space and projects the resulting pathlets from each individual view to the orthophoto. The pathlet projection algorithm maps each state pixel/angle combination in the camera-space to a pixel/angle combination in the ortho-space. The corresponding ortho-space pathlets consist of all states whose respective cells are at least 50% populated. The ortho-space pathlets are computed using an intersection algorithm similar to the algorithm used in Method B. To account for slight projection deformations, the intersection algorithm is relaxed by allowing combinations of states from ortho-space pathlets from different cameras if their orientations are within 45°.

Our proposed approach (Sect. 3) will be referred to as Method D.

4.2 Optimal Pathlet Selection

To provide the maximal results for the different approaches, we first define the region-of-interest (ROI) in the orthophoto. For our datasets (Fig. 1(c)) we define the sidewalks as the ROIs as these should be the only locations with moving targets. We then overlay the grid used to build the state space model (Sect. 3) and define grid cells as positive if more than 50% of its pixels are within the ROI in the overlapping camera region and negative otherwise. For Method C the boundaries of the overlapping camera region and ROI are mapped back to the individual camera views, where positive and negative cells are defined based on the respective grid sizes used for each camera view.

Since it is feasible for states to contain insufficient information from training to build an accurate Markovian transition model, we remove states with low weight (where a state's weight is equivalent to the sum of the track weights for the tracks that are mapped to it) from the state space model. First, we sort the states in descending order of weight and generate a sorted cumulative distribution function (CDF). Again, our goal is to generate pathlets that describe the environment (i.e., walkways), rather than the scene from the individual camera view. Thus, we want to include states from positive cells and remove states from negative cells. After state removal, let T_p, F_p, and F_n be the number of positive cells with states, the number of negative cells with states, and the number of positive cells without states, respectively. Then, precision ($P = {}^{T_p}/_{(T_p + F_p)}$) measures

the percentage of kept cells that belong to the ROI, while recall ($R = {}^{T_p}/_{(T_p + F_n)}$) measures the percentage of cells belonging to the ROI which are kept. Since precision can be high with a low recall and vice-versa, we use the F-measure $\left(F = 2 \cdot \frac{P \cdot R}{P+R}\right)$, which combines precision and recall into a single metric that is maximized when precision and recall are jointly maximized, to determine the quality of the kept states. To optimize the pathlets (for each method), we find the F-measure resulting from the highest weighted states kept if the sorted CDF is thresholded at $X\%$, and keep the states based on the threshold resulting in the maximum F-measure. Thus, the threshold is designed to maximize the states kept from positive cells while minimizing states kept from negative cells.

The output of each multi-camera fusion algorithm is a set of pathlets describing the environment in the ortho-space. We then project the ortho-space pathlets back to the original camera views. We overlay the individual camera view grid on each projected pathlet and determine its spatial extent in the camera view grid space by only keeping cells which are at least 50% covered by the projected cluster. The camera view pathlets are run through a connected components algorithm, separating disjoint pathlets into a set of connected pathlets, and small pathlets (those containing less than 5 cells) are removed, yielding the final set of camera view pathlets. We use a 10×10 pixel grid for all cameras and datasets except for Camera-4 in Dataset I (where we use a 20×20 pixel grid because of the zoom factor). A 5×5 pixel grid is used in the ortho view for all datasets.

4.3 Results

We use precision, recall, F-measure, and N_p (the number of pathlets containing at least 50% of their cells within the overlapping camera region) as the primary means to quantitatively compare the pathlets extracted from the different approaches. Intuitively, N_p provides a rough comparison to determine if an environment is relatively under- or over-segmented. Tables 1 and 2 show the four quantitative measurements corresponding to the threshold which optimizes the F-measure of the pathlet extraction methods (as described in Sect. 4.2) for each dataset.

Table 1. Quantitative measurements of pathlets extracted using tracking data from individual camera views on five datasets. (Cam-x / Cam-y / Cam-z)

Dataset	N_p	Precision	Recall	F-Measure
I	2 / 6	0.82 / 0.82	0.96 / 0.93	0.88 / 0.87
II	9 / 7	0.64 / 0.75	0.97 / 0.85	0.77 / 0.80
III	8 / 14	0.69 / 0.76	0.85 / 0.92	0.77 / 0.83
IV	9 / 4	0.63 / 0.65	0.96 / 0.86	0.76 / 0.74
V	10 / 6 / 6	0.64 / 0.66 / 0.75	0.96 / 0.85 / 0.91	0.77 / 0.75 / 0.82

Table 1 shows that recall is typically much higher than precision for pathlets extracted from each individual camera. This suggests that the pathlets extracted from a

Table 2. Quantitative measurements of pathlets extracted using four multi-camera fusion algorithms on five datasets. Bold font represents the highest F-measure received in each camera for the dataset. (Cam-x / Cam-y / Cam-z)

Dataset	Method	N_p	Precision	Recall	F-Measure
I	A	8 / 12	0.80 / 0.87	0.91 / 0.90	0.85 / 0.89
	B	6 / 7	0.95 / 0.97	0.75 / 0.72	0.83 / 0.82
	C	8 / 8	0.88 / 0.94	0.87 / 0.85	**0.88** / 0.89
	D	8 / 8	0.82 / 0.90	0.93 / 0.91	0.87 / **0.91**
II	A	4 / 4	0.84 / 0.74	0.72 / 0.70	0.77 / 0.72
	B	6 / 5	0.93 / 0.84	0.73 / 0.71	0.82 / 0.77
	C	8 / 7	0.92 / 0.85	0.77 / 0.76	0.84 / 0.80
	D	4 / 5	0.89 / 0.77	0.85 / 0.86	**0.87** / **0.81**
III	A	5 / 6	0.78 / 0.70	0.83 / 0.84	0.80 / 0.76
	B	8 / 8	0.89 / 0.85	0.69 / 0.70	0.78 / 0.77
	C	12 / 13	0.93 / 0.86	0.75 / 0.73	0.83 / 0.79
	D	8 / 9	0.86 / 0.79	0.91 / 0.91	**0.89** / **0.85**
IV	A	5 / 4	0.64 / 0.59	0.94 / 0.92	0.77 / 0.72
	B	4 / 4	0.70 / 0.65	0.74 / 0.72	0.72 / 0.68
	C	10 /7	0.73 / 0.65	0.84 / 0.80	**0.78** / 0.72
	D	5 / 4	0.74 / 0.69	0.81 / 0.79	0.77 / **0.74**
V	A	5 / 5 / 5	0.67 / 0.62 / 0.58	0.95 / 0.92 / 0.94	0.78 / 0.74 / 0.71
	B	5 / 4 / 6	0.93 / 0.90 / 0.83	0.70 / 0.67 / 0.73	0.80 / 0.77 / 0.77
	C	18 / 11 / 14	0.97 / 0.96 / 0.87	0.73 / 0.71 / 0.69	0.83 / 0.81 / 0.77
	D	6 / 5 / 5	0.85 / 0.82 / 0.76	0.89 / 0.87 / 0.86	**0.87** / **0.84** / **0.81**

single camera cover a majority of the ROI, but also bleed outside the ROI, resulting in non-walkway pathlets.

Based on Table 2, Method A generally has the least precision of the multi-camera fusion methods, which is expected given the naive nature of the algorithm. Furthermore, the method typically results in a higher recall than precision, meaning it tends to generate pathlets which cover, yet bleed outside the ROI.

Methods B and C generally result in the highest precision of the multi-camera fusion algorithms. This result is expected since both methods use the intersection algorithm described in Sect. 4.1. Consider an ideal case where two camera views are 180° apart focusing on a single sidewalk which lies between the cameras (e.g., similar to Dataset I in Fig. 1(b)). In this scenario tracks from one camera will cover the sidewalk and overlap to the right of the sidewalk, while tracks from the other camera will cover the sidewalk and overlap to the left of the sidewalk. Thus, the intersection of the corresponding pathlets will lie directly on the sidewalk resulting in a very high precision. However, since states must have been kept in all camera views to be present in the final combined pathlets, the intersection algorithm used in Methods B and C also generally causes the recall to be lower than in Methods A and D.

Method D, our proposed method, tends to balance precision and recall more than the other multi-camera fusion methods. Furthermore, it results in the highest average F-measure for all five datasets and the highest F-measure in all the datasets' individual

camera views except for two cases, where it is only 0.01 less than best the F-measure received from Method C. However, in most cases Method D requires less pathlets to model the environment than Method C (see N_p in Table 2), which suggests that Method C is prone to generating over-segmented pathlets.

Qualitative Analysis

Figure 2 shows the pathlets extracted for Dataset I using the KLT tracking data from each individual camera and from the multi-camera fusion methods using the highest weighted states resulting in the optimized F-measure (as described above). In each image the white area, black lines, and colored blobs correspond to the overlapping camera region, ROI outline, and extracted pathlets, respectively.

The pathlets extracted using the tracking data from both individual cameras capture the bi-directionality of the main sidewalk inside the overlapping camera region. However, tracks on the upper body of pedestrians cause the pathlets to extend beyond the sidewalk and outside the ROI. Furthermore, there are additional pathlets in both camera views outside the overlapping camera region which extend beyond the camera's ROI due to the projection of far-field data.

Method A combines the track data from the two individual cameras after they are projected to a common ortho-space. As shown in Fig. 1(b), a majority of the overlap from Cameras 1 and 4 occurs within the ROI. Consequently, the states in these areas will contain more tracks, resulting in a higher contribution to the total state weight. For this particular scenario this artifact of the camera overlap removes a majority of the false positive cells when optimizing the F-measure, resulting in pathlets that are primarily inside the ROI for the overlapping camera region. However, Fig. 2 clearly shows that Method A over-segments the scene, using multiple pathlets to describe traffic traveling in the same direction.

The intersection algorithm used in Method B causes the primary sidewalk to be over-segmented in both camera views. Interestingly, this over-segmentation does not occur in Method C which also uses an intersection algorithm. This is likely a result of the relaxation described in Sect. 4.1 to deal with projection deformations. Method C also produces pathlets for traffic entering/exiting the main sidewalk from/to an adjoining sidewalk, which were only extracted from Camera-4 but kept because of the intersection algorithms.

Method D (our proposed approach) correctly yields one distinct pathlet to define each direction of the main sidewalk, and also extracts pathlets for traffic entering/exiting from/to the adjoining sidewalk.

We focus on Methods C and D from now on since they are both quantitatively and qualitatively superior to the other approaches. Figures 3 and 4 show the pathlets extracted from Datasets II-IV for Methods C and D (our proposed method), respectively.

In Dataset II both algorithms extract pathlets to capture the bi-directionality of the two primary sidewalks. Furthermore, Method D captures movement on the less frequently traveled secondary sidewalk which connects the two primary sidewalks, which Method C fails to do. In Dataset III, Method C severly over-segments the scene while Method D again captures the bi-directionality of the primary sidewalks with single path-

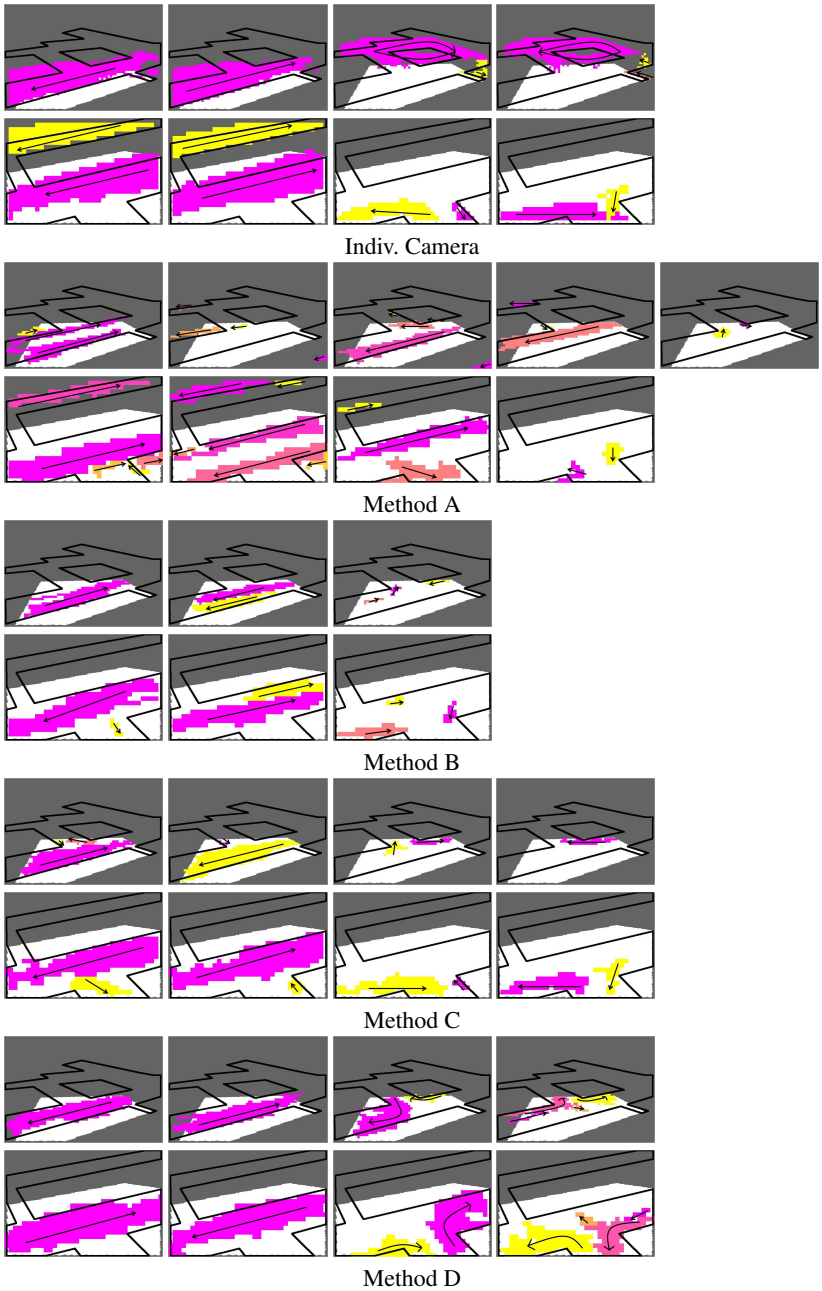

Fig. 2. Extracted pathlets using tracking data from the individual cameras and the four multi-camera information fusion algorithms for Dataset I. (Best viewed in color.)

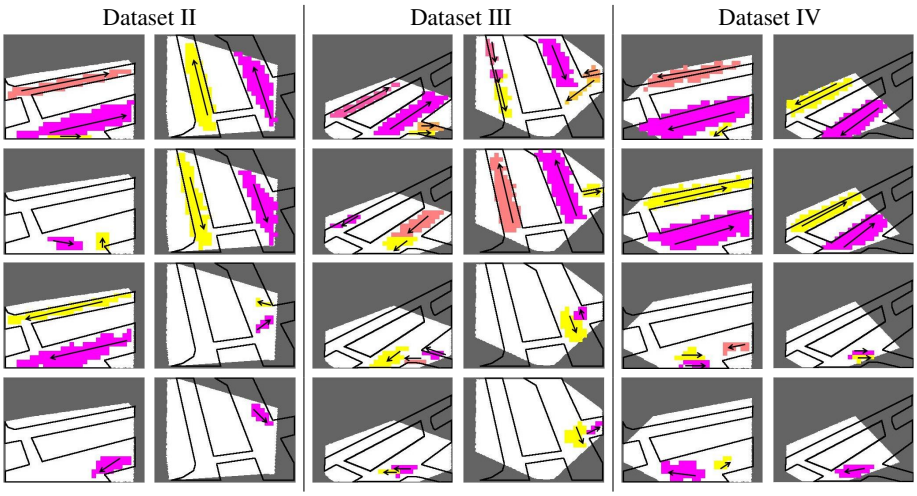

Fig. 3. Extracted pathlets using Method C for Datasets II-IV. (Best viewed in color.)

lets in each direction. Furthermore, Method D also captures a pathlet describing the connecting sidewalk and pathlets describing traffic entering and exiting from the 'T' junction. In Dataset IV both methods capture the bi-directionality of the primary walkways with Method C generating more pathlets describing tracks entering and exiting the 'T' junction. In Dataset V (not shown due to space constraints) Method D outperforms Method C which drastically over-segments the environment (see N_p in Table 2). Combining the quantitative measurements in Table 2 with the tendency Method C has for over-segmenting an environment, it is apparent that our proposed approach (Method D) performs the best at extracting environmental pathlets for the datasets.

5 Summary

We presented a novel approach to employ data from multiple cameras to generate pathlets. Our proposed approach weights tracks based on their spatial and orientation similarity to tracks collected simultaneously in other cameras. The weighted tracks are used to build a Markvoian state space model and Spectral Clustering is utilized to extract pathlets from a state-wise similarity matrix based on the origin/destination of tracks through the states. We compared our approach with pathlets extracted from the individual camera views and from three other multi-camera algorithms on five multi-camera datasets collected under varying conditions. Finally, we showed quantitatively and qualitatively that our proposed method outperforms the other methods. This research was supported in part by the US Air Force Research Laboratory Human Effectiveness Directorate (WPAFB) under contract No. FA8650-07-D-1220.

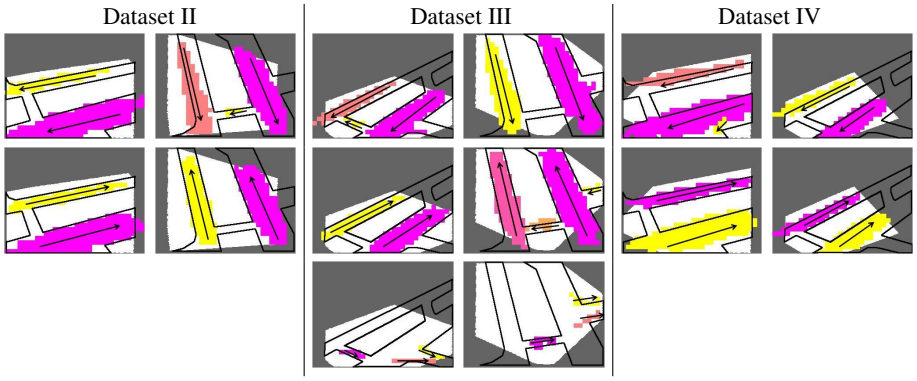

Fig. 4. Extracted pathlets using our proposed method (Method D) for Datasets II-IV. (Best viewed in color.)

References

1. Streib, K., Davis, J.W.: Extracting pathlets from weak tracking data. In: Proc. AVSS. (2010)
2. Stauffer, C., Grimson, W.E.L: Learning patterns of activity using real-time tracking. IEEE TPAMI **22** (2000) 747–767
3. Makris, D., Ellis, T.: Automatic learning of an activity-based semantic scene model. In: Proc. AVSS. (2003)
4. Wang, X., Tieu, K., Grimson, W.E.L.: Learning semantic scene models by trajectory analysis. In: Proc. ECCV. (2006)
5. Li, J., Gong, S., Xiang, T.: Scene segmentation for behaviour correlation. In: Proc. ECCV. (2008)
6. Wang, X., Ma, X., Grimson, E.: Unsupervised activity perception in crowded and complicated scenes using hierarchical bayesian models. IEEE TPAMI **31** (2009) 539–555
7. Shi, J., Tomasi, C.: Good features to track. In: Proc. CVPR. (1994)
8. Black, J., Ellis, T., Rosin, P.: Multi view image surveillance and tracking. In: IEEE Workshop on Motion and Video Computing. (2002)
9. Makris, D., Ellis, T., Black, J.: Bridging the gaps between cameras. In: Proc. CVPR. (2004)
10. Anjum, N., Cavallaro, A.: Trajectory association and fusion across partially overlapping cameras. In: Proc. AVSS. (2009)
11. Khan, S.M., Shah, M.: A multiview approach to tracking people in crowded scenes using a planar homography constraint. In: Proc. ECCV. (2006)
12. Loy, C.C., Xiang, T., Gong, S.: Time-delayed correlation analysis for multi-camera activity understanding. Intnl. Journal of Computer Vision (2010)
13. Li, J., Gong, S., Xiang, T.: Discovering multi-camera behaviour correlations for on-the-fly global activity prediction and anomaly detection. In: IEEE Intnl. Workshp on Visual Surveillance. (2009)
14. Wang, X., Tieu, K., Grimson, E.: Correspondence-free activity analysis and scene modeling in multiple camera views. IEEE TPAMI **32** (2010) 56–71
15. Zelnik-Manor, L., Perona, P.: Self-tuning spectral clustering. In: NIPS. (2004)
16. Criminisi, A., R.I., Zisserman, A.: A plane measuring device. Image and Vision Computing **17** (1999) 625–634