

Multi Feature Path Modeling for Video Surveillance

Imran N. Junejo, Omar Javed and Mubarak Shah
{ ijunejo, ojaved, shah }@cs.ucf.edu
University of Central Florida, Orlando, FL 32816, USA.

Abstract

This paper proposes a novel method for detecting nonconforming trajectories of objects as they pass through a scene. Existing methods mostly use spatial features to solve this problem. Using only spatial information is not adequate; we need to take into consideration velocity and curvature information of a trajectory along with the spatial information for an elegant solution. Our method has the ability to distinguish between objects traversing spatially dissimilar paths, or objects traversing spatially proximal paths but having different spatio-temporal characteristics. The method consists of a path building training phase and a testing phase. During the training phase, we use graph-cuts for clustering the trajectories, where the Hausdorff distance metric is used to calculate the edge weights. Each cluster represents a path. An envelope boundary and an average trajectory are computed for each path. During the testing phase we use three features for trajectory matching in a hierarchical fashion. The first feature measures the spatial similarity while the second feature compares the velocity characteristics of trajectories. Finally, the curvature features capture discontinuities in velocity, acceleration, and position of the trajectory. We use real-world pedestrian sequences to demonstrate the practicality of our method.

1. Introduction

Tracking objects is of primary interest for many applications such as surveillance, action monitoring, and path detection. Our goal is to learn the routes or paths most commonly taken by objects as they traverse through a scene and register any unusual activity. An example of an unusual behavior might be a person walking in a region not used by most people, a car following a zigzag path, or a person running in a region

where most people simply walk. A *path* is any established line of travel or access, and a *trajectory* can be defined as a path followed by an object moving through the space. Most objects follow a common trajectory while entering or exiting a scene due to presence of pavements, benches, or designated pathways. Therefore, a method is required that can model the usual trajectories of the object and indicate atypical trajectories that might call for further investigation through any higher level event recognition. We concern ourselves primarily with pedestrian movements across a scene but the method is general and can be extended to any scenario.

The motivation for such a system is manifold. Primary application of this method is video surveillance. At many public places, like an airport, we need to make sure that people are kept away from a certain area or on a street where one might want to make sure no one is walking drunk. Moreover, as common pathways are detected by clustering the trajectories of object, we can efficiently assign detected trajectory to its associated path model. Thus, the vision system needs only to store the path and the object labels instead of the whole trajectory set, resulting in a significant compression for storing surveillance data. The system may also be used to detect the need for paving new walkways, for example, if we detect a large number of pedestrians moving over an unpaved region, the learned path models would indicate an appropriate location for walkway construction. Figure 1 demonstrates a small number of trajectories extracted from our training sequence.



Figure 1: Sample trajectories plotted on an image plane.

The remaining paper is organized as follows. Section 2 gives a brief description of related work on path surveillance. The training phase is described in Section 3. Section 4 defines the procedure for distinguishing a non-conforming trajectory from the established paths. Finally, results and experiments are presented in Section 5, followed by the conclusion in Section 6.

2. Related Work

The first stage of path detection is tracking an object successfully across frames until it exits the scene. Tracking is essentially a correspondence problem; correspondence needs to be established between an object seen in the current frame and those seen in previous frames. Tracking is a widely researched problem and many suitable trackers exist for our purpose. We choose the tracker presented in [2] for testing our method.

Although path detection is relatively a new problem, we briefly survey this topic. Grimson W.E.L *et al.* [1] use a distributed system of cameras to cover a scene, and employ an adaptive tracker to detect moving objects. Tracks are clustered using spatial features based on the vector quantization approach. Once these clusters are obtained the unusual activities are detected by matching incoming trajectories to these clusters.

Dimitrios Makris and Tim Ellis [3] develop a spatial model to represent the routes in an image. A trajectory is matched with routes already existing in a database using a simple distance measure. If a match is found, the existing route is updated by a weight update function; otherwise a new route is created for the new trajectory. One limitation of this approach is that only spatial information is used for trajectory clustering and behavior recognition. The system cannot distinguish between a person walking and a person lingering around, or between a running and a walking person.

Boyd *et al.* [7] present a method that models the scene as a network of connected regions. They estimate the number of trips made from one region to another based on the inter-region boundary traffic counts accumulated over time. However, their method only determines the mean traffic intensities based on the calculated statistics and no information is given about trajectories. Johnson *et al.* [5] uses a neural network to model the trajectory distribution.

In this paper, we propose a simple and an innovative multi feature path detection and surveillance method that allows us to discriminate between trajectories with great confidence. We test our system on real-world sequences with pedestrians passing through a scene. The system is currently implemented on a single camera, but it can be extended to any multi-camera viewing system.

3. Training for Path Detection

The tracker gives the trajectories for objects moving across the camera. We train our system with sequences captured using a single stationary camera. Generally the trackers are able to uniquely label objects appearing in the sequence. Therefore, it is possible to maintain a history of the route taken by the object. For any object i tracked through n frames, the 2D image coordinates for the trajectory obtained can be given as:

$$T_i = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}.$$

Note that the trajectories will be of varying lengths, depending on the location and velocity of the person. The trajectories obtained through the tracker are noisy; therefore, we smooth the trajectories using a moving average filter.

To train our system, we track people walking around the scene and obtain commonly used paths. We cluster the similar trajectories using the min-cut graph algorithm recursively. A node of this graph represents a trajectory. Each node of the graph is connected with every other node, thus making a complete graph. The weight of an edge is the distance that we obtain by comparing two trajectories using the Hausdorff distance measure. For two trajectories A and B, the Hausdorff distance, $D(A, B)$, is defined as:

$$D(A, B) = \max \{d(A, B), d(B, A)\},$$

where,

$$d(A, B) = \max_{a \in A} \min_{b \in B} \|a - b\|$$

One advantage of using Hausdorff distance is that it can compare two sets of different cardinality. Thus this measure allows us to compare two trajectories of different lengths. Figure 2 shows a sample graph. Similar trajectories will have small weights because of lesser Hausdorff distance, and vice versa. Using the graph-cuts we are able to recursively partition the graph into two pieces, each consisting of a group of similar trajectories. Refer to Kolmogorov *et al.* [8] for theory and implementation of min-cut/max-flow algorithms.

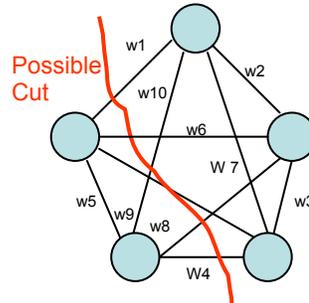


Figure 2: A sample complete graph of 5 nodes. The weight of an edge is the Hausdorff distance between two trajectories. The red line indicates a possible min-cut of the graph.

We define an *envelope* as the spatial extent of a path. The envelope is found by determining pair wise correspondence between trajectories in a cluster and selecting the boundary points. The correspondence is obtained using the Dynamic time Warping (DTW) algorithm. An average trajectory is also learned from the correspondences (see Figure 3).

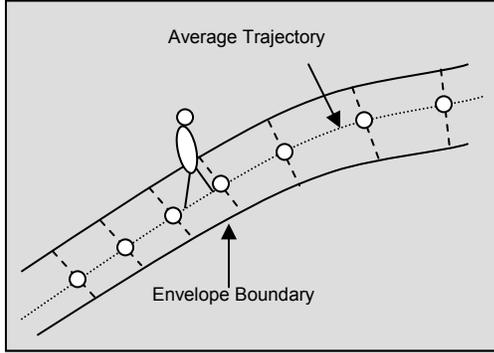


Figure 3: A typical scene with a trajectory of an object. An average trajectory and an envelope boundary are calculated for each set of clustered trajectories.

4. Scene Model

We would like to use a path model that distinguishes between trajectories that are:

- Spatially dissimilar
- Spatially similar but of different speeds
- crooked vs. straight

To achieve these goals, we first learn the usual paths by applying the graph cuts to group trajectories. Once these paths have been learned, we perform three hierarchical steps based on:

- Spatial Features
- Velocity Features
- Curvature Features

We start from the top most step and move on to a next step only if the current step is conforming to the learned path features. In the first step, we look at the spatial location of the trajectory. An object's trajectory is compared to the paths already present in the database.

Two conditions are used to determine spatial similarity. First 90% of points in the test trajectory should lie within the envelope of the path. Second the Hausdorff distance between the average path trajectory and the test trajectory should be less than the Hausdorff distance between path envelope boundaries. If these conditions are not satisfied, the trajectory is marked as anomalous, otherwise we proceed to motion based verification.

In the second step, we want to discriminate between trajectories of varying motion characteristics. Trajectory whose velocity is similar to the velocity characteristics of an existing route is considered similar. Velocity for a trajectory $P_i(x_i, y_i, t_i)$, $i = 0, 1, \dots, N$, is calculated as:

$$\mathbf{v}_i = \left(\frac{x_{i+1} - x_i}{t_{i+1} - t_i}, \frac{y_{i+1} - y_i}{t_{i+1} - t_i} \right), i = 0, 1, \dots, N-1$$

We extract the average velocity of the test trajectory. We use the Gaussian distribution to model the velocities of the trajectories in the path model. The Mahalanobis distance measure is used to decide if the test velocity is anomalous.

$$\tau = \sqrt{(\mathbf{v}_i' - \mathbf{m}_p)^T (\Sigma)^{-1} (\mathbf{v}_i' - \mathbf{m}_p)}$$

Where \mathbf{v}_i' is velocity from the test trajectory, \mathbf{m}_p is the mean and Σ is the covariance matrix of our path velocity distribution.

The third step allows us to capture the discontinuity in the velocity, acceleration and position of our trajectory. Thus we are able to discriminate between a person walking in a straight line and a person walking in an errant path. The velocity \mathbf{v}_i' and acceleration \mathbf{v}_i'' , first derivative of the velocity, is used to calculate the curvature of the trajectory. Curvature is defined as:

$$\kappa = \frac{\sqrt{y''(t)^2 + x''(t)^2 + (x'(t)y''(t) - x''(t)y'(t))^2}}{(\sqrt{x'(t)^2 + y'(t)^2 + 1})^3}$$

Where x' and y' are the x and y components of the velocity \mathbf{v}_i' . We measure mean and variance of κ 's for trajectories in our path model and fit a Gaussian distribution. We compare the curvature of the test trajectory with our distribution using the Mahalanobis distance. By using this measure we are able to detect irregular motion. For example, a drunkard walks in a zigzag path, or a person slowing down and making a u-turn.

Thus, initially we detect non-conforming trajectories on the basis of spatial dissimilarity. In case the given trajectory is spatially similar to one of the path models, the similarity in the velocity features of the trajectories in that path and the given trajectory is computed. If the motion features are also similar then a final check of curvature is made. The trajectory is deemed to be anomalous if it fails to satisfy any one of the spatial, velocity or curvature constraints.

5. Results

The proposed system has been tested on multiple sequences with a variety of motion trajectories. The sequences have a resolution of 320 x 240 pixels and captured at multiple locations and each location contained multiple paths of travel. Our tracker is able to accurately establish correspondences over a variety of environmental conditions. The training sequences are:

- **LongWalks.** This is a real sequence of 9284 frames with 27 different trajectories forming different paths after clustering. The length of the trajectories varies from 250 points to almost 800 points. The clustered trajectories are shown in figure 4.
- **ShortWalks.** This is a shorter sequence of 3730 frames with 15 different trajectories forming two unique paths. The clustered trajectories are shown in figure 4.

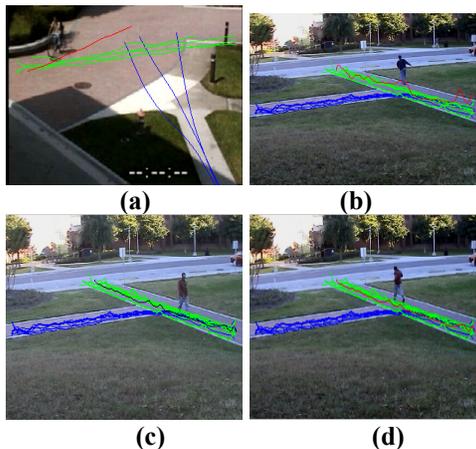


Figure 4: Results obtained from the two sequences. Image (a) detects a bicyclist in our ShortWalks sequence. Image (b), (c), and (d) show instances of a drunkard walking, a person running, and a person walking, respectively, in our LongWalks sequence. Red trajectories denote unusual behavior.

Some results are shown in Figure 4. Clustered trajectories obtained through graph cuts are shown with similar colors indicating an identified path. Figure 4(a) shows a bicyclist traversing the scene with a trajectory not matching our database; hence it is rejected during the first step. As shown in the figure, all the trajectories are clustered correctly. Figure 4(b) shows a person walking in drunken manner. The curvature of this trajectory does not match the curvature of the trained model and is detected. Figure 4(c) shows a person walking at a normal pace and following an established trajectory. Therefore, this trajectory is accepted by our system. Finally, a

person is running in Figure 4(d) having different velocity characteristics than our trained model. Even though the trajectory conforms to our spatial features, it fails to outdo our motion features; hence it is rejected. The system gives satisfactory results for all our sequences and is fairly efficient.

6. Conclusion

This paper proposes a hierarchical method for path detection and surveillance. We have shown that graph-cuts can be used for clustering trajectories. The model uses spatial, velocity and curvature features for unusual behavior detection. An unusual behavior might be a person traversing an area not traversed before, a person moving at different speed than the usual, or a person moving in varying directions. We have tested the system on multiple sequences and have obtained accurate results. We plan to use multi-camera views to further extend our system. The next step of our research is recognition of more complex events by attaching meanings to the trajectories.

References

- [1] W.E.L. Grimson, C. Stauffer, R. Romano and L. Lee, "Using Adaptive Tracking to Classify and Monitor Activities in a Site", CVPR-98
- [2] Omar Javed and Mubarak Shah, "Tracking and Object Classification for Automated Surveillance", the seventh European Conference on Computer Vision, Denmark, 2002.
- [3] Dimitrios Makris and Tim Ellis, "Path Detection in Video Surveillance", Image and Vision Computing Journal, vol.20/12, pp 895-903, October 2002.
- [4] Cen Rao, Alper Yilmaz and Mubarak Shah, "View-Invariant Representation and Recognition of Actions", IJCV Vol. 50, Issue 2, 2002.
- [5] Neil Johnson and David Hogg, "Learning the Distribution of Object Trajectories for Event Recognition", Proc. BMVC95, Birmingham, England, 1995.
- [6] Vlachos, M., Gunopulos, D., and Kollios, G, "Robust Similarity Measure for Mobile Objects Trajectories". Proc. of DEXA Workshops. (2002) 721-728
- [7] Jeffrey E. Boyd, Jean Meloche, and Y. Vardi, "Statistical Tracking in Video Traffic Surveillance", Proc. ICCV99, Corfu Greece, Sept. 1999.
- [8] Yuri Boykov and Vladimir Kolmogorov, "An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Computer Vision", EMMCVPR, September 2001.