UNSUPERVISED MODELING OF OBJECT TRACKS FOR FAST ANOMALY DETECTION

Tomas Izo and W. Eric L. Grimson

Computer Science and Artificial Intelligence Lab
Massachusetts Institute of Technology, Cambridge, MA, USA

ABSTRACT

A key goal of far-field activity analysis is to learn the usual pattern of activity in a scene and to detect statistically anomalous behavior. We propose a method for unsupervised, multi-attribute learning of a model of moving object tracks that enables fast reasoning about new tracks, both partial and complete. We group object tracks using spectral clustering and estimate the spectral embedding efficiently from a sample of tracks using the Nyström approximation. Clusters are modeled as Gaussians in the embedding space and new tracks are projected into the embedding space and matched with the cluster models to detect anomalies. We show results on a week of data from a busy urban scene.

Index Terms— image analysis, clustering, unsupervised learning

1. INTRODUCTION

A key goal of any visual surveillance system is to automatically determine when an observed scene contains unusual or unexpected activity. In the past this task was performed by a human expert: someone familiar with the scene who is able to recognize when something out of the ordinary occured. A typical surveillance site may have so many sensors in different locations that it is no longer feasible for a person to monitor all of them. Machine vision systems are needed to mine the collected data for potentially interesting activity. This has fostered a new area of machine vision research, aimed at building statistical models of the usual pattern of activity in scenes.

In this paper we focus on the behavior of individual moving objects in a complex far-field scene, such as surroundings of buildings (parking lots, streets, sidewalks), airport or train station halls or any large spaces observed by a far-field, wide field-of-view camera. A central issue with this problem is dealing with the huge volume of data. A day of observation of a typical scene may contain tens of thousands of moving objects of various sizes (people, groups, bicycles, motorbikes, cars, trucks, etc.) taking a wide variety of different paths through the scene. We want to distill, from this huge volume of data, compact models of the common (and uncommon) activities in the scene, without prior assumptions about those activities. Such models would enable us to detect unusual activities even as they are occuring, by flagging trajectories when they begin to deviate from normal modes. This could be used to alert a human observer or to cue a pan-tilt-zoom system to focus in on a particular event. With rich enough models, we could also detect unusual events by factors other than trajectory, such as time of day or type of object in a particular location.

We assume that our system tracks moving objects in the scene, providing positions as well as other attributes (e.g. size or silhouette) in every frame. We propose an algorithm that, given this data, learns a model of object motion in the scene in an unsupervised fashion by clustering object tracks, i.e., trajectories with additional attributes such as size, speed and direction.

Several recent papers address the problem of modeling object paths in a far-field moving scene. The main differences are in the choice of clustering algorithm—fuzzy k-means [4], dominant set [6], graph cuts [9, 5, 3, 11], self-organizing maps [7], agglomerative [1]—and the choice of distance measure between object trajectories (see [12] for a comparison of distance measures). Results are shown on relatively simple scenes and small data sets ranging from 10s of tracks to about 1200 [4, 6]. Most of the mentioned approaches only cluster trajectories based on position, ignoring attributes of the objects such as shape. Those that use additional attributes generally group clusters based on each attribute separately (e.g. subclustering [4]) rather than in a global fashion.

The key contributions of the approach presented here are combining the following:

(1) The algorithm is capable of using a very large dataset for learning a model of object motion through the scene. This is necessary in order to effectively model active, complex scenes where the number of observed objects per hour reaches several hundred or more. We demonstrate results on a collection of *ca.* 40,000 tracks representing a week of data from an active outdoor scene.

(2) The similarity measure between tracks allows for combining multiple object attributes in a principled manner.

(3) The model representation is efficient enough for us to reason about incoming objects online. This includes the ability to reason about partial object tracks.

Our method is particularly useful in surveillance systems that must make decisions about scene activity while it is in progress. For instance, attentive surveillance systems, in which a movable camera captures high resolution images of scene activity, require a decision process to task the camera to particular locations. For simple scenes, this can be done with a clever scheduling algorithm; complex scenes in which tasking a camera to every moving object is not feasible may require an attention model based on knowledge of the typical scene activity.

2. ALGORITHM

(1) Given a set of tracks $\{T_i\}$, cluster them into k groups based on similarity along multiple attributes. We use an approximate spectral method that clusters a large amount of data using a much smaller sample.
(2) Model object behavior in the scene as a mixture of Gaussians in the spectral embedding space.
(3) Given a (partial or complete) new track, project it into the spectral embedding space by comparison to the Nyström sample used in step 1.
(4) Find anomalies by thresholding on the likelihood of the track under the mixture model.

3. COMPARING TRACKS

An object track $T = \{\boldsymbol{o}_i^T\}$ is an ordered set of observations describing the estimated state of the object as it moves through the scene. The elements of the observation vectors $\boldsymbol{o}_i^T$ describe the object's attributes, such as size, image coordinates of the centroid, velocity, etc. A good similarity measure should evaluate two tracks as very similar when the spatial trajectories are collocated in the scene and object attributes such as velocity and size at spatially corresponding points in the tracks are similar. If two objects take the same path but have dramatically different velocities or travel in opposite directions, their tracks should not be similar.

To achieve this, we define the distance between tracks A and B as a vector of attribute-specific distances—one for each object attribute along which we wish to compare tracks:

$$\boldsymbol{D}(A,B) = [d_j(A,B)]. \tag{1}$$

To calculate the attribute-specific distance d_j between tracks A and B, we find the average difference in that attribute between observations in track A and their corresponding closest observations (in image coordinates) in track B. Let $\mathrm{pos}(\boldsymbol{o})$ be the image position of the observation $\boldsymbol{o}$, and $\boldsymbol{o}^B_{\nu(A,i)}$ be the observation in track B whose position is closest to $\boldsymbol{o}_i^A$ in track A:

$$\boldsymbol{o}^B_{\nu(A,i)} = \underset{\boldsymbol{o}_j^B}{\mathrm{argmin}}\, \|\mathrm{pos}(\boldsymbol{o}_i^A) - \mathrm{pos}(\boldsymbol{o}_j^B)\|. \tag{2}$$

The directed attribute-specific distance between tracks is:

$$d_j(A \to B) = \frac{1}{|A|}\sum_i d(\boldsymbol{o}_i^A(j), \boldsymbol{o}^B_{\nu(A,i)}(j)), \tag{3}$$

and the undirected (symmetric) distance is:

$$d_j(A,B) = \min(d_j(A \to B), d_j(B \to A)). \tag{4}$$

We use the minimum, as in [11], to ensure that partial tracks get clustered together with other tracks along the same path.

The function $d(\boldsymbol{o}_1(j), \boldsymbol{o}_2(j)$ in equation (3) is simply the scalar difference in the j-th attribute between the two observations. For instance, if the attribute is size, it is the difference between the sizes of the objects.

In our experiments, we use the area of the object's bounding box, the speed, direction of motion and position in the image as the object attributes. Other features such as object shape or appearance can also be incorporated.

We convert the distances $\boldsymbol{D}$ from (1) to similarities with a multivariate Gaussian kernel:

$$S(A,B) = \exp(-\boldsymbol{D}^T(A,B)\Sigma_{A,B}^{-1}\boldsymbol{D}(A,B)) \tag{5}$$

Because we are combining distances along several different attributes, the parameters in Σ should be set such that each attribute is weighted similarly in the cumulative distance. To do this, we set $\Sigma_{A,B}$ for each pair of tracks separately by setting the diagonal elements to be the average square distance from A or B to the rest of the tracks in the dataset, whichever is larger:

$$\Sigma_{A,B} = \mathrm{diag}(\sigma_j^{A,B}), \tag{6}$$

$$\sigma_j^{A,B} = \max\left(\frac{1}{N}\sum_k d_j^2(A,T_k), \frac{1}{N}\sum_k d_j^2(B,T_k)\right). \tag{7}$$

4. CLUSTERING TRACKS

We use the Normalized Cuts spectral clustering algorithm from [2] to group tracks into clusters:

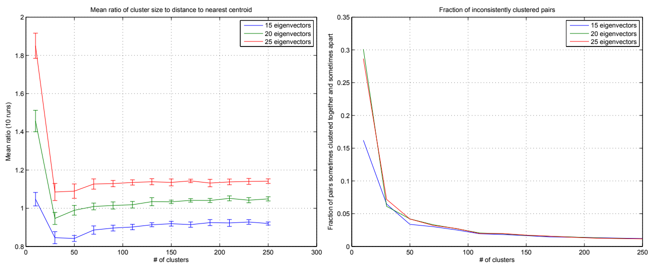
(1) Compute the NxN affinity matrix A of pairwise similarities between all N data points (each of which is a track).

(2) Scale affinity matrix A by the degree matrix D: $\mathrm{A}' = \mathrm{D}^{-1/2}\mathrm{A}\mathrm{D}^{-1/2}$. The degree matrix is a diagonal matrix of the row sums of A.

(3) Calculate the $N_E \ll N$ largest eigenvectors of the scaled affinity matrix A$'$ and assemble them as columns of the eigenvector matrix U in descending order by eigenvalue.

(4) Scale each row of the eigenvector matrix $\mathrm{U} = (u_{ij})$ by the square root of the corresponding row sum of A: $u'_{ij} = u_{ij}/\sqrt{d_i}$.

(5) The spectral embedding of each data point is the corresponding row of the scaled eigenvector matrix U$'$.



(a) (b)

Fig. 1. (a) Mean ratio of cluster size to distance to nearest cluster. (b) Fraction of inconsistently clustered pairs. 10 runs were performed for each number of clusters. The curves correspond to different dimensionalities of the embedding.

(6) Use k-means to cluster data points in the spectral embedding space.

Calculating the full similarity between each pair of data points in step 1 of the algorithm would be prohibitively expensive. Instead, we can estimate the eigenvectors of the entire affinity matrix using only comparisons between each data point and a sample of the entire dataset using the Nyström approximation, as in [2]:

(1) Calculate the n x n matrix A_n of pairwise similarities between a subset of size $n \ll N$ of the entire dataset and the n x $(N-n)$ matrix B of similarities between the n sample points and the rest of the data. The full N x N similarity matrix A can be written as

$$\mathrm{A} = \begin{bmatrix} \mathrm{A}_n & \mathrm{B} \\ \mathrm{B}^T & \mathrm{C} \end{bmatrix}. \tag{8}$$

Note that C, the bulk of the matrix, will never be computed.

(2) Let $\boldsymbol{a}_r$ be the row sums of A_n, and $\boldsymbol{b}_r$ and $\boldsymbol{b}_c$ the row and column sums of B. Estimate the row sums $\hat{\boldsymbol{d}}$ of the full matrix A (needed in steps 2 and 4 above) as

$$\hat{\boldsymbol{d}} = \begin{bmatrix} \boldsymbol{a}_r + \boldsymbol{b}_r \\ \boldsymbol{b}_c + \mathrm{B}^T \mathrm{A}_n^{-1} \boldsymbol{b}_r \end{bmatrix}. \tag{9}$$

(3) Divide each element in A_n and B by the square root of the product of the estimated row and column sum of the full matrix A, thus obtaining the scaled versions A'_n and B'. The relationship of the scaled full affinity matrix A' to A'_n and B' is the same as the relationship in equation (8).

(4) Let U be the eigenvectors and Λ the eigenvalues of A'_n such that $\mathrm{A}'_n = \mathrm{U}\Lambda\mathrm{U}^T$. Estimate the eigenvectors of A' as

$$\hat{\mathrm{U}} = \begin{bmatrix} \mathrm{U} \\ \mathrm{B}'^T \mathrm{U} \Lambda^{-1} \end{bmatrix}. \tag{10}$$

5. CLASSIFYING NEW TRACKS

To classify new tracks, we project them into the spectral embedding space obtained from the clustering using the method described in [8], where spectral clustering with the Nyström extension was used to create an neuroanatomical atlas from fiber tractography paths. The process is similar to the computation done for each row of the matrix $\hat{\mathrm{U}}$ in section 4. Given a set of m new tracks, we calculate the n x m matrix S of similarities between the new tracks and each of the tracks from the sample used to obtain matrix A in section 4. Let $\boldsymbol{s}_c$ be the column sums of S. We scale S^T to obtain S'^T using the row sums estimated as $\boldsymbol{s}_r = \boldsymbol{s}_c + \mathrm{S}^T \mathrm{A}_n^{-1} \boldsymbol{b}_r$ and the column sums $\boldsymbol{a}_r + \boldsymbol{b}_r$. The embedding of the new tracks then are the rows of $\mathrm{S}'^T \mathrm{U} \Lambda^{-1}$ (see [8]), scaled by the corresponding row sums $\boldsymbol{s}_r$. Note that this is the same computation as was done in (9) and (10).

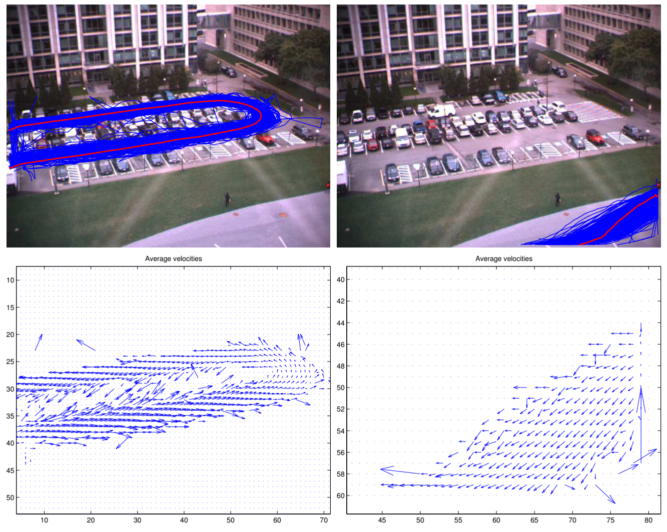


Fig. 2. Top: two of the 10 most salient clusters. Most typical track (closest to the mean) is highlighted. Bottom: each arrow shows average velocity of tracks in this cluster that pass through the arrow's origin.

6. RESULTS

For our experiments we used a dataset of *ca.* 40,000 tracks from a very busy outdoor scene, corresponding to one full week of activity. We obtained the tracks using the background subtraction and tracking algorithm from [10]. Tracking was performed at 30fps but the tracks were pruned to retain only observations at least 500ms apart. The attributes along which we compared tracks were position of the object centroid in the image, area of the bounding box, direction of motion and speed.

To determine a suitable number of clusters, we ran our clustering algorithm for various numbers of clusters and various values for N_E (the dimensionality of the spectral embedding) using a sample of *ca.* 3,000 tracks to estimate the embedding vectors. For each clustering, we calculate the av-

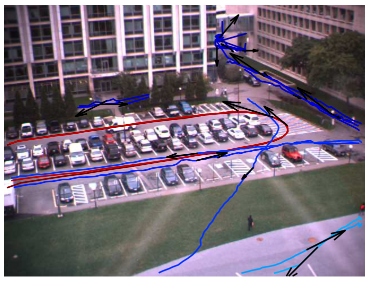


Fig. 3. Typical tracks from the top 20 most significant clusters. Arrows indicate velocity; color shows average size along track (warmer color indicates larger size).

erage ratio of cluster size (in terms of mean distance to centroid over all cluster members) to the distance from the mean to the nearest cluster centroid (shown in Figure 1a). As clusters in the data begin to be subdivided, this ratio eventually stabilizes. We also calculate the fraction of pairs of tracks that sometimes get clustered together in the same cluster and sometimes apart (Figure 1b). Based on these criteria, we select 150 as the number of clusters for our data set.

Figure 2 shows two of the most significant clusters (in terms of number of members) along with average velocities in those clusters. Both clusters consist of tracks with similar paths through the scene and similar speeds and direction of motion. The clustering is robust to erroneous and broken tracks.

In Figure 3, we show the most typical track (the one closest to the mean) from each of the top 20 most significant clusters, indicating also the velocity and the average size of the object. We can see that paths of different direction, velocity, size and location are clustered separately.

In Figure 4, we show examples of tracks with low likelihood under the scene model. In (a) the vehicle enters the pedestrian zone, backs out and continues along the road. In (b) the trajectory of the vehicle is not unusual but the velocities are: the vehicle stops for several seconds just before the turn. In (c) the person takes an unusual path through the grassy area. In (d) the person drags an object to the end of the lot, leaves it there and walks back. In this case, the changes in size and path both cause the event to appear unusual. The types of anomalies shown in (b) and (d) would be impossible to detect without being able to compare tracks in terms of multiple object attributes.

The experiments were run on a desktop workstation with dual 3GHz processors. The one-time clustering step takes several days of computation; the classification of a single new track takes a few seconds in our MATLAB implementation.

7. CONCLUSTIONS AND FUTURE WORK

We demonstrated a system capable of learning a multi-attribute scene model from a very large amount of data. Though we

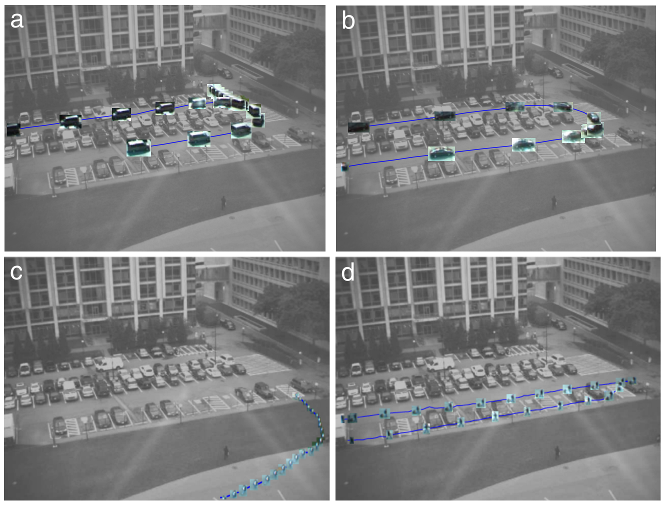


Fig. 4. Examples of unusual tracks (anomalies).

show promising results, much further analysis remains to be done. For instance, we intend to examine how the approach scales with more object attributes (appearance, time of day, etc) added to the distance measure. Because classifying new tracks is faster with a smaller sample used for the Nyström approximation, we are also interested in determining how the sample size affects clustering performance.

8. REFERENCES

[1] D. Buzan, S. Sclaroff, and G. Kollios. Extraction and clustering of motion trajectories in video. In *ICPR*, 2004.

[2] C. Fowlkes, S. Belongie, F. Chung, and J. Malik. Spectral grouping using the Nystrom method. *PAMI*, 26(2), 2004.

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

[4] W. Hu, X. Xiao, Z. Fu, D. Xie, T. Tan, and S. Maybank. A system for learning statistical motion patterns. *PAMI*, 28(9), 2006.

[5] I. N. Junejo, O. Javed, and M. Shah. Multi feature path modeling for video surveillance. In *ICPR*, 2004.

[6] X. Li, W. Hu, and W. Hu. A coarse-to-fine strategy for vehicle motion trajectory clustering. In *ICPR*, 2006.

[7] A. Naftel and S. Khalid. Motion trajectory learning in the dft-coefficient feature space. In *Proc. IEEE International Conference on Computer Vision Systems*, 2006.

[8] L. O'Donnell and C.-F. Westin. High-dimensional white matter atlas generation and group analysis. In *MICCAI*, 2006.

[9] F. Porikli. Learning object trajectory patterns by spectral clustering. In *Proc. IEEE Int. Conf. Multimedia and Expo*, 2004.

[10] C. Stauffer and W. Grimson. Adaptive background mixture models for real-time tracking. In *CVPR*, 1999.

[11] X. Wang, K. Tieu, and W. E. L. Grimson. Learning semantic scene models by trajectory analysis. In *ECCV*, 2006.

[12] Z. Zhang, K. Huang, and T. Tan. Comparison of similarity measures for trajectory clustering in outdoor surveillance scenes. In *ICPR*, 2006.