Learning Trajectory Patterns by Clustering: Experimental Studies and Comparative Evaluation

Brendan Morris and Mohan Trivedi
Computer Vision and Robotics Research Laboratory
University of California, San Diego
La Jolla, California 92093-0434
{b1morris, mtrivedi}@ucsd.edu

Abstract

Recently a large amount of research has been devoted to automatic activity analysis. Typically, activities have been defined by their motion characteristics and represented by trajectories. These trajectories are collected and clustered to determine typical behaviors. This paper evaluates different similarity measures and clustering methodologies to catalog their strengths and weaknesses when utilized for the trajectory learning problem. The clustering performance is measured by evaluating the correct clustering rate on different datasets with varying characteristics.

1. Introduction

A major research area in computer vision is the study of activities and behavior. Recently there has been high interest in automatic activity and behavior understanding. Using unsupervised methods, researchers try to observe a scene, learn prototypical activities, and use the prototypes for analysis. This paradigm has been of particular interest for surveillance [1, 2] and traffic monitoring [3–5] where methods to categorize observed behavior, detect abnormal actions for quick response, and even predict future occurrences is desired. Because of the large number of cameras in use for these applications there is a constant stream of large amounts of data making it difficult to manually analyze each individually which necessitates the use of unsupervised methods. In these cases, activity is characterized by motion and can be succinctly represented with a trajectory. It is possible to collect trajectories over sufficient time and learn typical behaviors through clustering. Unfortunately, even with much work in the area it is unclear what are the best methods for clustering. There are a wide number of similarity functions for trajectories and researchers continue their design as well as little agreement of how clustering should be performed. A recent survey by Morris and Trivedi [6] presented the wide variety of procedures for trajectory learning and modeling. This paper examines a number of popular trajectory clustering procedures to find their strengths and weakness with the intention of determining which might be the best for trajectory learning. The evaluation has three separate components which include comparison of trajectory distance measures (Table 1), comparison of different clustering methods (Table 2), and analysis on a variety of dataset with varying characteristics (Table 3).

Table 1. Trajectory Distance Measures

Technique	Publication
HU	Hu 2007 [7]
PCA	Bashir 2007 [8]
DTW	Keogh 2000 [9]
LCSS	Buzan 2004 [10]
PF	Piciarelli 2006 [3]
MODH	Atev 2006 [4]

Table 2. Clustering Techniques

Technique	Publication
Direct	Morris 2008 [11]
Divisive (rb,rbr)	Billotti 2005 [12]
Agglomerative	Buzan 2004 [10]
Hybrid (cham)	Karypris 1999 [13]
Graph	Li 2006 [14]
Spectral	Hu 2007 [7]

2. Distance Measures

Previous work by Zhang *et al*. [15] compared the use of a few popular distance measures at the time, the Hausdorff distance, a HMM-based distance, Euclidean distance, Euclidean distance in a PCA subspace, dynamic time warping (DTW), and longest common subsequence (LCSS). We expand the comparison by including new similarity measures

978-1-4244-3991-1/09/$25.00 ©2009 IEEE

$$LCSS(F_i, F_j) = \begin{cases} 0 & T_i = 0 \mid T_j = 0 \\ 1 + LCSS(F_i^{T_i-1}, F_j^{T_j-1}) & d_E(f_{i,T_i}, f_{j,T_j}) < \epsilon \;\&\; |T_i - T_j| < \delta \\ \max\left(LCSS(F_i^{T_i-1}, F_j^{T_j}), LCSS(F_i^{T}, F_j^{T_j-1})\right) & \text{otherwise} \end{cases} \quad (1)$$

that have been designed specifically for trajectories while ignoring both Hausdorff and HMM which were shown to have poor performance. Table 1 lists the distance measures adopted in recent literature which are assessed in this work. The examination includes fixed length measures, Hu Euclidean and PCA, as well as time-normalized distances, DTW, LCSS, Piciarelli and Foresti (PF), and modified Hausdorff (MODH).

2.1. Notation

A trajectory

$$F = \{f_1, \ldots, f_t, \ldots, f_T\} \quad (2)$$

is a collection of flow vectors f_t representing the spatio-temporal characteristics of moving objects at each time t of the total track life T. A flow vector generally indicates location and dynamics, $f_t = [x, y, \dot{x}, \dot{y}, \ddot{x}, \ddot{y}]$, but in this work we restrict ourselves to just spatial location, $f = [x, y]$. This is a common practice as it results in a natural interpretation of spatial proximity given the Euclidean the distance between flow points

$$d_E(f_t, f_\tau) = \sqrt{(x_t - x_\tau)^2 + (y_t - y_\tau)^2}. \quad (3)$$

Some of the following distance measures must use fixed length data and can not be used on raw trajectory data because they typically have varying length. Instead, a resampled version of a track is used and the trajectory notation is overloaded

$$F = \{f_1, \ldots, f_k, \ldots, f_L\}. \quad (4)$$

A resampled trajectory is referenced by an index k rather than time t and has a fixed length L. The resampling implementation used in this paper interpolates points to have equal distance between flow vectors [16].

2.2. HU

The HU distance is computed as the average Euclidean distance between points on two trajectories [7].

$$D_{HU}(F_i, F_j) = \frac{1}{L} \sum_{k=1}^{L} d_E(f_{i,k}, f_{j,k}), \quad (5)$$

This distance function relies on similar trajectories having the same point distribution with consecutive points in corresponding tracks in spatial proximity.

2.3. PCA

Instead of working in the trajectory coordinate space, PCA is used to transform the trajectories into a lower dimensionality subspace. The x and y coordinates of a trajectory are concatenated into a one dimensional vector and projected onto the subspace by PCA decomposition The PCA distance is computed as the Euclidean distance between PCA coefficients, a_l,

$$D_{PCA}(F_i, F_j) = \frac{1}{N_\lambda} \sum_{l=1}^{N_\lambda} d_E(a_{i,l}, a_{j,l}). \quad (6)$$

Only $N_\lambda << 2L$ coefficients are retained to limit the size of the space. N_λ is chosen by examining eigenvalues λ_k to retain 95% of the dataset variation [8]. The PCA distance is similar to Hu but works in a lower dimensional space for reduced computation and robustness through the PCA shape decomposition. Note trajectories must be of equal length for PCA decomposition.

2.4. DTW

The above distance measures require fixed length trajectories which do not normally occur because observation duration is variable. DTW is used to compare unequal length signals by finding a time warping that minimizes the total distance between matching points [17].

$$\begin{aligned} D_{DTW}(F_i, F_j) &= \frac{(d_{DTW}(F_i, F_j) + d_{DTW}(F_i, F_j))}{2} \\ d_{DTW}(F_i, F_j) &= \frac{1}{T_i} \sum_{t=1}^{T_i} d_E(\phi_{i,t}, \phi_{j,t})\; m_t / M_\phi \end{aligned} \quad (7)$$

where ϕ_i and ϕ_j are the time warping functions that minimize the distance between aligned points, m_t is a path weighting coefficient, and M_ϕ is a path normalization factor. The warping path ϕ is efficiently found using dynamic programming.

2.5. LCSS

LCSS is another alignment tool for unequal length data but is more robust to noise and outliers than DTW because not all points need to be matched. Instead of a one-to-one mapping between points, a point with no good match can be ignored to prevent unfair biasing. The LCSS distance suggested by [18] is defined as

$$D_{LCSS}(F_i, F_j) = 1 - \frac{LCSS(F_i, F_j)}{\min(T_i, T_j)}, \quad (8)$$

where the $LCSS(F_i, F_j)$ value (1) specifies the number of matching points between two trajectories. $F^t = \{f_1, \ldots, f_t\}$ denotes all the flow vectors in trajectory F up to time t. The LCSS, like DTW, can also be efficiently computed using dynamic programming.

2.6. PF

In a similar spirit to DTW and LCSS, Piciarelli and Foresti [3] defined another distance measure to deal with time drift. They observed that matching tracks would generally agree early (consistent starting points) but over time matched points had a tendency to drift further away because of speed differences. Accordingly, their trajectory distance measure finds matching points within a time window that grows larger at each time

$$D_{PF}(F_i, F_j) = \frac{1}{T_i} \sum_{t=1}^{T_i} d_{PF}(f_{i,t}, F_j) \tag{9}$$

where

$$\begin{aligned} d_{PF}(f_{i,t}, F_j) &= \min_{\tau} \left(\frac{d_E(f_{i,t}, f_{j,\tau})}{Z_\tau} \right), \\ \tau &\in \{\lfloor (1-\delta)t \rfloor \ldots \lceil (1+\delta)t \rceil\}. \end{aligned} \tag{10}$$

Z_τ is a normalization constant that measures the variance of point τ. The definition is noteworthy because it allows comparison with incomplete trajectories (developing tracks) making it well suited for online clustering unlike DTW or LCSS.

In this work, $Z_\tau = 1$ in order to compare two trajectories rather than a trajectory and a cluster as originally designed. The temporal window is also slightly modified to grow logarithmically with trajectory length $\tau \in \{\lfloor t - \delta \log t \rfloor \ldots \lceil t + \delta \log t \rceil\}$ to prevent very large windows for long trajectories.

2.7. Modified Hausdorff

The Hausdorff distance has been commonly used to compare two unequal size sets but is not well suited for trajectories because it does not account for ordering [15]. The modified Hausdorff distance $D_{MODH}(F_i, F_j)$ [4] was designed to respect the time-ordering of points and reduce sensitivity to outliers by allowing slack when matching.

$$\begin{aligned} D_{MODH}(F_i, F_j) &= \operatorname*{ord}_{f_{i,t} \in F_i}^{\alpha} h(f_{i,t}, F_j) \\ h(f_{i,k}, F_j) &= \min_{f_{j,\tau} \in \mathcal{N}(C(f_{i,t}))} d_E(f_{i,k}, f_{j,\tau}) \end{aligned} \tag{11}$$

where $\mathcal{N}()$ is a neighborhood window, $C(f_{i,t})$ is the point in F_j that correspond to point $f_{i,t}$ in F_i, and $\operatorname{ord}_{f_{i,t} \in F_i}^{\alpha} h(f_{i,t}, F_j)$ denotes the value of $h(f_{i,t}, F_j)$ that is larger than α percent of all other h values over F_i. The distance between trajectories is thus a prototypical distance chosen from among the best matching points. The trajectory alignment is controlled by the correspondence function $C(.)$ which assumes that corresponding points occur at the same fraction of total track length. This convention accounts for speed variation within similar spatial patterns.

3. Clustering Algorithms

Besides examining the effects of different trajectory distance measure, the quality of clusters returned by different types of clustering methods is explored to determine if certain techniques are better suited for trajectories. The classes of clustering algorithms we consider are direct methods, hierarchical agglomerative and divisive procedures, hybrid divisive-agglomerative techniques, graph cuts, as well as spectral methods. A summary of recent research utilizing these different clustering techniques is shown in Table 2.

For ease of clustering, a similarity matrix $S = \{s_{ij}\}$, which represents a fully connected graph, is constructed from the trajectory distances using a Gaussian kernel function

$$s_{ij} = e^{-D^2(F_i, F_j)/2\sigma^2} \in [0, 1]. \tag{12}$$

where D represents one of the distance measure defined previously and the parameter σ describes the trajectory neighborhood. Large values of σ cause further apart trajectories to have a higher similarity score while small values lead to a more sparse similarity matrix (more entries will be very small). The S matrix along with the desired number of clusters are used as input into the differing clustering algorithms which are discussed below.

3.1. Direct

The direct clustering methods find the K clusters simultaneously. A initial guess of clusters is iteratively optimized by adjusting each cluster component in unison to find a globally satisfying solution

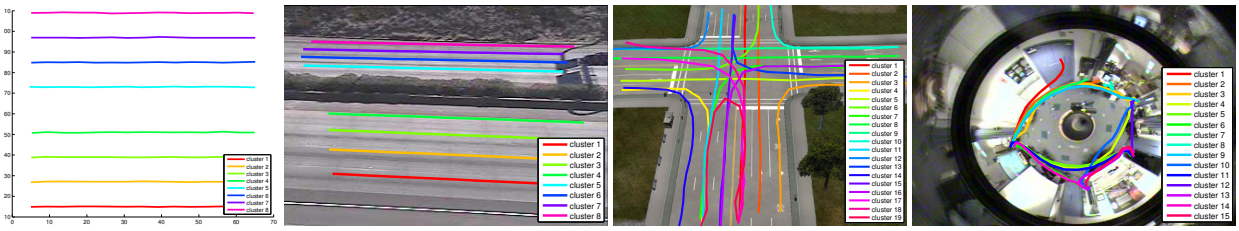
Popular direct optimization solvers in the Euclidean space are k-means and the soft assignment version fuzzy c means (FCM). These are used as the baseline clustering techniques for comparison.

3.2. Agglomerative

Agglomerative clustering is a bottom-up strategy that initially treats each trajectory as an individual cluster and merges similar clusters hierarchically in a tree-like structure, stopping when only K clusters remain. At each merge step, a hard decision on cluster membership is made limiting the algorithms ability to adjust at a higher tree level.

3.3. Divisive

Divisive clustering is the top-down dual to agglomerative clustering where the entire trajectory training set is consid-



(a) (b) (c) (d)

Figure 1. Different Datasets (a) I5SIM (b) I5 (c) CROSS (d) LABOMNI

Table 3. Experimental Dataset Characterization

	N	K	σ_V	σ_T	ξ	Δ_c/Δ	Δ_c/Δ_m
i5sim	800	8	0.19	1.29	0.99	0.07	0.27
i5sim2	1600	8	0.81	18.78	0.95	0.08	0.31
i5sim3	1600	16	0.81	18.78	0.95	0.08	1.39
i5	806	8	2.38	4.10	1.00	0.16	1.30
cross	1900	19	5.07	4.27	0.85	0.07	0.97
labomni	209	15	0.31	142.26	0.71	0.22	1.44

ered a single cluster. The K clusters are obtained by performing $K-1$ repeated bisections where each bisecting cluster split results an optimal 2-way division of the similarity matrix. In addition to ensuring local optimality at the bisections, a global optimization step can be used to optimize the solution across all bisections.

3.4. Hybrid

Hybrid clustering solutions combine both divisive and agglomerative techniques. By using different criterion functions during the partitioning and agglomeration phases, more complex (non-globular) clusters can be discovered. The dataset is first clustered into $M > K$ clusters using one of the partition methods and the final K clusters are obtained by merging some of the M clusters.

3.5. Graph

Similar to the divisive clustering method, graph methods seek to divide the full dataset into individual clusters [19]. Instead of operating directly on the similarity matrix, a nearest neighbor graph is constructed where a trajectory is a vertex. Each vertex is connected by a weighted edge to its most similar trajectories. The K clusters are found using a min-cut partitioning algorithm which finds a division of the graph with minimal loss of edge weights.

3.6. Spectral

Spectral clustering has become popular recently because it can be efficiently computed and improved performance over more traditional clustering algorithms such as k-means. Spectral methods do not make any assumptions on the distribution of data points and instead relies on eigen decomposition of the similarity matrix which approximates an optimal graph partition [20]. We compare compare 4 flavors of spectral algorithms by selecting to decompose either the Laplacian of Shi and Malik [19] or Ng *et al.* [20] followed by a final clustering of eigenvectors with either k-means or FCM.

4. Datasets

Experiments are conducted using six datasets with varying properties. They include several simulated scenes as well as surveillance scenes where trajectories are extracted using a background based object tracker. Figure 1 illustrates each scene with true clusters and Table 3 summarizes the datasets. Each set is characterized by the number of trajectories N, the number of cluster labels K, speed deviation σ_V, length deviation σ_T, shape complexity ξ [15],

$$\xi = \frac{d_E(f_T, f_1)}{\sum_i d_E(f_{i+1}, f_i)} \tag{13}$$

and separability Δ_c/Δ and Δ_c/Δ_m. The average separability is

$$\Delta = \frac{1}{NK} \sum_{n=1}^{N} \sum_{c=1}^{K} d_{nc} \tag{14}$$

and the cluster tightness Δ_c and minimum cluster separability Δ_m are defined as

$$\Delta_c = \frac{1}{N} \sum_{\substack{n=1 \\ g_n=c}}^{N} d_{nc} \tag{15}$$

$$\Delta_m = \min_n \left[\min_c \; d_{nc} \right], \quad c \neq g_n \tag{16}$$

Table 4. Best CCR Performance (Average over 5 runs)

	kmeans	fcm	hu	pca	dtw	lcss	pf	modh
i5sim	0.8162	0.8900	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
i5sim2	0.7250	0.8154	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
i5sim3	0.4901	0.4984	0.5735	0.5574	0.9994	1.0000	0.9944	0.7725
i5	0.6397	0.8000	0.9107	0.7655	0.9722	0.9975	0.9613	0.9975
cross	0.6937	0.7163	0.9963	0.9947	0.9958	0.9869	0.9947	0.9916
labomni	0.7952	0.7923	0.8900	0.9091	0.8383	0.9091	0.8325	0.8325

where g_n is the ground truth label for track n and

$$d_{nc} = \frac{1}{N_c} \sum_{g_j = g_c} d_E(F_n, F_j) \tag{17}$$

with N_c the number of trajectories with label g_c.

4.1. I5SIM

The I5SIM datasets simulate trajectories obtained on a 4 lane highway with traffic in both directions (8 total lanes). The trajectory points are in real world coordinates. The first set contains only free flow traffic with a Gaussian speed distribution of 70 mph with a 5 mph standard deviation. The second and third sets contain the free flow traffic as well as trajectories during congestion (25 mph). These were designed to compare performance of different clustering goals. The trajectories in the I5SIM2 dataset are labeled by lane number (just spatial coordinates). Those in I5SIM3 were labeled by lane and flow type, differentiating trajectories in the same lane but traveling at different speeds, resulting in 16 labels (8 lanes at free flow and 8 during congestion).

4.2. I5

The I5 dataset contains trajectories obtained by visual tracking of vehicles from a highway mounted camera overlooking a busy interstate. The track labels correspond to one of 8 lane numbers as in the I5SIM sets. The raw tracker output was automatically filtered to remove clearly erroneous trajectories which occur because of occlusions.

4.3. CROSS

The CROSS dataset depicts a four way traffic intersection. These provide more complex trajectory shapes than in the highway datasets. The 19 acceptable intersection maneuvers include turns and even a u-turn.

4.4. LABOMNI

The final dataset examines humans rather than vehicles. An omni-directional camera was placed in the middle of a lab to observe trajectories from a less constrained environment than encountered by vehicle traffic. The participants were not aware of the data collection to ensure naturally occurring motion patterns. The trajectories have a long time duration and tend to have a large degree of overlap in the image plane.

5. Experimental Evaluation

The experimental results are presented in the following section. The best classification results are displayed in Table 4. These results are the average performance over 5 runs for each dataset and similarity type using 48 different clustering method variations. The average results versus cluster method, distance measure, and dataset are presented in Fig. 3.

5.1. Evaluation Criteria

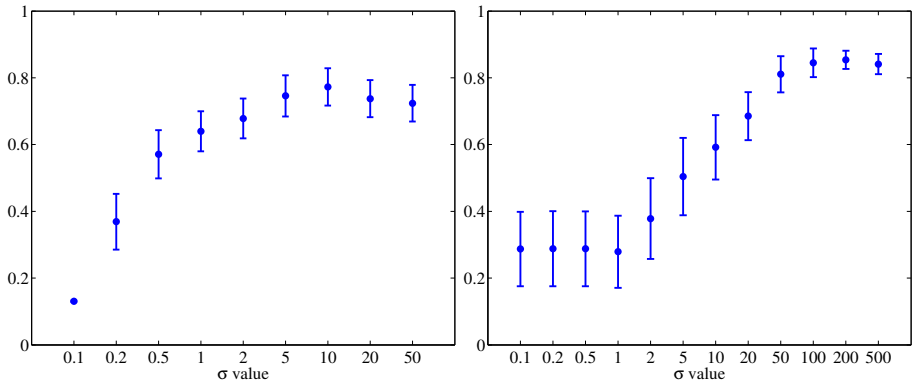
Since the labels returned by a clustering run was arbitrary, the accuracy of a clustering was evaluated by finding the one-to-one mapping between the ground truth and clustering labels which maximized the number of matches. The assignment problem can be solved using the Hungarian algorithm [21] when recast to minimize the number of mismatched labels. Given the label mapping, the cluster quality is measured by the correct clustering rate (CCR) [15]

$$CCR = \frac{1}{N} \sum_{c=1}^{K} p_c \tag{18}$$

where N is the total number of trajectories and p_c denotes the total number of trajectories matched to the c-th cluster.

5.2. Procedure

The CLUTO [22] software package is used for agglomerative, divisive, hybrid (CHAMELEON [13]), and graph based clustering. The software provides a number of options and optimization criteria for each cluster method which results in a total of 44 different cluster variants. An additional 4 spectral cluster variants were implemented in Matlab for a total of 48 cluster variants applied for each similarity measure to each dataset. Every clustering combination was run 5 times with random initialization to better represent expected performance.



(a) (b)

Figure 2. Clustering quality for different σ values for (a) HU and (b) DTW averaged across all datasets. As σ increases the performance improves resulting in higher CCR and lower variance.

The experimental evaluation consisted of three main parts. The effect of the neighborhood parameter σ was investigated, a sweep was done through all distance measure parameters to ensure near optimal values, and finally the clustering evaluation was performed by varying the distance measure, cluster method, and dataset.

5.3. Gaussian Kernel Evaluation

The effect of trajectory neighborhood on clustering was examined by varying the parameter σ in (12). Fig. 2 shows improved performance as σ increases. The average CCR not only increases but the variance decreases. Although the quality appears to saturate at a particular σ choice, larger values than this cause little performance degradation. An average similarity $\frac{1}{N^2}\sum_i\sum_j s_{ij} = 0.1$ was used to produce good results.

5.4. Clustering Method Evaluation

The plot in Fig. 3(a) shows that on average the choice of clustering method has little effect on the quality of the results. It is noteworthy to mention that the soft membership of FCM improves performance 6% over k-means making it a clear winner between the baseline approaches. All cluster methods perform significantly better than the baseline except for the direct method which is the same category both k-mean and FCM fall into. Although the graph results were only 5% lower, graph based clustering was significantly more difficult as results were very sensitive to the graph neighborhood definition.

5.5. Distance Measure Evaluation

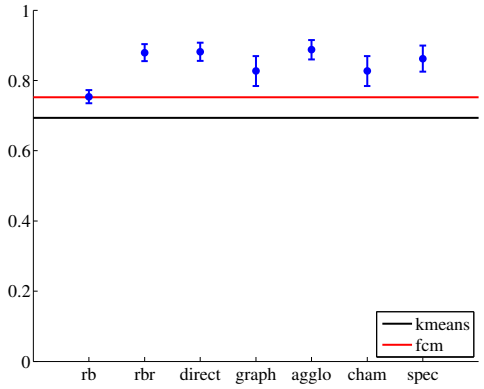
The average CCR results as a function of distance measure is shown in Fig. 3(b). This shows the effort in designing measures that allow the use of raw variable length trajectories is not wasted since the sampled measures, HU and PCA, perform almost 10% worse. Unfortunately, the newer trajectory specific distances, PF and MODH, have comparable performance with DTW and LCSS.

Further insight can be found by examining the columns of Table 4. The performance of HU, PCA, and MODH all degrade for the I5SIM3 dataset where there are different speeds in the same lane. The speed information is thrown out during resampling when using HU and PCA and it is also ignored by the modified Hausdorff distance because the corresponding points are mapped based on total trajectory length. While LCSS performs uniformly well, it is surprising that both DTW and PF which do not have outlier robustness exhibit corresponding performance. The need for outlier suppression is lessened for trajectory data because of the smoothing inherent in the tracker (*e.g.* Kalman or particle filter). Unless the tracker makes a gross mistake it is unlikely for a trajectory to contain outlier points that would greatly influence warping match distance.

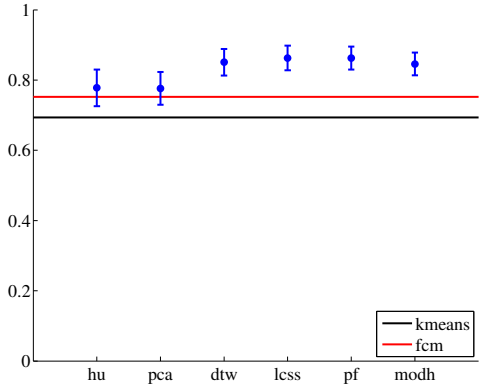
5.6. Dataset Evaluation

The preceding sections implied little difference in effectiveness when using different clustering methods or different time aligned distances but inspection of Fig. 3(c) clearly differentiates performance between datasets. The CCR results are quite high for the more simple I5SIM, I5SIM2, and CROSS datasets. With $\Delta_c/\Delta_m < 1$, these sets have tight clusters well separated from one another.

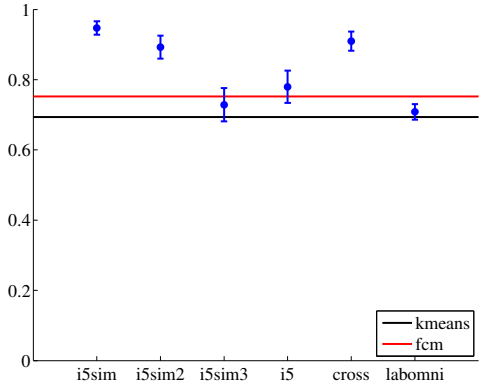
The bar graph in Fig 4 indicates a performance distribution across distance functions for each dataset which is lost in the averaged plot. Fig. 5 shows a detailed view of I5SIM2 and I5SIM3. These sets had 8 highway lanes with trajectories collected from 2 different speed profiles, free flow and congestion, but only I5SIM3 required differentiation based on speed as well as lane number. All distance methods performed very well in the I5SIM2 set, Fig. 5(a), except LCSS which actually did worse than FCM. There



(a)



(b)



(c)

Figure 3. Average CCR performance plotted against experimental variables. (a) Clustering algorithm (b) Distance measure (c) Dataset

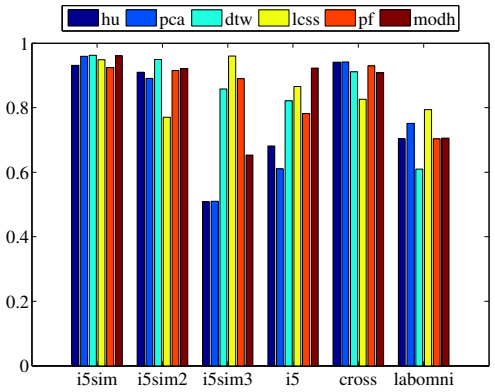
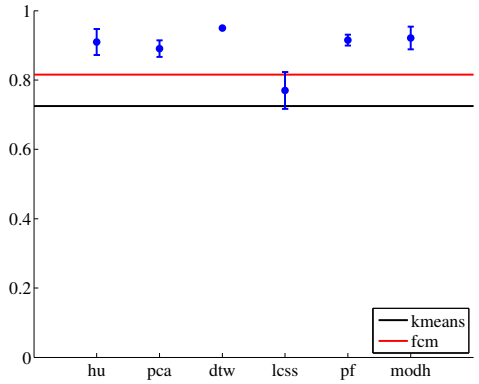
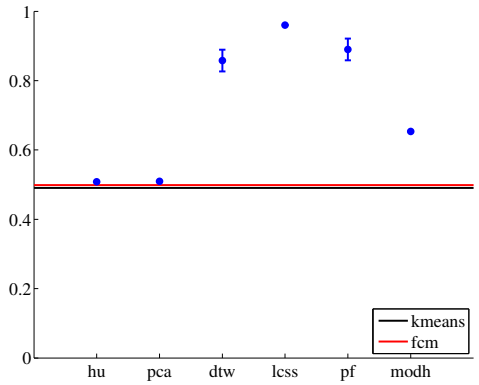


Figure 4. Average performance for the different similarity measures for each dataset.



(a)



(b)

Figure 5. (a) Average CCR for I5SIM2. The direct and divisive methods perform poorly for LCSS. (b) Average CCR for I5SIM3. The time alignment measures {DTW, LCSS, PF} perform significantly better.

was a large variation in performance given the clustering method and though perfect clustering was possible (see Table 4) the direct and divisive solutions lowered the average performance. Fig. 5(b) shows the dramatic improvements possible with the right distance choice. The DTW, LCSS, and PF distances were able to resolve both position and speed differences with a high degree of accuracy.

Viewing Table 4 we see the LABOMNI dataset performance was similar across all the distance types, even the baseline k-means and FCM. HU and PCA which reduce dimensionality and focus on shape performed better than all the time alignment techniques, except LCSS, because the long length of trajectories which allow ample opportunity for misalignment..

Another interesting result from Table 4 is the significant CCR loss for PCA in the I5 dataset. $\xi = 1$ and high Δ_c/Δ_m score indicates straight overlapping lanes due to camera perspective which causes the northbound lanes furthest from the camera to appear very close in the image plane. Unlike the LABOMNI set which has a lower ξ value, trajectories cannot be distinguished well by intermediate points and the PCA decomposition filters out the fine differences.

6. Concluding Remarks

This work evaluated the performance of a number of clustering procedures for the trajectory clustering task. The evaluation consisted of a comparison of 6 trajectory distance measures, 7 clustering methods, and 6 varied datasets. Without prior knowledge, the choice of clustering method and distance measure was not important as long as it operated on full unsampled tracks, though LCSS was consistently a top performer. Performance was actually dictated by the trajectory properties encountered in a dataset. When trajectories were very long the data reduction techniques worked well by focusing on coarse shape and position and when dynamics were considered an important separating factor the time-normalized distances dominated.

Acknowledgments

We acknowledge the sponsorship of UC Digital Media Program for the experimental infrastructure and thank Mr. Minh Van Pham Ly of UC Berkeley who assisted as a UCLEADS program intern.

References

[1] C. Stauffer and W. E. L. Grimson, "Learning patterns of activity using real-time tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 747–757, Aug. 2000.

[2] D. Makris and T. Ellis, "Learning semantic scene models from observing activity in visual surveillance," *IEEE Trans. Syst., Man, Cybern. B*, vol. 35, no. 3, pp. 397–408, Jun. 2005.

[3] C. Piciarelli and G. L. Foresti, "On-line trajectory clustering for anomalous events detection," *Pattern Recognition Letters*, vol. 27, no. 15, pp. 1835–1842, Nov. 2006.

[4] S. Atev, O. Masoud, and N. Papanikolopoulos, "Learning traffic patterns at intersections by spectral clustering of motion trajectories," in *IEEE Conf. Intell. Robots and Systems*, Bejing, China, Oct. 2006, pp. 4851–4856.

[5] B. T. Morris and M. M. Trivedi, "Learning, modeling, and classification of vehicle track patterns from live video," *IEEE Trans. Intell. Transp. Syst.*, vol. 9, no. 3, pp. 425–437, Sep. 2008.

[6] ——, "A survey of vision-based trajectory learning and analysis for surveillance," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 8, pp. 1114–1127, Aug. 2008.

[7] W. Hu, D. Xie, Z. Fu, W. Zeng, and S. Maybank, "Semantic-based surveillance video retrieval," *IEEE Trans. Image Process.*, vol. 16, no. 4, pp. 1168–1181, Apr. 2007.

[8] F. I. Bashir, A. A. Khokhar, and D. Schonfeld, "Object trajectory-based activity classification and recognition using hidden markov models," *IEEE Trans. Image Process.*, vol. 16, no. 7, pp. 1912–1919, Jul. 2007.

[9] E. Keogh and M. Pazzani, "Scaling up dynamic time warping for datamining applications," in *Intl. Conf. on Knowledge Discovery and Data Mining*, sep 2000, pp. 285–289.

[10] D. Buzan, S. Sclaroff, and G. Kollios, "Extraction and clustering of motion trajectories in video," in *Proc. IEEE Inter. Conf. on Pattern Recog.*, Aug. 2004, pp. 521–524.

[11] B. Morris and M. Trivedi, "An adaptive scene description for activity analysis in surveillance video," in *Proc. IEEE Inter. Conf. on Pattern Recog.*, Tampa, Florida, Dec. 2008.

[12] D. Biliotti, G. Anotonini, and J. P. Thiran, "Multi-layer hierarchical clustering of pedestrian trajectories for automatic counting of people in video sequences," in *IEEE Workshop on Application of Computer Vision.*, Brekenridge, CO,, Jan. 2005, pp. 50–57.

[13] G. Karypis, E.-H. Han, and V. Kumar, "Chameleon: A hierarchical clustering algorithm using dynamic modeling," *IEEE Computer*, vol. 22, no. 8, pp. 68–75, Aug. 1999.

[14] X. Li, W. Hu, and W. Hu, "A coarse-to-fine strategy for vehicle motion trajectory clustering," in *Proc. IEEE Inter. Conf. on Pattern Recog.*, 2006, pp. 591–594.

[15] Z. Zhang, K. Huang, and T. Tan, "Comparison of similarity measures for trajectory clustering in outdoor surveillance scenes," in *Proc. IEEE Inter. Conf. on Pattern Recog.*, 2006, pp. 1135–1138.

[16] B. Morris and M. Trivedi, "Learning and classification of trajectories in dynamic scenes: A general framework for live video analysis," in *Proc. IEEE International Conference on Advanced Video and Signal based Surveillance*, Santa Fe, New Mexico, Sep. 2008.

[17] L. Rabiner and B. Juang, *Fundamentals of Speech Recognition*. Englewood Cliffs, New Jersey: Prentice-Hall, 1993.

[18] M. Vlachos, G. Kollios, and D. Gunopulos, "Discovering similar multidimensional trajectories," in *Proc. IEEE Conf. on Data Engineering*, Feb. 2002, pp. 673–684.

[19] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, Aug. 2000.

[20] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Advances in Neural Information Processing Systems*, vol. 14, sep 2002, pp. 849–856.

[21] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Research Logistic Quarterly*, vol. 2, pp. 83–97, 1955.

[22] (2008) CLUTO 2.1.1 - software for clustering high-dimensional datasets. [Online]. Available: http://glaros.dtc.umn.edu/gkhome/cluto/cluto/overview