Clustering of Vehicle Trajectories

Stefan Atev, Grant Miller, and Nikolaos P. Papanikolopoulos, *Fellow, IEEE*

***Abstract*—We present a method that is suitable for clustering of vehicle trajectories obtained by an automated vision system. We combine ideas from two spectral clustering methods and propose a trajectory-similarity measure based on the Hausdorff distance, with modifications to improve its robustness and account for the fact that trajectories are ordered collections of points. We compare the proposed method with two well-known trajectory-clustering methods on a few real-world data sets.**

***Index Terms*—Clustering of trajectories, time-series similarity measures, unsupervised learning.**

I. Introduction

THE trajectories of vehicles through a traffic scene are an information-rich feature that reveals the structure of the scene, provides clues to the events taking place, and allows inference about the interactions between traffic objects. Several vision algorithms are designed to collect trajectory data, either as their primary output or as a by-product of their operation.

The focus of this paper is to present a method for unsupervised clustering of vehicle trajectories with widely varying lengths. We augment the preliminary results presented in [1] with a comparison of the proposed method with two well-established approaches for trajectory clustering on several sets of trajectories for which manual ground truth was obtained. The intended application for the method is as an extension to the traffic data-collection system in [2], but automatic clustering has applications in collision prediction, as in [3].

Researchers in data mining and management are chiefly concerned with deriving similarity measures on trajectories. The sophistication of these measures is necessitated by the use of relatively simple clustering and indexing schemes, e.g., agglomerative clustering or k-means. One example of this approach is discussed in [4], which uses the longest common subsequence (LCSS) measure to cluster vehicle trajectories. The edit distance on real sequences in [5] and [6] is closely related to LCSS. The need for compact representation and fast retrieval of trajectory data has resulted in dimensionality reduction approaches, e.g., the segmented principal component analysis (PCA) in [7]. Another popular distance measure for trajectories and general time series is dynamic time warping (DTW), which is exemplified in [8], where DTW is applied to a rotation-invariant representation of trajectories. In an earlier work, [9] argued in favor of LCSS over DTW, which is a view shared by [10], where the same conclusion is reached. In both cases, LCSS only significantly outperforms DTW on data with large amounts of pointwise outliers that are ignored by the distance threshold of LCSS; the method proposed in this paper is similarly robust to the presence of outlying points, but that aspect turns out to be not important for our experiments, where DTW clearly outperforms LCSS.

This paper is similar to [11], which compares the unmodified Hausdorff distance, LCSS, and DTW on a task that is very similar to the task considered here. In addition, the authors test a plain Euclidean distance and a Euclidean distance after linear dimensionality reduction with PCA. The experiments with the last two distances require resampling of trajectories, which loses information (in the particular case, speed information and spatial fidelity). We particularly do not attempt to compare methods that require fixed-dimensional trajectory representation, because they require a commitment to a specific trajectory model or features. Our only assumption is that the spatial aspects of the problem are more important than the dynamic aspects, i.e., a vehicle's path through the scene is more important than the speed/acceleration. This assumption is also implicitly made in DTW and LCSS, and we will not consider the necessary modifications to remove it from the methods. Another work that compares an unmodified Hausdorff variant with LCSS is [12], which find the measures to perform similarly when outliers are accounted for.

The first objective of this paper is to compare DTW, LCSS, and a modified Hausdorff (MH) distance when applied to the task of clustering trajectories of vehicles at traffic scenes. Our focus is not on indexing and retrieval but, rather, on unsupervised learning. The second purpose of this paper is to investigate the performance of spectral clustering methods for the problem domain. As our results show, there is no undue computational burden when working with moderately sized data sets. The clustering method that we use is a modification of the method in [13], which uses a symmetrically normalized graph Laplacian as the affinity matrix. Rather than using a fixed global scale, as in [13], we have adopted the scale-selection strategy suggested in [14], where local scaling was

Manuscript received April 13, 2008; revised April 11, 2008, June 13, 2009, December 26, 2009, and March 13, 2010; accepted March 16, 2010. Date of publication May 10, 2010; date of current version September 3, 2010. This work was supported in part by the U.S. Army Research Laboratory and the U.S. Army Research Office under Contract 911NF-08-1-0463 (Proposal 55111-CI), by the National Science Foundation through under Grant IIS-0219863, Grant IIP-0443945, Grant IIP-0726109, Grant CNS-0708344, and Grant IIP-0934327, by the Minnesota Department of Transportation, and by the ITS Institute at the University of Minnesota. The Associate Editor for this paper was Q. Ji.

S. Atev is with the Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN 55455 USA, and also with Vital Images, Inc., Minnetonka, MN 55343-4411 USA (e-mail: atev@cs.umn.edu).

G. Miller was with the Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN 55455 USA. He is now with Concrete Software Inc., Eden Prairie, MN 55344-7707 USA (e-mail: gmiller@cs.umn.edu).

N. P. Papanikolopoulos is with the Department of Computer Science and Engineering and Security in Transportation Technology Research and Applications, Center for Transportation Studies, University of Minnesota, Minneapolis, MN 55455 USA (e-mail: npapas@cs.umn.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITS.2010.2048101

introduced to the same spectral clustering variant. The use of spectral clustering is motivated by the demonstrated good performance of various methods, as summarized in [15]. Instead of using a distortion metric to optimize a scale parameter, as in [13], we use it to automatically detect the number of clusters in the data. Finally, we directly incorporate the directed Hausdorff distance in the local scaling method in [14] rather than first symmetrizing the distance to fit the framework of that work.

All of the approaches compared in this paper are similarity based, use a nonparametric trajectory representation, and are invariant to the dynamic properties of the trajectories, except for direction. Model-based (parametric) approaches allow for the normalization of trajectories of various lengths and number of points but are not considered in this comparison, because we want to evaluate model-free methods. We use the output of an automated tracking system operating at frame rates; therefore, we do not consider methods for resampling the trajectories. Different representations of trajectories are investigated in [16], where the discrete wavelet transform, PCA, discrete Fourier transform, and linear predictive coding coefficients are all considered as normalization procedures for the clustering of time series. Hidden Markov models have also been used to represent trajectories and time series, as in [17] and [18]. Finally, [19] exemplifies the modeling of univariate time series by autoregressive models.

Applications that can benefit from unsupervised learning of trajectory patterns are demonstrated in [20], which were used in an accident detection and prediction system. The extraction and clustering of trajectories is used for event detection in [21]. A relevant discussion of how trajectory information can improve tracking performance is provided in [22], where previously observed trajectories provide information to disambiguate subsequent tracking. Similarly, [23] uses trajectory clustering to determine lane boundaries in a traffic video. The boundaries aid shadow detection and elimination, which also ultimately improves subsequent tracking. In [24], the trajectory information aids vehicle classification. A trajectory prior learned through clustering also enables anomaly detection, as exemplified in [25], which uses the spectral clustering method in [13] without modifications.

The rest of this paper is organized as follows. In Section II, we summarize our data collection method and include the definitions of LCSS and DTW for completeness. The proposed similarity measure for trajectories is developed in Section III, and the specifics of the clustering methods used are outlined in Section IV. The data sets and performance measure used are explained in Section V, whereas results are described in Section VI. We conclude with our remarks and proposed future work in Section VII.

II. Background

We start this section by briefly describing our data-collection method. The tracking algorithm used is similar to the tracking algorithm used in our previous works [26], [27]. The remainder of the section will be used to summarize the LCSS and DTW similarity measures as used in the comparisons. A typical output of our tracking method is shown in Fig. 1.

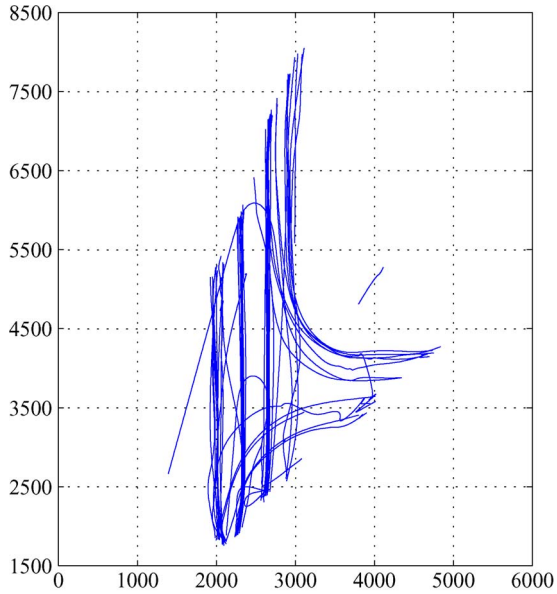


Fig. 1. Sample data set. Trajectories collected at an urban intersection.

A. Vehicle-Tracking Method

We track vehicles in color and black-and-white video from static traffic cameras. The primitive features for the tracker are connected components that are extracted from the foreground, as identified by a background subtraction method that models each pixel as a Gaussian mixture with a few components. Before segmentation, illumination-level filtering and video stabilization are applied to reduce artifacts and errors due to camera vibrations and sudden illumination changes. The image-level tracking determines foreground region associations from one time instant to another based on fractional overlap and uses some heuristics to detect region splits and merges that may indicate dynamic and static occlusions. Regions tracked without splits and merges for a few video frames are instantiated into targets for tracking, unless they already belong to a target.

The second level of tracking represents targets as sets of connected regions. Using camera calibration data, a 3-D box model is fitted to the convex hull of each target's connected regions. The center of the box model projected on the ground plane is the vehicle location estimate, and the box dimensions are the vehicle size. Although we do not use these measurements for classification purposes, we use the dimensions to filter out outliers (e.g., tracked pedestrians) and incorrectly initialized targets. The extended Kalman filter is used to accommodate the nonlinearity due to the perspective projection. In addition, once a target is determined to have left the scene, we use the intermediate output of the stochastic filter as input to the Kalman smoother, thus ensuring that the location and size estimate at each time step is conditioned on all available measurements. As shown in Fig. 1, the trajectories that are recovered are extremely smooth and well behaved. Finally, because the tracker operates at frame rates, the trajectories are sampled at a rate of 30 Hz, which eliminates the need to interpolate them.

B. LCSS and DTW Similarity Measures

The version of LCSS that we use is defined in [4]. Given two scalar sequences $\mathbf{a} = \{a_i\}_{i=1}^{n}$ and $\mathbf{b} = \{b_j\}_{j=1}^{m}$, the 1-D LCSS similarity $\ell_{\epsilon,\delta}(\mathbf{a}, \mathbf{b})$ is defined as follows:

$$\ell_{\epsilon,\delta}(\mathbf{a}, \mathbf{b}) = \begin{cases} 0, & n = 0 \vee m = 0 \\ 1 + \ell_{\epsilon,\delta}(\mathbf{a}', \mathbf{b}'), & |a_n - b_m| < \epsilon \wedge |n - m| \leq \delta \\ \max\left\{\ell_{\epsilon,\delta}(\mathbf{a}', \mathbf{b}), \ell_{\epsilon,\delta}(\mathbf{a}, \mathbf{b}')\right\}, & \text{otherwise} \end{cases} \tag{1}$$

where $\mathbf{a}' = \{a_i\}_{i=1}^{n-1}$, and similarly, $\mathbf{b}' = \{b_j\}_{j=1}^{m-1}$. The parameter ϵ controls the sensitivity of the method, and the parameter δ controls the maximum time difference between samples that are aligned.

The similarity $\ell_{\epsilon,\delta}(\mathbf{a}, \mathbf{b})$ is transformed to a distance $L_{\epsilon,\delta,\rho}(\mathbf{a}, \mathbf{b})$ by the following steps:

$$L_{\epsilon,\delta,\rho}(\mathbf{a}, \mathbf{b}) = \begin{cases} \infty, & \min\{n, m\} < \rho \max\{n, m\} \\ \min\{n, m\} / \ell_{\epsilon,\delta}(\mathbf{a}, \mathbf{b}), & \text{otherwise} \end{cases} \tag{2}$$

and the parameter ρ ensures that series of very different lengths will be considered different, regardless of their overlap. In our experiments, outlier trajectories are prefiltered; therefore, we set the value of ρ very low.

For multivariate series, the distance $L_{\epsilon,\delta,\rho}$ is separately evaluated for each component, and the root mean square of all individual scalar distances is used. Finally, note that, unlike [4], we do not want the similarity measures to be translation invariant; therefore, we do not minimize the distance over all possible translations of the inputs. The standard algorithm for LCSS computation with dynamic programming has $O(mn)$ time complexity. In specific cases, such as for cases with low likelihood of overlap, methods, e.g., in [28] and [29], that deliver performance closer to $O((m+n)\log(n+m))$ can be used.

The DTW distance between two multivariate time series $X = \{\mathbf{x}_i\}_{i=1}^{n}$ and $Y = \{\mathbf{y}_j\}_{j=1}^{m}$ is defined as

$$W(X, Y) = \min_{f} \frac{1}{n} \sum_{i=1}^{n} \left\|\mathbf{x}_i - \mathbf{y}_{f(i)}\right\|_2 \tag{3}$$

where we have assumed, without loss of generality, that X is the longer sequence, and the minimization is over all mappings $f : [1, n] \to [1, m]$ that satisfy the following requirements:

$$f(1) = 1 \tag{4}$$

$$f(n) = m \tag{5}$$

$$f(i) \leq f(j) \qquad \text{for all } 1 \leq i < j \leq n. \tag{6}$$

The distance $W(X, Y)$ can be computed in $O(nm)$ time by using dynamic programming, as described in [8], for example. In our experiments, we do not use approximations to the DTW distance. The trajectories used in [4] and [8] needed no interpolation due to the sufficiently high sampling rate; therefore, we can directly use these formulations in our experiments.

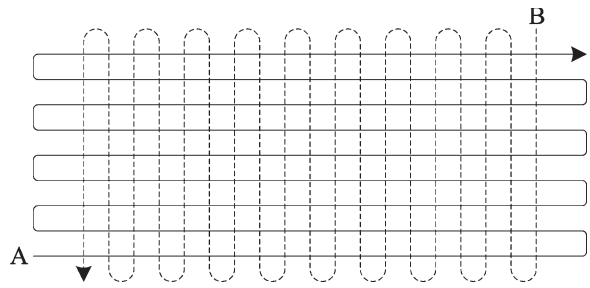


Fig. 2. Trajectories A and B are close together when treated as unordered sets as each point in A is close to one in B. The proposed modification increases the reported distance by a factor of 4.6.

III. Trajectory Similarity

In this section, we propose a trajectory similarity measure based on a modification of the Hausdorff distance. Given a universal set Ω and a metric $d : \Omega \times \Omega \to \mathbb{R}$, the (directed) Hausdorff distance between the sets $P \subseteq \Omega$ and $Q \subseteq \Omega$ is defined as

$$h(P, Q) = \max_{\mathbf{p} \in P} \left\{ \min_{\mathbf{q} \in Q} d(\mathbf{p}, \mathbf{q}) \right\} \tag{7}$$

and it should be noted that, in general, $h(P, Q) \neq h(Q, P)$. The standard extension of h to a metric is to define $H(P, Q) = \max\{h(P, Q), h(Q, P)\}$. We do not consider the case when P or Q may be uncountable sets, because we deal with finitely sampled time series. As will become apparent in our discussion of local scaling, we will use the semimetric h and not the symmetric distance H.

One deficiency of the Hausdorff distance is that even a single outlier point in either P or Q can result in an arbitrarily large distance between the sets, even if they are otherwise identical. In addition, time series are ordered collections of points, and the Hausdorff distance is defined on unordered collections. As shown in Fig. 2, two trajectories that are quite different may be very close together under the Hausdorff distance if treated as unordered point sets.

We address the task-specific deficiencies of the Hausdorff distance by modifying it in two ways: 1) We reject a certain proportion of the worst matches under the maximum in (7), and 2) we restrict the set under which the minimum is taken to a subset of the second set. We have

$$h_{\alpha,N,C}(P, Q) = \operatorname*{\overset{\alpha}{ord}}_{\mathbf{p} \in P} \left\{ \min_{\mathbf{q} \in N_Q(C_{P,Q}(\mathbf{p}))} d(\mathbf{p}, \mathbf{q}) \right\} \tag{8}$$

where $C_{P,Q} : P \to Q$ is a function that assigns to each point $\mathbf{p} \in P$ a corresponding point $\mathbf{q} \in Q$, and $N_Q : Q \to \mathcal{P}(Q)$ is a function that specifies a subset of Q as the neighborhood of the point $q \in Q$. Together, N_Q and $C_{P,Q}$ impose a structure over which the matching is performed. The operator $\operatorname{ord}_{s \in S}^{\alpha} f(s)$ denotes the value among the image $f(S)$ of the set S that is greater than $\alpha|f(S)|$ of all the values. Note that the directed

Hausdorff distance is a special case of (8), because $h(P,Q) = h_{1,N,C}(P,Q)$ for any combination of maps N_Q and $C_{P,Q}$ that satisfy $N_Q(C_{P,Q}(\mathbf{p})) = Q$ for all $\mathbf{p} \in P$ (i.e., an unrestricted correspondence structure). The meaning of α can intuitively be understood using the following facts:

$$\operatorname*{ord}_{s\in S}^{1} f(s) = \max_{s\in S} f(s) \tag{9}$$

$$\operatorname*{ord}_{s\in S}^{0} f(s) = \min_{s\in S} f(s) \tag{10}$$

$$\operatorname*{ord}_{s\in S}^{1/2} f(s) = \operatorname*{median}_{s\in S} f(s). \tag{11}$$

The rejection of a certain proportion of large distances in the Hausdorff distance has been suggested for edge template matching in [30], where it was observed that the maximum of the shortest distances is extremely sensitive to outliers, because even a single spurious point in one of the sets can affect the distance. The following properties of the distance that make it a good candidate for shape matching are identified in [31]: 1) the relative insensitivity to local shape perturbations; 2) the applicability to partial shape matching (weakened slightly by our structure imposition); and 3) due to the relatively straightforward geometric intuition behind the distance. The imposition of a correspondence structure on the two sets was not considered in these early works, because they treated objects and shapes as unordered collections of points in the plane.

A choice of $\alpha < 1$ makes the distance robust to a fraction of outliers $1-\alpha$ but also allows for partial trajectories to match and can lead to violations of the triangle inequality. As [32] points out, such violations can result in inconsistent clustering. As our early results in [1] indicated, the overall clustering results are sensitive to the choice of α—in runs of the clustering algorithm with different values of α, increasing differences in α led to increasingly different clustering results, because many more partial matches are allowed. These two considerations prompted us to fix the value of α to 0.88 in subsequent experiments. The value was experimentally determined in our earlier work.

Our choices for a correspondence function C and a neighborhood structure N both rely on a parameterization of the trajectories. Our goal is to discover spatial structures of the trajectories; therefore, we chose the arc-length parameterization. Given a trajectory $P = \{\mathbf{p}_i\}_{i=1}^n$ with $\mathbf{p}_i \in \mathbb{R}^2$, we first define the length of P as

$$|P| = \sum_{i=2}^{n} \|\mathbf{p}_i - \mathbf{p}_{i-1}\|_2 \tag{12}$$

which can be defined for any kind of series using the same norm as the one used in the Hausdorff distance for matching.

For a trajectory P, we define the subsequence formed by the first k points as $P_k = \{\mathbf{p}_i\}_{i=1}^k$ (therefore, $P = P_n$) and define the relative position of the jth point $\mathbf{p}_j$ in P as $\pi_P(\mathbf{p}_j)$, i.e.,

$$\pi_P(\mathbf{p}_j) = \frac{|P_j|}{|P|}. \tag{13}$$

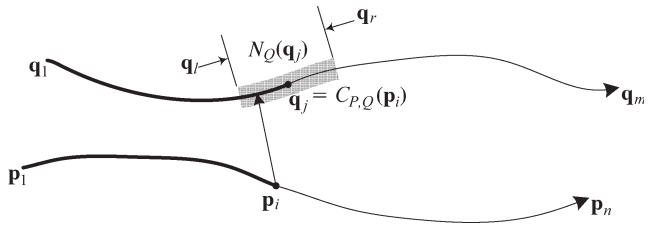


Fig. 3. Correspondence function $C_{P,Q}$ and neighborhood N_Q as used in the definition of $h_{\alpha,N,C}$. The following relationships hold for points indicated on the diagram: $\pi_P(\mathbf{p}_i) = \pi_Q(\mathbf{q}_j)$, and $\pi_Q(\mathbf{q}_j) - \pi_Q(\mathbf{q}_l) = \pi_Q(\mathbf{q}_r) - \pi_Q(\mathbf{q}_j) = w/2$.

Using the notation $P(j) = \mathbf{p}_j$, we also define the reverse transformation $r_P : [0,1] \to P$ as

$$r_P(a) = P\left(\arg\min_{1\le j\le n} \left| a - \frac{|P_j|}{|P|} \right| \right). \tag{14}$$

The correspondence structure $C_{P,Q}$ is then defined as

$$C_{P,Q}(\mathbf{p}) = Q\left(r_Q\left(\pi_P(\mathbf{p})\right)\right) \tag{15}$$

whereas the neighborhood structure N_Q is defined as

$$N_Q(\mathbf{q}_0) = \{\mathbf{q} \in Q \,|\, |\pi_Q(\mathbf{q}_0) - \pi_Q(\mathbf{q})| < w/2\} \tag{16}$$

where the parameter w controls the relative size of the neighborhood. Note that w has a similar function to the parameter δ of the LCSS-based distance $L_{\epsilon,\delta,\rho}$ but the underlying parameterization is not time based as it is in LCSS.

An illustration of $C_{P,Q}$ and N_Q is given in Fig. 3, which should make the somewhat abstract definitions more clear.

Now that the parameters α and w are specified, we will denote the Hausdorff distance as $h_{\alpha,w}$, where it is understood that N_Q and $C_{P,Q}$ use the arc-length parameterization for correspondence and w for neighborhood size. The standard spectral clustering method in [13] uses affinities (similarities) using the Gaussian kernel with a global scale parameter σ, i.e.,

$$k_{\mathrm{G}}(\mathbf{p},\mathbf{q}) = \exp\left(-\frac{d^2(\mathbf{p},\mathbf{q})}{2\sigma^2}\right) \tag{17}$$

where $d^2(\mathbf{p},\mathbf{q})$ is the squared distance between $\mathbf{p}$ and $\mathbf{q}$. One problem with using a fixed global scale parameter σ is that this approach assumes that clusters are equally dense and have similar scales. Because spectral clustering is very sensitive to the choice of scale, and the selection of a proper value for σ is nontrivial, [14] proposed a local scaling method that estimates the scale of the data in the neighborhood of each data point, i.e.,

$$k_{\mathrm{L}}(\mathbf{p},\mathbf{q}) = \exp\left(-\frac{1}{2}\frac{d(\mathbf{p},\mathbf{q})^2}{\sigma(\mathbf{p})\sigma(\mathbf{q})}\right) \tag{18}$$

where the scale σ is now dependent on the data points. The standard way of applying the locally scaled kernel calls for the use of a symmetric distance. Instead, we split the squared distance in (18) into a product of two directed distances. We have

$$k_{\mathrm{D}}(\mathbf{p},\mathbf{q}) = \exp\left(-\frac{1}{2}\frac{d(\mathbf{p},\mathbf{q})}{\sigma(\mathbf{p})}\frac{d(\mathbf{q},\mathbf{p})}{\sigma(\mathbf{q})}\right). \tag{19}$$

The splitting of the squared distance into the product of $d(\mathbf{p},\mathbf{q})$ and $d(\mathbf{q},\mathbf{p})$ suggests that there is no need for d to be symmetric (the kernel will be symmetric regardless), which means that we can directly use the directed Hausdorff distance $h_{\alpha,w}$. Intuitively, $d(\mathbf{p},\mathbf{q})/\sigma(\mathbf{p})$ is the distance from $\mathbf{p}$ to $\mathbf{q}$, as shown on a scale appropriate for the neighborhood of $\mathbf{p}$, and $d(\mathbf{q},\mathbf{p})/\sigma(\mathbf{q})$ is the distance from $\mathbf{q}$ to $\mathbf{p}$, as shown on a scale appropriate for $\mathbf{q}$'s neighborhood. If the underlying distance is symmetric, k_{D} is equivalent to k_{L}.

We choose the scale $\sigma(\mathbf{p})$ based on the distance of $\mathbf{p}$ to its ninth nearest neighbor, using similar reasoning as in [14], i.e., the distance to a close neighbor is representative of the sample density in the region. Although the neighbor index should theoretically increase with the number of points, that condition would prevent clustering in data sets with large sample imbalances; therefore, a small fixed constant is preferable. This scaling approach is insensitive to the dimensionality of the data, because the increasing distances between perturbed points in high dimensions also affect the nearest neighbors. The algorithm produces practically identical results with the fifth and seventh nearest neighbors; thus, we kept a value consistent with our earlier work. In practice, the algorithm is much less sensitive to the choice of nearest neighbor than to the choice of global scale [14]. If the σ value is outside the range 0.4 and 2 m, we clip the value to the appropriate boundary. With that definition, we can write the final similarity measure that we use as

$$k(\mathbf{p},\mathbf{q}) = \exp\left(-\frac{h_{\alpha,w}(\mathbf{p},\mathbf{q})h_{\alpha,w}(\mathbf{q},\mathbf{p})}{2\sigma(\mathbf{p})\sigma(\mathbf{q})}\right) \tag{20}$$

which is used whenever we need to use the MH distance in a spectral clustering framework.

At this point, we would like to summarize the differences and similarities between the proposed distance and the LCSS and DTW methods. The actual distances computed by both DTW and the MH distance directly depend on an underlying metric and have the same units as the point distance itself; in contrast, LCSS always returns a unitless distance that is based on a count. The 1-D LCSS distance between two series of length N can take on only $N+1$ distinct values, because it ultimately depends on an integer count that can range between 0 and N. The length normalization of the DTW distance [the factor of $1/n$ in (3)] is somewhat ambiguous when two sequences of widely differing lengths are compared, because the normalization factor depends on the number of points of only one of the series, whereas the MH distance is based on a robust norm that needs no normalization by the number of points at all. The parameterization of LCSS used to control the "slack" in the matching is time based (i.e., the parameter δ limits a correspondence neighborhood based on time), whereas the proposed Hausdorff distance controls the matching based on the arc length, which is important when the series under comparison have longer periods without change (e.g., a vehicle stopped to wait for a traffic light does not change its position). Although both DTW and the MH distance allow for a meaningful computation of relative distances, the LCSS distance is less discriminating because of the early application of the threshold ϵ. The time complexity of all distances, in general, is $O(mn)$.

IV. Trajectory Clustering

In this section, we describe the two clustering methods used and also clarify the rules that we use to decide which of the trajectories produced by the tracking method in Section II are considered outliers for our purposes. In the following exposition, d_{ij} will denote the distance between the ith and jth input trajectories (used by the agglomerative method), and k_{ij} will denote their corresponding similarity after appropriate scale selection (used in the spectral method).

A. Outlier Filtering

Visual tracking can fail under various circumstances; therefore, not all automatically collected trajectories correspond to a single vehicle, and in extreme cases, the objects being tracked are not vehicles at all. This case is particularly true for traffic intersections, where tracking is very difficult in moderately dense traffic due to occlusions. Such outlier trajectories, if included in the input to the clustering algorithms, produce many singleton clusters and impede the learning process. We remove outliers in advance, to the extent possible. Because our tracking method produces dimension estimates and positions in real-world units, we have a sensible method for prefiltering most trajectories generated by tracking failure or by merged targets.

A trajectory of a tracked object that violates any of the following rules is considered an outlier and not used in our experiments.

1) At least 90 samples (30 for highways) must be available.
2) The object length must exceed the width.
3) The object length must be between 1.5 and 6 m.
4) The width must be between 1 and 4 m.
5) The maximum instantaneous velocity over the trajectory must be between 10 and 100 km/h.
6) The objects' turning rate must be below 45° per frame.

A certain number of trajectories that conform to all aforementioned criteria were considered outliers in the ground truth. We kept these outliers in the input to the algorithms to test the methods' robustness. This approach is in contrast with [11], which manually eliminated all outliers in their experiments. None of the clustering methods explicitly detect outliers; therefore, we excluded the outlier trajectories from the clustering confusion matrix (i.e., the score that an algorithm received was not influenced by the cluster that it assigned to the outliers).

B. Agglomerative Clustering

The agglomerative clustering method that we use is identical to the one in [4], i.e., each trajectory is initially assigned a singleton cluster. The two closest clusters are iteratively merged until all clusters are separated by a distance greater than a prespecified threshold τ. If I and J denote the sets of indices of the trajectories of two disjoint clusters, the distance between the clusters is

$$d_{\mathrm{avg}}(I,J) = \frac{1}{|I||J|}\sum_{i\in I}\sum_{j\in J} d_{ij}. \tag{21}$$

In our tests, we specify the desired number of clusters rather than choosing a value of τ, because the number of clusters is known from the manual ground-truth data. The range of values for τ that produce the same number of clusters is indicative of the sensitivity of the underlying distance measure and will be part of our experiments.

C. Spectral Clustering

As already explained, we use a variant of the clustering algorithm proposed in [13], but we use the local scaling method in [14]. We include the description of the algorithm here for completeness. Note that we automatically choose the number of clusters by using the distortion metric used in [13] to select the optimal global scale.

The input to the spectral clustering method is the affinity matrix $\mathbf{K}$ with elements k_{ij} for $1 \leq i, j \leq n$. The first step of the process is to normalize the rows and columns of $\mathbf{K}$, which yields a normalized affinity $\mathbf{L}$, i.e.,

$$\mathbf{L} = \mathbf{W}^{-1/2}\mathbf{K}\mathbf{W}^{-1/2} \tag{22}$$

where the ith element of the diagonal matrix $\mathbf{W}$ is defined as

$$w_i = \sum_{1 \leq j \leq n} k_{ij}. \tag{23}$$

In an ideal noise-free case, the matrix $\mathbf{L}$ is block diagonal and has g eigenvalues with value 1 and $n - g$ zero eigenvalues, where g is the actual number of clusters in the data. Unfortunately, counting the number of eigenvalues of $\mathbf{L}$ that are equal to 1 is not a practical way of figuring out the number of clusters as pointed out in [14]. However, we can at least find a suitable range for the number of clusters using the spectrum of $\mathbf{L}$. Let $1 \geq \lambda_1 \geq \lambda_2 \geq \cdots \lambda_n \geq 0$ be the eigenvalues of $\mathbf{L}$. We estimate the minimum number of clusters in the data $g_{\min}$ by counting how many eigenvalues are greater than 0.99, and we estimate the maximum number of clusters $g_{\max}$ by counting how many eigenvalues are greater than 0.8. These thresholds were experimentally chosen. For some of our tests, the possible range was very narrow, whereas it was quite large for others.

Once we choose the search range for the number of clusters, we compute the eigenvectors of $\mathbf{L}$ that correspond to the top $g_{\max}$ eigenvalues. Let $\mathbf{v}_i$ denote these vectors for $1 \leq i \leq g_{\max}$. In the case of a repeated eigenvalue, we require that the corresponding eigenvectors are mutually orthogonal. The clustering algorithm proceeds by using these steps that are repeated for each integer value of g, inclusively ranging from $g_{\min}$ to $g_{\max}$.

1) For all integer values of g between $g_{\min}$ and $g_{\max}$, form the $n \times g$ matrix $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_g]$.
2) Normalize each row of $\mathbf{V}$ to have a unit length $\mathbf{R} = \mathbf{S}\mathbf{V}$, where $\mathbf{S}$ is diagonal with elements $s_i = (\sum_{j=1}^{c} \mathbf{V}_{ij})^{-1/2}$.
3) Treat the rows of $\mathbf{R}$ as g-dimensional data points and cluster them using k-means. Let $\mu_1, \mu_2, \ldots, \mu_g$ denote the g cluster centers (as row vectors), and let $c(i)$ denote the cluster that corresponds to the ith row $\mathbf{r}_{(i)}$.
4) The within-class scatter is $W = \sum_{i=1}^{n} \|\mathbf{r}_{(i)} - \mu_{c(i)}\|_2^2$, and the total scatter is $T = \sum_{i=1}^{n} \sum_{j=1}^{k} \|\mathbf{r}_{(i)} - \mu_j\|_2^2$. Compute the distortion score $\rho_g = W/(T - W)$ as the ratio of the within-class to between-class scatter.

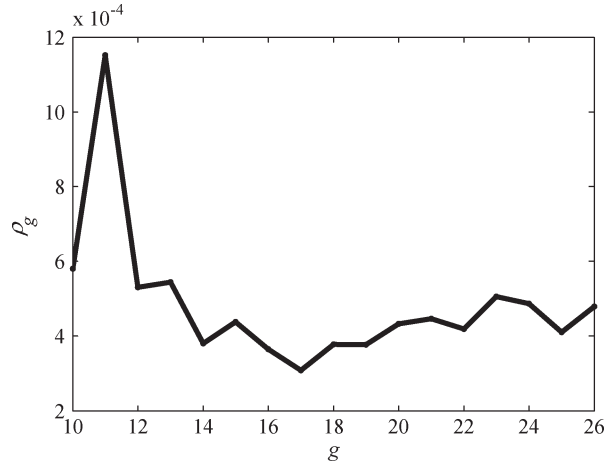


Fig. 4. Distortion ρ_g for data set 1 [see Fig. 5(a)] using the Hausdorff distance.

The value of g that produces the least distortion ρ_g is the automatically determined number of clusters, and the $c(i)$ obtained with that number of clusters is the class indicator function for the input trajectories. We show a plot of ρ_g versus g in Fig. 4. Note that the actual cost ρ_g is not monotonic or well behaved; thus, a search over all reasonable values of g is necessary.

V. Experiments

In this section we explain the experiments performed. After discussing the metrics for comparing the different clustering algorithms, we show the test data with the manual ground-truth labeling and finally discuss the choice of parameters for the various algorithms.

A. Comparing Clusterings

The clustering similarity measure used in our earlier work [1] can be shown to be an upper bound on the misclassification error probability. As such, it is well suited to comparing clusterings that are generally very similar or are produced by very similar methods. Although such a measure is good for testing the stability and sensitivity of a single method, it is not very sensible for comparing different methods. To compare the different clustering methods, we use the variation of information distance in [33], which discusses several desirable properties for clustering-comparison metrics. It is proven that only unbounded distances, e.g., the variation of information metric, satisfy all desirable properties for comparison of clustering methods. The misclassification error measure that we use is also discussed in [33], and we include results from it in all cases where the method produced the same number of clusters as the ground truth.

We compare two clusterings $C_1 : [1, \ldots, n] \rightarrow [1, \ldots, k_1]$ (mapping the n trajectories to k_1 labels) and $C_2 : [1, \ldots, n] \rightarrow [1, \ldots, k_2]$ (mapping the same n trajectories to k_2 labels) by

TABLE I
CONFUSION MATRICES WITH CORRESPONDING ERRORS

Method	$\mathbf{C}$	$d_{\mathrm{VI}}(\mathbf{C})$	$d_{\mathrm{CE}}(\mathbf{C})$	$d_{\mathrm{CE}^*}(\mathbf{C})$
1	$\frac{1}{227}\begin{pmatrix}10 & 0 & 0\\ 4 & 2 & 56\\ 0 & 145 & 10\end{pmatrix}$	0.474	0.903	0.070
2	$\frac{1}{227}\begin{pmatrix}0 & 10 & 0\\ 0 & 0 & 62\\ 155 & 0 & 0\end{pmatrix}$	0.000	1.000	0.000

first building the confusion matrix $\mathbf{C}$ with elements c_{ij} as follows:

$$c_{ij} = \frac{1}{n}\left|\{m|C_1(m) = i \wedge C_2(m) = j\}\right| \tag{24}$$

which means that c_{ij} is the number of points placed in cluster i by the mapping C_1 and in cluster j by the mapping C_2, normalized by the total number of points.

Table I shows an example of a confusion matrix for data set 4 [see Fig. 5(d)] with two of the tested methods: 1) agglomerative DTW (method 1) and 2) spectral Hausdorff (method 2). The table also lists the values of the error measures defined as follows. The data set has 227 nonoutlier trajectories, hence, the factor of 1/227.

The variation of information distance $d_{\mathrm{VI}}(\mathbf{C})$ measures the information lost and gained when switching from clustering C_1 to C_2 and is computed as

$$d_{\mathrm{VI}}(\mathbf{C}) = H(C_1) + H(C_2) - 2I(C_1, C_2) \tag{25}$$

where $I(C_1, C_2)$ is the mutual information between clusterings C_1 and C_2, and $H(C_1)$ and $H(C_2)$ are the respective entropies of the clusterings. These quantities can be computed from $\mathbf{C}$ as follows:

$$I(C_1, C_2) = \sum_{i=1}^{k_1}\sum_{j=1}^{k_2} c_{ij}\log\frac{c_{ij}}{m_{1,i}m_{2,j}} \tag{26}$$

$$H(C_1) = -\sum_{i=1}^{k_1} m_{1,i}\log m_{1,i} \tag{27}$$

$$H(C_2) = -\sum_{j=1}^{k_2} m_{2,j}\log m_{2,j} \tag{28}$$

where $m_{1,i} = \sum_{j=1}^{k_2} c_{ij}$ and $m_{2,j} = \sum_{i=1}^{k_1} c_{ij}$ are the marginals of $\mathbf{C}$. The distance d_{VI} is insensitive to permutations of the rows or columns of the confusion matrix and achieves a value of 0 only when two clusterings completely agree (up to a permutation of labels). As previously discussed, the distance is unbounded.

The classification error distance $d_{\mathrm{CE}}(\mathbf{C})$ is normally used to compare classification methods (where the class labels must agree). We use it only when $k_1 = k_2$ and use the standard definition, i.e.,

$$d_{\mathrm{CE}}(\mathbf{C}) = 1 - \sum_{i=1}^{k_1} c_{ii}. \tag{29}$$

We want an error measure that is invariant to permutations of the labels; therefore, we use the derived clustering error distance $d_{\mathrm{CE}^*}(\mathbf{C})$, which is defined as follows:

$$d_{\mathrm{CE}^*}(\mathbf{C}) = \min_{\mathbf{P}\in\Pi_k} d_{\mathrm{CE}}(\mathbf{PC}) \tag{30}$$

where Π_k is the set of all $k \times k$ permutation matrices, and $k = k_1 = k_2$. It usually is not necessary to do an exhaustive search over all $k!$ permutation matrices, because it is possible to directly compute some elements of the permutation matrix when an element c_{ij} is the largest element in both its corresponding column and row. Both d_{CE} and d_{CE^*} take on values between 0 and 1, with 0 indicating no errors and 1, indicating complete disagreement. Unlike d_{CE}, the measure d_{CE^*} cannot achieve the 1 bound, because a permutation that puts a nonzero element on the diagonal of $\mathbf{PC}$ can always be found.

B. Trajectory Data

We use data from the four traffic scenes shown in Fig. 5 to test the clustering algorithms. Two of the data sets, i.e., 1 and 2, are from traffic intersections, and the rest are from free-flowing highway traffic. The input to the algorithms is the set of trajectories after outlier prefiltering. The ground-truth partitions of trajectories for the four scenes are indicated on each panel. A histogram of trajectory lengths combined over all data sets is shown in Fig. 5(e) to illustrate the great variability in trajectory lengths (the x-axis is logarithmic).

C. Clustering Experiments

To the extent possible, we use equivalent parameters in all similarity measures. One particular problem with LCSS was that, for some choices of ϵ, the algorithm cannot produce the desired number of clusters, because many of the trajectories were at infinite distance from each other (the range for the LCSS distance in $\mathbb{R}^2$ is $[\sqrt{2}/2, \infty)$, but any two trajectories separated by a distance exceeding ϵ everywhere result in a distance of ∞). In such situations, we show the results of the algorithm using two different settings of ϵ. The first setting is 12 ft (the lane width in the scenes), and the second one is the smallest value of ϵ that was large enough so that the desired number of clusters was produced.

In cases where the automatically detected number of clusters differs from the true number of clusters, we present two scores for the proposed algorithm. To give an idea of the suitability of the proposed trajectory distance, we also used agglomerative clustering with the MH distance.

For LCSS and the MH distance, we used a slack parameter of half the trajectory lengths, i.e., we set $w = 0.5$ for the modified distance and $\delta = 0.5$ for LCSS (very similar to the value in [4]). Finally, the parameter α for the Hausdorff method was set to 0.88, which was determined to be optimal on a subset of data set 1 in [1]. Note that the optimal α in [1] was chosen without recourse to the ground-truth data—it was the value that minimized the distortion score discussed in Section IV. The same value was used on all other data sets with good results and was not optimized for each of the remaining data sets.

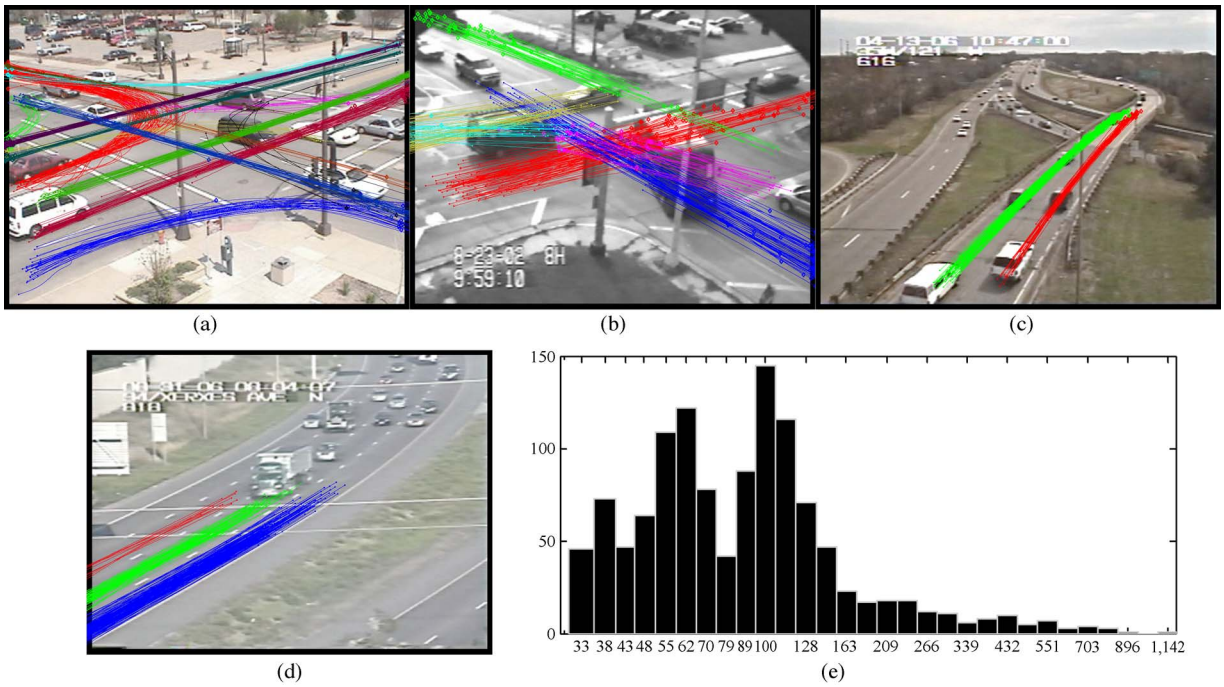


Fig. 5. Four scenes from which the trajectories were collected and the distribution of their lengths. (a) Scene 1: 403 trajectories, 38 (9.4%) outliers. (b) Scene 2: 391 trajectories, six (1.5%) outliers. (c) Scene 3: 161 trajectories, one (0.6%) outlier. (d) Scene 4: 240 trajectories, 13 (5.4%) outliers. (e) Distribution of trajectory lengths.

VI. Results

The variation of information between the various clustering methods and the ground truth is shown in Table II. The clustering error is also shown for clusterings that produce the same number of clusters as the ground truth. The actual number of clusters in the input is known to all algorithms, except for spectral Hausdorff (HS), spectral DTW (DS), and LS-ϵ, which automatically determine it. The LCSS-based methods also indicate the value of ϵ used, because it varied in one of the experiments. The best result for each distance measure (as measured by d_{VI}) is highlighted in *italics*, and the best overall result is highlighted in **boldface**. The method names in Table II are given as follows:

Name	Distance	Type	Setting of k
HA	Hausdorff	Agglomerative	Manual
HS	Hausdorff	Spectral	Automatic
HM	Hausdorff	Spectral	Manual
DA	DTW	Agglomerative	Manual
DS	DTW	Spectral	Automatic
DM	DTW	Spectral	Manual
LA-ϵ	LCSS	Agglomerative	Manual
LS-ϵ	LCSS	Spectral	Automatic
LM-ϵ	LCSS	Spectral	Manual

A top–down view of the output of the proposed HS method is shown in Fig. 6.

A. Discussion

The first very clear result in Table II is that the MH distance better reflects the notion of trajectory similarity; even when used in an agglomerative fashion, the distance outperforms both DTW and LCSS. The MH distance also outperforms the other distances when used with the spectral clustering algorithm. The second observation is that, when given the correct number of clusters, the spectral methods outperform the equivalent agglomerative method with the LCSS and Hausdorff distances. The fact that the LCSS distance used with larger thresholds does not benefit from the spectral clustering is a result of the unreasonably high thresholds used (which were necessary so that the agglomerative method produced the required number of clusters). The advantages of the spectral clustering are particularly clear in data set 3, where the clusters contain several partially overlapping trajectories due to the widely varying starting point of successful tracking. Although data set 3 appears very simple, the great variability in trajectory length and trajectory endpoints challenges all agglomerative clustering variants, because the intercluster and intracluster distances both have large variances. The freeway data sets did not challenge the spectral method at all—both the number of clusters and class memberships were perfectly recovered. Surprisingly, HS outperformed HM on the second data set, although it underestimated the true number of clusters [compare Fig. 5(b) with Fig. 6(b)]. This result can be explained by the fact that HS merged two clusters that had significant overlap but otherwise produced results identical to the ground truth for all other trajectories. Due to the overlap, HM also merged the same clusters, but it split another sparse cluster into two subclusters to obtain the specified six clusters.

The performance of LCSS for this problem, when used in the agglomerative fashion, is discouraging. When given a very reasonable ϵ threshold, which is equal to the actual interlane width, LA-12 failed to merge enough clusters to

TABLE II
COMPARISON TO GROUND-TRUTH CLUSTERING FOR THE FOUR SCENES IN FIG. 6

Method	Data set 1			Data set 2			Data set 3			Data set 4		
	k	d_{VI}	d_{CE^*}	k	d_{VI}	d_{CE^*}	k	d_{VI}	d_{CE^*}	k	d_{VI}	d_{CE^*}
Dynamic Time Warping												
DA	*14*	*0.671*	*0.255*	6	1.048	0.486	2	0.913	0.438	*3*	*0.474*	*0.070*
DS	24	1.321	–	*6*	*0.535*	*0.229*	*3*	*0.852*	–	4	0.700	–
DM	14	1.114	0.375	*6*	*0.535*	*0.229*	2	0.961	0.537	3	0.647	0.370
Modified Hausdorff Distance												
HA	14	0.124	0.044	6	0.373	0.151	2	0.895	0.419	**3**	**0.000**	**0.000**
HS	17	0.191	–	**5**	**0.152**	–	**2**	**0.000**	**0.000**	**3**	**0.000**	**0.000**
HM	**14**	**0.102**	**0.030**	6	0.213	0.101	**2**	**0.000**	**0.000**	**3**	**0.000**	**0.000**
LCSS With $\epsilon = 12$												
LA-12	21	1.477	–	18	1.509	–	3	1.107	–	14	2.229	–
LS-12	26	1.477	–	*5*	*0.558*	–	3	0.664	–	*3*	*0.743*	*0.176*
LM-12	*14*	*1.314*	*0.397*	6	0.611	0.239	*2*	*0.610*	*0.119*	*3*	*0.743*	*0.176*
LCSS With Varying ϵ												
	$\epsilon = 22$			$\epsilon = 43$			$\epsilon = 15$			$\epsilon = 30$		
LA-ϵ	*14*	*1.600*	*0.468*	6	1.943	0.561	*2*	*0.895*	*0.419*	*3*	*1.751*	*0.595*
LS-ϵ	26	1.731	–	15	2.148	–	3	1.491	–	4	1.927	–
LM-ϵ	14	1.637	0.490	*6*	*1.887*	*0.517*	2	1.185	0.331	3	1.770	0.551

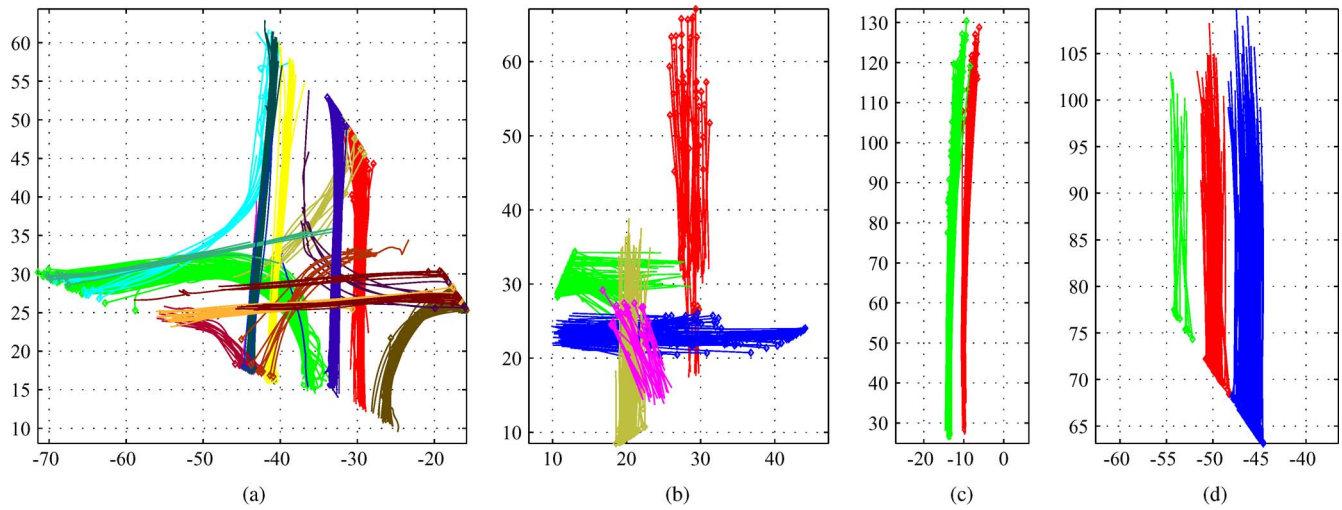


Fig. 6. Result of the proposed method on the four data sets. The grid units are in meters. (a) Data set 1. (b) Data set 2. (c) Data set 3. (d) Data set 4.

reach the specified correct number of clusters. The problem was that LCSS produced only two types of distances—very close to the minimum possible and ∞—and any outliers or well-separated clusters produced many infinite distances. Increasing ϵ improved the performance of the method for the freeway data sets but had the opposite effect on the intersection data sets and negatively affected the spectral clustering results.

Although the performance of DTW was not very good, we believe that this result is mostly due to the very strict constraints of (4) and (5) that require the starting points of the trajectories to be matched. Due to the unpredictability of target initialization in the tracker and the effects of foreshortening, trajectories in the same cluster could have varying starting and ending positions. The Hausdorff distance does not impose such a constraint; therefore, it has a clear advantage over DTW on data set 3 and, to a lesser extent, on data set 4. We were surprised by the poor results when using DTW in a spectral framework and have no definitive explanation for this observation; however, note that, on the data set with the largest intracluster variation in trajectory lengths, data set 1, DS, and DM did not perform well, whereas the same methods outperformed DA for data set 2, which exhibits the least variation in intracluster trajectory lengths.

B. Time Cost of Methods

Two significant factors affect the runtime of the compared methods: 1) the computation of a pairwise distance matrix and 2) the actual clustering. The asymptotic complexity of the first step is very similar for all of the compared methods; in practice, as Table III shows, the MH distance and LCSS have very similar runtimes, and DTW is slower by a constant factor. The reason for the longer runtime of DTW is that we do not use a band-limited version of the distance (the slack parameters δ for LCSS and w for the proposed distance reduce the runtime). All implementations of this component are in C++ but have not been heavily optimized.

TABLE III
RUNTIME OF THE METHODS (IN SECONDS)

Name	Distance	Type	Setting of k
HA	Hausdorff	Agglomerative	Manual
HS	Hausdorff	Spectral	Automatic
HM	Hausdorff	Spectral	Manual
DA	DTW	Agglomerative	Manual
DS	DTW	Spectral	Automatic
DM	DTW	Spectral	Manual
LA-ϵ	LCSS	Agglomerative	Manual
LS-ϵ	LCSS	Spectral	Automatic
LM-ϵ	LCSS	Spectral	Manual

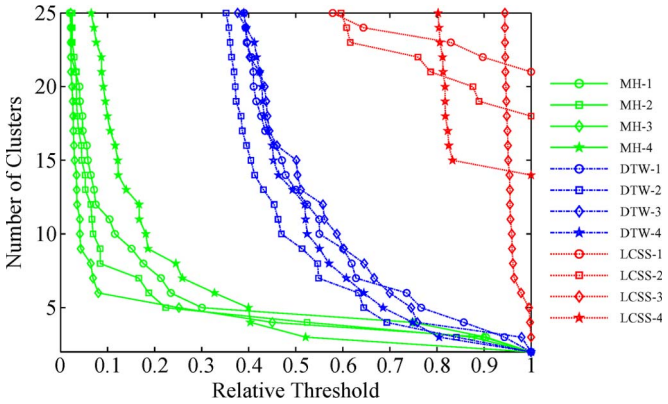


Fig. 7. Plot of the threshold in the agglomerative clustering versus the number of generated clusters. The MH, DTW, and LCSS distance with $\epsilon = 12$ are shown for the four data sets. The threshold is relative to the largest finite value obtained (i.e., the ∞ of LCSS is ignored).

The runtimes of spectral clustering and agglomerative clustering are comparable. Both times are overstated for different reasons. In the case of spectral clustering, we compute all eigenvalues and eigenvectors of $\mathbf{K}$, although in general, we only need a few of the top ones. This condition is a limitation of the dense eigensolver in Matlab. The agglomerative clustering is also implemented in Matlab but makes less use of built-in functions (e.g., `eig` and `kmeans`) and, as such, runs at a performance disadvantage.

C. Sensitivity of Agglomerative Clustering

Fig. 7 illustrates the behavior of the agglomerative clustering with the various distance measures. The width of the steps indicates the range over which the distance threshold can be varied without changing the number of clusters. Both DTW and the MH distance behave as expected, i.e., as the number of clusters gets smaller, the threshold range increases. The MH distance is very insensitive to the threshold when a few clusters are desired but is more sensitive than DTW for smaller thresholds. The erratic behavior of the LCSS distance is because the distances returned are either very small or infinite. As the plot shows, LCSS cannot merge clusters past a given point as all distances become infinite. Note that the steep portions of the plots for the MH distance that occur prior to the correct number of clusters is reached for each of the four data sets. This result means that the correct number of clusters may be easier to automatically detect using the Hausdorff distance by searching for the knee point on the cluster size versus the threshold graph.

VII. CONCLUSION AND FUTURE WORK

We have presented a trajectory-similarity measure based on the directed Hausdorff distance and applied it with both a spectral and an agglomerative clustering method. The imposition of some correspondence structure between the trajectories is a modification necessary to make the Hausdorff distance applicable for the comparison of ordered point sets, whereas the rejection of a certain amount of worst matches in the distance improves its robustness to noise and increases the tolerance to partial matches. Rather than using a distortion metric to select a globally optimal scale parameter, as [13] suggests, we used the metric to determine the optimal number of clusters from a set of possibilities obtained by examination of the spectrum of the affinity matrix. The local scale selection in [14] performed very well for the clustering of trajectories; hence, there was no need to perform a search for any scale parameters. One possible future refinement of the spectral methods would be to allow for outlier detection.

It should be possible to combine the stricter correspondence of the DTW distance with the ideas of the proposed Hausdorff distance, as the strict monotonicity of the matching seems preferable for time series. In general, the average distance over corresponding pairs of points used by DTW seemed more likely to be influenced by outliers than the robust trimmed maximum that we utilized. The relaxation of the starting/ending-point constraints in DTW can also improve the suitability of the distance for matching trajectories with differing lengths and allow for some partial matches. However, it is not clear that both the monotonicity constraint of (6) and the trimmed maximum of (8) can be combined in a method of complexity $O(mn)$ to compare an n-element and an m-element trajectory. We intend to investigate this question in the future.

The LCSS-based distance measure performed disappointingly, even on simpler data sets. Mostly, trajectories were considered very similar or very dissimilar, with no apparent middle case. This result would make the LCSS distance unsuitable for use in spectral clustering with local scaling as the relative distances would similarly be clustered at a few extreme values. Using a robust loss in the DTW distance should allow for some of the desirable qualities of LCSS to be emulated.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their valuable input and suggestions.

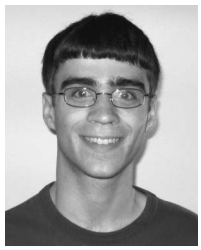
REFERENCES

[1] S. Atev, O. Masoud, and N. P. Papanikolopoulos, "Learning traffic patterns at intersections by spectral clustering of motion trajectories," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2006, pp. 4851–4856.
[2] H. Veeraraghavan, O. Masoud, and N. P. Papanikolopoulos, "Computer vision algorithms for intersection monitoring," *IEEE Trans. Intell. Transp. Syst.*, vol. 4, no. 2, pp. 78–89, Jun. 2003.
[3] S. Atev, H. Arumugam, O. Masoud, R. Janardan, and N. P. Papanikolopoulos, "A vision-based approach to collision prediction at traffic intersections," *IEEE Trans. Intell. Transp. Syst.*, vol. 6, no. 4, pp. 416–423, Dec. 2005.
[4] D. Buzan, S. Sclaroff, and G. Kollios, "Extraction and clustering of motion trajectories in video," in *Proc. Int. Conf. Pattern Recog.*, Aug. 2004, vol. 2, pp. 521–524.

[5] L. Chen, M. T. Özsu, and V. Oria, "Symbolic representation and retrieval of moving object trajectories," in *Proc. ACM SIGMM Int. Workshop Multimedia Inf. Retrieval*, Oct. 2004, pp. 227–234.
[6] L. Chen, M. T. Özsu, and V. Oria, "Robust and fast similarity search for moving object trajectories," in *Proc. ACM SIGMOD Int. Conf. Manag. Data*, 2005, pp. 491–502.
[7] F. I. Bashir, A. A. Khokhar, and D. Schoenfeld, "Segmented trajectory based indexing and retrieval of video data," in *Proc. IEEE Int. Conf. Image Process.*, Sep. 2003, vol. 2, pp. 623–626.
[8] M. Vlachos, D. Gunopulos, and G. Das, "Rotation invariant distance measures for trajectories," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2004, pp. 707–712.
[9] M. Vlachos, G. Kollios, and D. Gunopoulos, "Discovering similar multidimensional trajectories," in *Proc. 18th Int. Conf. Data Eng.*, Feb. 2002, pp. 673–684.
[10] F. Duchene, C. Garbay, and V. Rialle, "Similarity measure for heterogeneous multivariate time series," in *Proc. Eur. Signal Process. Conf.*, Sep. 2004, pp. 1605–1608.
[11] Z. Zhang, K. Huang, and T. Tan, "Comparison of similarity measures for trajectory clustering in outdoor surveillance scenes," in *Proc. Int. Conf. Pattern Recog.*, Aug. 2006, pp. 1135–1138.
[12] G. Antonini and J. P. Thiran, "Trajectories clustering in ICA space: An application to automatic counting of pedestrians in video sequences," in *Proc. Adv. Concepts Intell. Vis. Syst.*, Aug. 2004. [Online]. Available: http://infoscience.epfl.ch/record/87110/
[13] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Advances in Neural Information Processing Systems*, T. G. Dietterich, S. Becker, and Z. Ghahramani, Eds. Cambridge, MA: MIT Press, 2002, pp. 849–856.
[14] L. Zelnik-Manor and P. Perona, "Self-tuning spectral clustering," in *Advances in Neural Information Processing Systems*, L. K. Saul, Y. Weiss, and L. Bottou, Eds. Cambridge, MA: MIT Press, 2005, pp. 1601–1608.
[15] D. Verma and M. Meilă, "A comparison of spectral clustering algorithms," Dept. Comput. Sci. Eng., Univ. Washington, Seattle, WA, Tech. Rep. UW-CSE-03-05-01, 2003.
[16] K. Klpakis, D. Gada, and V. Puttagunta, "Distance measures for effective clustering of ARIMA time series," in *Proc. IEEE Int. Conf. Data Mining*, Nov. 2001, pp. 273–280.
[17] J. Alon, S. Sclaroff, G. Kollios, and V. Pavlovic, "Discovering clusters in motion time-series data," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, Jun. 2003, vol. 1, pp. 375–381.
[18] F. M. Porikli, "Learning object trajectory patterns by spectral clustering," in *Proc. IEEE Int. Conf. Multimedia Expo.*, Jun. 2004, vol. 2, pp. 1171–1174.
[19] Y. Xiong and D.-Y. Yeung, "Mixtures of ARMA models for model-based time series clustering," in *Proc. IEEE Int. Conf. Data Mining*, Dec. 2002, pp. 717–720.
[20] W. Hu, X. Xiao, D. Xie, T. Tan, and S. Maybank, "Traffic accident prediction using 3-D model-based vehicle tracking," *IEEE Trans. Veh. Technol.*, vol. 53, no. 3, pp. 677–694, May 2004.
[21] 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.
[22] D. R. Magee, "Tracking multiple vehicles using foreground, background, and motion models," *Image Vis. Comput.*, vol. 22, no. 2, pp. 143–155, Feb. 2004.
[23] J. W. Hsieh, S. H. Yu, Y. S. Chen, and W. F. Hu, "Automatic traffic surveillance system for vehicle tracking and classification," *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 2, pp. 175–187, Jun. 2006.
[24] B. Morris and M. Trivedi, "Robust classification and tracking of vehicles in traffic video streams," in *Proc. IEEE Conf. Intell. Transp. Syst.*, Sep. 2006, pp. 1078–1083.
[25] Z. Fu, W. Hu, and T. Tan, "Similarity-based vehicle trajectory clustering and anomaly detection," in *Proc. IEEE Int. Conf. Image Process.*, Sep. 2005, vol. 2, pp. 602–605.
[26] S. Atev, O. Masoud, R. Janardan, and N. P. Papanikolopoulos, "A collision prediction system for traffic intersections," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Aug. 2005, pp. 169–174.
[27] S. Atev and N. P. Papanikolopoulos, "Multiview vehicle tracking with constrained filter and partial measurement support," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2008, pp. 2277–2282.
[28] J. W. Hunt and T. G. Szymanski, "A fast algorithm for computing longest common subsequences," *Commun. ACM*, vol. 20, no. 5, pp. 350–353, May 1977.
[29] D. S. Hirschberg, "Algorithms for the longest common subsequence problem," *J. Assoc. Comput. Mach.*, vol. 24, no. 4, pp. 664–675, Oct. 1977.
[30] M.-P. Dubuisson and A. K. Jain, "A modified Hausdorff distance for object matching," in *Proc. Int. Conf. Pattern Recog.*, Oct. 1994, vol. 1, pp. 566–568.
[31] D. P. Huttenlocher, G. A. Klanderman, and W. J. Rucklidge, "Comparing images using the Hausdorff distance," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 15, no. 9, pp. 850–863, Sep. 1993.
[32] N. Meratnia and R. A. de By, "Aggregation and comparison of trajectories," in *Proc. ACM Int. Symp. Adv. Geographic Inf. Syst.*, Nov. 2002, pp. 49–54.
[33] M. Meilă, "Comparing clusterings—An information-based distance," *J. Multivariate Anal.*, vol. 98, no. 5, pp. 837–895, May 2005.

Stefan Atev received the B.A. degree in computer science and mathematics from Luther College, Decorah, IA, in 2003 and the M.S. degree in computer science from the University of Minnesota, Minneapolis, in 2007. He is currently working toward the Ph.D. degree with the Department of Computer Science and Engineering, University of Minnesota.

He is a Senior Algorithm Development Scientist with Vital Images, Inc., Minnetonka, MN. His research interests include computer vision, machine learning, and medical image processing.



Grant Miller received the B.S. degree in computer science from the University of Minnesota, Mineapolis, in 2009.

He was with the Department of Computer Science and Engineering, University of Minnesota, working on mobile applications. He is currently with Concrete Software Inc., Eden Prairie, MN. His research interests include computer vision and programming languages.

Mr. Miller received an honorable mention at the 2008 Computer Research Association Outstanding Undergraduate Award Competition.

Nikolaos P. Papanikolopoulos (F'07) received the Diploma degree in electrical and computer engineering from the National Technical University of Athens, Athens, Greece, in 1987 and the M.S.E.E. degree in electrical engineering and the Ph.D. degree in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, in 1988 and 1992, respectively.

He is currently a Distinguished McKnight University Professor with the Department of Computer Science and Engineering, University of Minnesota, Minneapolis, and the Director of the Security in Transportation Technology Research and Applications, Center for Transportation Studies. His research interests include computer vision, sensors for transportation applications, robotics, and control. He is the author or coauthor of more than 250 journal papers and conference proceedings in the aforementioned research areas.

Dr. Papanikolopoulos was a finalist for the Anton Philips Award for Best Student Paper at the 1991 IEEE International Conference on Robotics and Automation and the recipient of the Best Video Award at the 2000 IEEE International Conference on Robotics and Automation. Furthermore, he was the recipient of the Kritski Fellowship in 1986 and 1987. He was a McKnight Land Grant Professor with the University of Minnesota for the period 1995–1997 and has received the National Science Foundation (NSF) Research Initiation and Faculty Early Career Development Awards. He was also received the Faculty Creativity Award from the University of Minnesota. One of his papers (with O. Masoud as a coauthor) received the IEEE Vehicular Technology Society 2001 Best Land Transportation Paper Award. Finally, he has received grants from Defense Advanced Research Projects Agency, Department of Homeland Security, the U.S. Army, the U.S. Air Force, Sandia National Laboratories, NSF, Microsoft, Lockheed Martin, Idaho National Engineering and Environmental Laboratory, U.S. Department of Transportation, Minnesota Department of Transportation, Honeywell, and 3M, totaling more than $17 M.