Unsupervised Trajectory Clustering via Adaptive Multi-Kernel-based Shrinkage

Hongteng Xu[1,2], Yang Zhou[3], Weiyao Lin[3], Hongyuan Zha[1]
[1]College of Computing, [2]School of EE, Georgia Tech
[3]Department of EE, Shanghai Jiao Tong University
hxu42@gatech.edu, zhz128ly@sjtu.edu.cn, wylin@sjtu.edu.cn, zha@cc.gatech.edu

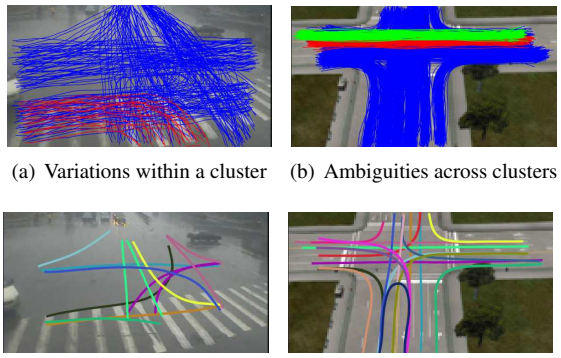
Abstract

This paper proposes a shrinkage-based framework for unsupervised trajectory clustering. Facing to the challenges of trajectory clustering, e.g., large variations within a cluster and ambiguities across clusters, we first introduce an adaptive multi-kernel-based estimation process to estimate the 'shrunk' positions and speeds of trajectories' points. This kernel-based estimation effectively leverages both multiple structural information within a trajectory and the local motion patterns across multiple trajectories, such that the discrimination of the shrunk point can be properly increased. We further introduce a speed-regularized optimization process, which utilizes the estimated speeds to regularize the optimal shrunk points, so as to guarantee the smoothness and the discriminative pattern of the final shrunk trajectory. Using our approach, the variations among similar trajectories can be reduced while the boundaries between different clusters are enlarged. Experimental results demonstrate that our approach is superior to the state-of-art approaches on both clustering accuracy and robustness. Besides, additional experiments reveal the effectiveness of our approach when applied to trajectory analysis applications, e.g., anomaly detection and route analysis.

1. Introduction

Trajectory clustering is fundamental in many applications including behavior analysis [25, 1, 24, 20], scene analysis [32, 22, 27, 28], and video surveillance [17, 11]. In many scenarios, due to the existence of large amount of trajectories and trajectory patterns, automatically clustering trajectories into suitable pattern groups without manually labeled data (i.e., unsupervised clustering) becomes essential [17, 11, 28].

Although trajectory itself records positions of object over time and contains significant information for clustering, directly perform clustering over these positions often creates poor results. Two reasons lead to the failure of the simple strategy. 1) *Variations within a cluster*: Due to the uncertain nature of object motion or the inaccuracy of trajectory extraction, trajectories from the same cluster may have large variations (e.g., the red trajectories in Fig. 1(a)[1]). This variation greatly decreases the similarity between trajectories in a cluster. 2) *Ambiguities across clusters*: In many scenarios, trajectories with similar shapes and close positions may belong to different clusters due to their underly semantic difference. For example, in Fig. 1(b), similar traffic trajectories belong to different clusters (labeled by red and green) since they are on different traffic lanes. As a result, the information of position is ambiguous on the boundary between clusters.



(a) Variations within a cluster (b) Ambiguities across clusters

(c) The shrinkage result of (a) (d) The shrinkage result of (b)

Figure 1. (a, b) The illustration of the challenges of trajectory clustering. (c, d) The shrunk trajectories and the clustering results obtained by our approach.

To overcome the problems above, we propose a shrinkage-based framework for unsupervised trajectory clustering. The proposed approach introduces an adaptive multi-kernel-based estimation process to estimate the *shrunk* positions and speeds of trajectory points, so as to reduce the variations of similar trajectories and enlarge the boundaries between different clusters. In order to maintain the information of the original trajectories, we further introduce a speed-regularized optimization process which lever-

[1]All the figures in this paper are best viewed in color.

ages the estimated speed to regularize the optimal shrunk positions. In this way, the optimal shrunk result can effectively inherit both the shape information in the original trajectories and the discriminative patterns obtained by the kernel-based estimation process. Applying the above two processes iteratively, trajectories are shrunk to the centers of their clusters, as in Fig. 1(c) and Fig. 1(d).

Our approach can be viewed as a combination of the mean shift clustering and the manifold-based model. The center of a cluster is represented by a manifold and its structure corresponds to a motion pattern. Trajectories belonging to a cluster are regarded as noisy traversals of a manifold, which contains structural information. Therefore, trajectory clustering means suppressing the noise in trajectories and reconstructing the manifolds. Taking advantages of the structural information of trajectories, we improve traditional mean shift algorithm via adaptive multi-kernel-based estimation and speed-based regularization. By shrinking trajectories with our approach, the clustering structures of trajectories are enhanced, making the information of trajectory positions more suitable for clustering.

The contributions of our work are three folds. First, we introduce an adaptive multi-kernel-based estimation process to estimate the shrunk positions and speeds of trajectory points. This kernel-based estimation process effectively leverages both multiple structural information within a trajectory and the local motion pattern across multiple trajectories. Second, we introduce a speed-regularized optimization process to further optimize the shrunk positions, so as to make the optimal shrunk result inherit both the shape information in the original trajectories and the discriminative patterns obtained by the kernel-based estimation process. Third, we establish a novel framework for addressing the unsupervised trajectory clustering problem, which iteratively performs position estimation and optimization to shrink trajectory points before clustering.

2. Related Works

Many works have been proposed on trajectory analysis and they can be categorized into supervised methods and unsupervised ones. Since supervised methods [14, 18] requires sufficient labeled trajectories to learn reliable models about trajectory patterns, they have limitations in handling scenarios with large amount of trajectories and trajectory patterns. Therefore, we focus on unsupervised methods in this paper, which find trajectory patterns by applying an unsupervised clustering over input trajectories.

Similarity Measuring. Many unsupervised clustering methods focus on developing effective similarity metrics to model the similarity between trajectories [13, 23, 3, 19]. In [23], a longest common subsequence (LCSS) distance is introduced, which is directly based on the analysis of objects' coordinates. Treating trajectories as time series, Chen and Ng [3] developed an edit distance with real penalty (ERP) to measure the similarity between trajectories. Lin et al. [15] introduce heat maps to describe trajectories and perform heat surface matching to measure trajectory similarities. Besides, Hu et al. [11] and Nawaz et al. [19] further represent and measure trajectory similarities in the Fourier or Wavelet domain. However, since trajectory patterns vary a lot in different scenarios, these methods still have limitations in adapting with different input trajectories.

Cluster Pattern Modeling. As many unsupervised processes perform clustering by iteratively finding and updating trajectory clusters [16, 12, 19, 2, 31], finding suitable ways to model cluster patterns in each iteration is essential. Cluster patterns can be described as trajectory distribution [12], cluster center [19], semantic path [2, 31], and model parameters [17, 11, 19]. In recent years, probabilistic models are widely utilized to model cluster patterns [6, 31, 17, 11]. In [6], the dynamics of trajectories is modeled by Gaussian processes. In [17], the points of trajectories are modeled by Gaussian mixture models and the dynamics of trajectories is modeled by hidden Markov models (HMM), which understands activity from the model of trajectories. A Dirichlet process mixture model (DPMM) is applied to trajectory clustering in [11], which can be extended to online clustering. Although these methods show satisfactory performances in some scenarios, their performances are often affected by the cluster initialization (or pre-clustering) results [17]. A poor cluster initialization result may limit the clustering accuracy of these methods.

Mean shift and Manifold-based Modeling. Mean shift algorithms [7, 4] have been widely applied to unsupervised clustering, which can be viewed as a manifold denoising algorithm [26]. On the other hand, manifold learning methods have also been used to cluster data from noisy observations. The early works in [21, 8, 5] learn structure of manifold from samples based on the assumption of smoothness. An outlier removal algorithm is proposed in [29] to deal with non-Gaussian noise. Multiple-manifold learning algorithms using curvature-based measure are proposed in [9, 30], which are useful to learn discriminative manifolds jointly and synthesize data. Recently, Wang et al. [26] combine traditional mean shift with smooth manifold learning and propose a manifold blurring mean shift algorithm for manifold denoising. However, none of these methods are developed focusing on trajectory clustering. Without fully leveraging the information of trajectories, the performance of these methods on trajectory clustering is limited.

3. Proposed Approach

Suppose that we have a set of trajectories $\{\boldsymbol{T}_i\}_{i=1}^{M}$, which corresponds to the motion processes of M objects. Each trajectory can be represented as $\boldsymbol{T}_i = [\boldsymbol{p}_{i,1}, ..., \boldsymbol{p}_{i,T}]^\top$, where $\boldsymbol{p}_{i,t} \in \mathbb{R}^D$ is the position point of the i-th object at time t in

the D-dimensional space. For each point $\boldsymbol{p}_{i,t}$, the speed of object at this point is estimated as $\boldsymbol{v}_{i,t} = (\boldsymbol{p}_{i,t+1} - \boldsymbol{p}_{i,t})/\delta t$, where δt is the time step. For convenience, we represent the speed of $\boldsymbol{T}_i$ as $\boldsymbol{V}_i = \boldsymbol{\Delta T}_i = [\boldsymbol{v}_{i,1}, ..., \boldsymbol{v}_{i,T}]^\top$. Suppose that there exist K discriminative motion patterns. Each trajectory $\boldsymbol{T}_i$ contains one of these motion patterns. What we want to do is to cluster these trajectories with high accuracy.

In this paper, we propose an adaptive multi-kernel-based estimation for the positions and the speeds of trajectories and further optimize the estimated points with the regularization based on the estimated speeds. Applying the two steps iteratively, we achieve suppressing the noise in the trajectories and shrinking them to the centers of their clusters accordingly, which attains a much more discriminative representation of original trajectories.

3.1. Adaptive Multi-Kernel-based Estimation

For each point $\boldsymbol{p}_{i,t} \in \boldsymbol{T}_i$, we first find its neighbors $\{\boldsymbol{p}_n\}_{n=1}^N$. Each $\boldsymbol{p}_n$ satisfies $\|\boldsymbol{p}_{i,t} - \boldsymbol{p}_n\|_2 \leq r$, where r is the radius of the neighborhood. Note that these neighbors might belong to other trajectories. Then we can obtain an estimate of $\boldsymbol{p}_{i,t}$ and the corresponding estimate of its speed $\boldsymbol{v}_{i,t}$ via nonparametric estimation as follows,

$$\hat{\boldsymbol{p}}_{i,t} = \frac{\sum_{n=1}^N K(\boldsymbol{p}_{i,t}, \boldsymbol{p}_n)\boldsymbol{p}_n}{\sum_{n=1}^N K(\boldsymbol{p}_{i,t}, \boldsymbol{p}_n)}, \tag{1}$$

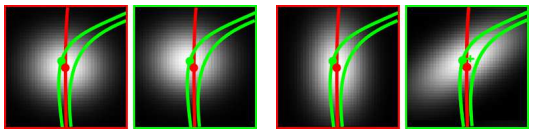
$$\hat{\boldsymbol{v}}_{i,t} = \frac{\sum_{n=1}^N K(\boldsymbol{p}_{i,t}, \boldsymbol{p}_n)\boldsymbol{v}_n}{\sum_{n=1}^N K(\boldsymbol{p}_{i,t}, \boldsymbol{p}_n)}, \tag{2}$$

Here, $K(\cdot, \cdot)$ is the kernel function, which is critical for the rationality of estimation. Different from previous works in [26, 10] merely using position or speed information to construct kernel function, we design a compound kernel for fully utilizing the structural information of trajectory from multiple aspects. Specifically, the proposed kernel is consist of three components, which is defined as follows: for $\boldsymbol{p}_{n'} \in \boldsymbol{T}_i$ and $\boldsymbol{p}_n \in \boldsymbol{T}_j$,

$$K(\boldsymbol{p}_{n'}, \boldsymbol{p}_n) = K_p(\boldsymbol{p}_{n'}, \boldsymbol{p}_n)K_s(\boldsymbol{p}_{n'}, \boldsymbol{p}_n)K_a(\boldsymbol{p}_{n'}, \boldsymbol{p}_n), \tag{3}$$

where

- $K_p(\boldsymbol{p}_{n'}, \boldsymbol{p}_n) = \exp\left(-\|\boldsymbol{p}_{n'} - \boldsymbol{p}_n\|_2^2/(2\alpha^2)\right)$ is a kernel measuring **the consistency of position**. Points with similar positions are close in the kernel space.
- $K_s(\boldsymbol{p}_{n'}, \boldsymbol{p}_n) = \exp\left(-\|\boldsymbol{v}_{n'} - \boldsymbol{v}_n\|_2^2/(2\beta^2)\right)$ is a kernel measuring **the consistency of speed**. Points with similar speeds are close in the kernel space.
- $K_a(\boldsymbol{p}_{n'}, \boldsymbol{p}_n) = \exp\left(-\|\boldsymbol{a}_i - \boldsymbol{a}_j\|_2^2/(2\gamma^2)\right)$, where $\boldsymbol{a}_i = [\boldsymbol{p}_{i,1}; \boldsymbol{p}_{i,T}]$ contains the source and the destination of the trajectory $\boldsymbol{T}_i$, is a kernel measuring **the consistency of aim**. The source and the destination of trajectory implies the aim of the trajectory, that is "where is it from and where will it go to". The position points belonging to the trajectories with similar aim should be close in the kernel space. Note that the aim of trajectory is flexible whose definition can be various according to different practical situations. The specific definition here is suitable for our work.



(a) Position based kernel K_p (b) Proposed $K = K_pK_vK_a$

Figure 2. The kernel function in the red (green) frame corresponds to the red (green) point on the red (green) trajectory. In the last subfigure, the green cross represents the estimated new positions of the green point. For a clear illustration, only few trajectories are shown and the kernel function is interpolated linearly.

Here we do not estimate positions and speeds independently based on K_p and K_v. With the help of the proposed kernel, the information of position and speed are fully adapted with each other — they leverage information with each other and share the information from the domain of aim. Note that we do not use high-order information of trajectory, i.e., the acceleration of point, because this information is very sensitive to noise, which is generally unreliable in practical situations. Fig. 2 illustrates the rationality and the superiority of our kernel function. It can be found that in the position with various crossing trajectories, the kernel merely based on the similarity of position does not have discriminative power — the close position points belonging to different trajectories have almost the same kernels. On the contrary, applying our kernel function, the kernels of the close position points are different, which along their own directions. Moreover, the estimated position shifts away from the ambiguous position. It tends to the position of the trajectory with the similar aim. Therefore the new position further increases the discriminative power.

The positions estimated by (1) formulate an estimated trajectories $\widehat{\boldsymbol{T}}_i = [\hat{\boldsymbol{p}}_{i,1}, ..., \hat{\boldsymbol{p}}_{i,T}]^\top$, $i = 1, ..., M$. Note that the estimate of speed $\widehat{\boldsymbol{V}}_i = [\hat{\boldsymbol{v}}_{i,1}, ..., \hat{\boldsymbol{v}}_{i,T}]^\top$ is not equal to $\boldsymbol{\Delta}\widehat{\boldsymbol{T}}_i$. It provides us with important information for optimizing the new trajectory.

3.2. Speed-regularized Trajectory Optimization

For each estimated trajectory we wish to preserve the relative positions among points so that the shape of the trajectory will not be too different from that of the original one. Some traditional methods can achieve this aim. For example, we can optimize the estimated trajectory with a regularization of differential smoothness (RDS) as follows,

$$\widetilde{\boldsymbol{T}}_i = \arg\min_{\boldsymbol{T}} \|\boldsymbol{T} - \widehat{\boldsymbol{T}}_i\|_F^2 + \lambda\|\boldsymbol{\Delta T}\|_F^2, \tag{4}$$

where the speed of the trajectory is energy-limited. Besides, we can also apply the manifold blurring mean shift (MBMS) algorithm [26], optimize the trajectory as follows,

$$\tilde{\boldsymbol{p}}_{i,t} = \boldsymbol{p}_{i,t} + (\boldsymbol{I} - \boldsymbol{U}_{i,t}\boldsymbol{U}_{i,t}^{\top})(\hat{\boldsymbol{p}}_{i,t} - \boldsymbol{p}_{i,t}), \tag{5}$$

where $\boldsymbol{U}_{i,t}$ contains the first L ($L < D$) eigenvectors of the matrix $\boldsymbol{P} = [\boldsymbol{p}_{i,t-d}, ..., \boldsymbol{p}_{i,t+d}]$ representing the context of $\boldsymbol{p}_{i,t}$ in $\boldsymbol{T}_i$. Here, the difference between the estimated trajectory and the original one is constrained to be orthogonal to the local tangent spaces of the proposed manifold. The final optimized trajectory is $\widetilde{\boldsymbol{T}}_i = [\tilde{\boldsymbol{p}}_{i,1}, ..., \tilde{\boldsymbol{p}}_{i,T}]^{\top}$.

The problem of the two methods above is merely utilizing the structural information of single trajectory. In the case of noisy trajectory, its structural information (i.e., its position and its speed) is unstable, which cannot provide us with sufficient prior information to optimize the trajectory itself. Fortunately, in the previous step, we have obtained the estimated speed of trajectory based on our adaptive multi-kernel estimation, which contains the structural information from other trajectories implicitly. Leveraging the estimated speed to regularize the optimization of trajectory, we obtain the final optimal trajectory by solving the following optimization problem,

$$\widetilde{\boldsymbol{T}}_i = \arg\min_{\boldsymbol{T}} \|\boldsymbol{T} - \widehat{\boldsymbol{T}}_i\|_F^2 + \lambda \|\boldsymbol{W}_i(\boldsymbol{\Delta}\boldsymbol{T} - \widehat{\boldsymbol{V}}_i)\|_F^2. \tag{6}$$

Here $\boldsymbol{W}_i = \mathrm{diag}(w_{i,1}, ..., w_{i,T})$, whose diagonal elements are calculated as $w_{i,t} = \|\hat{\boldsymbol{p}}_{i,t+1} - \hat{\boldsymbol{p}}_{i,t}\|_2$. The first term is the data fidelity term, which ensures that the final optimal trajectory $\widetilde{\boldsymbol{T}}_i$ should be close to the trajectory obtained by the adaptive multi-kernel-based estimation. The second term is a regularization term, which ensures that the speed of the final optimal trajectory $\widetilde{\boldsymbol{T}}_i$ should be close to the estimated speed $\widehat{\boldsymbol{V}}_i$. The significance of the regularization at various positions is adaptive, which is controlled by the factor $\lambda w_{i,t}^2$. For the positions having large changes, the significance of the speed-based regularization is high, which guarantees the smoothness of $\widetilde{\boldsymbol{T}}_i$. The optimization problem (6) is a regularized least-square problem. Denote $\boldsymbol{W}_i^{\top}\boldsymbol{W}_i$ as $\boldsymbol{\Gamma}_i$, the solution of (6) is

$$\widetilde{\boldsymbol{T}}_i = (\boldsymbol{I} + \lambda\boldsymbol{\Delta}^{\top}\boldsymbol{\Gamma}_i\boldsymbol{\Delta})^{-1}(\widehat{\boldsymbol{T}}_i + \lambda\boldsymbol{\Delta}^{\top}\boldsymbol{\Gamma}_i\widehat{\boldsymbol{V}}_i), \tag{7}$$

where $\boldsymbol{I}$ is an identity matrix.

3.3. Iterative Scheme and Clustering

Repeating the adaptive multi-kernel-based trajectory estimation and the speed-regularized trajectory optimization iteratively, the trajectories within the same cluster are shrunk together. As a result, we propose our adaptive multi-kernel-based shrinkage (AMKS) algorithm in Algorithm 1.

Ideally, the final shrunk trajectories $\{\widetilde{\boldsymbol{T}}_i\}_{i=1}^M$ should aggregate to the centers of clusters. These shrunk trajectories have very strong discriminative power, which can be used as features to cluster original trajectories. Applying simple clustering methods, i.e., k-means with Euclidean distance to the features, we can cluster trajectories with high accuracy, which will be shown in the section 4.

Algorithm 1 Adaptive Multi-Kernel-based Shrinkage

Input: Trajectories $\{\boldsymbol{T}_i\}_{i=1}^M$, number of iteration J.
Output: Shrunk trajectories $\{\widetilde{\boldsymbol{T}}_i\}_{i=1}^M$.

```
Initialize T_i^(0) = T_i, i = 1, ..., M.
for j = 1 : J do
    for i = 1 : M do
        Adaptive Multi-kernel Trajectory Estimation:
        Compute V_i = Δ T_i^(j-1).
        for t = 1 : T do
            Estimate p̂_{i,t} and v̂_{i,t} by (1, 2).
        end for
        Speed-regularized Trajectory Optimization:
        Given T̂_i, V̂_i and W_i, compute T_i^(j) by (7).
    end for
end for
T̃_i = T_i^(J), i = 1, ..., M.
```

3.4. Further Analysis

Advantages. Our algorithm can be viewed as an improvement of traditional methods. Specifically, if we only repeat the step of nonparametric trajectory estimation and use Gaussian kernel in (1), our algorithm is degraded to traditional mean shift (MS) algorithm [4]. If we use Gaussian kernel in (1) and replace the step of speed-regularized trajectory optimization (7) by the manifold blurring (5), our algorithm is equal to MBMS algorithm [26]. Compared with MS and MBMS, the main advantages of our algorithm include: 1) We apply an adaptive multi-kernel-based estimation process, which synthesizes multiple structural information of multiple trajectories. The adaptive multi-kernel-based estimation ensures that the useful information from other trajectories will be used while the harmful information will be rejected. 2) We introduce a speed-regularized trajectory optimization process to guarantee the smoothness and the discrimination of the optimized trajectories.

Computational Complexity and Acceleration. In each iteration, the computational complexity of our algorithm is $\mathcal{O}((MT)^2D + MT^3D)$, where the first term is for the step of adaptive multi-kernel-based trajectory estimation and the second is for solving the regularized least-square problem (7). Fortunately, focusing on trajectory clustering we may not need to update the neighbors of points at each iteration so that the first term is negligible in practice. Moreover, because the matrix $\boldsymbol{I} + \lambda\boldsymbol{\Delta}^{\top}\boldsymbol{W}_i^{\top}\boldsymbol{W}_i\boldsymbol{\Delta}$ is sparse and symmetric, the computational complexity of the second term

can be reduced to $\mathcal{O}(MTD)$. Therefore, the computational complexity of our algorithm is $\mathcal{O}(MTD)$, which is at most equal to that of MBMS. In the following content, we call our algorithm updating neighborhoods iteratively as "AMKS" and call the accelerated algorithm computing neighborhoods only once as "FastAMKS". Fig. 3 shows the performance of acceleration using "FastAMKS". We can find that the run time of "FastAMKS" w.r.t. the number of points of trajectories (MT) increases much more slowly than that of original "AMKS" does. Moreover, experimental results in the next section will show that the performance of "FastAMKS" on clustering is comparable, even better than that of "AMKS".

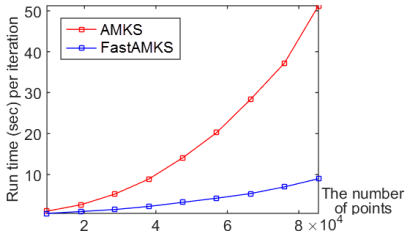


Figure 3. Comparison of our AMKS and FastAMKS algorithm on run time w.r.t. the number of points of trajectories MT.

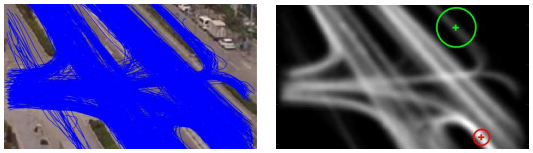
Configuration of Parameters. Our algorithm has six significant parameters: the radius of neighborhood r, the three bandwidths $\{\alpha, \beta, \gamma\}$ of kernels, the significance of regularization term λ and the number of iteration J. Different from traditional methods [4, 21, 5, 26, 9], we apply an adaptive strategy to configure these parameters. Specifically, given trajectories $\{\boldsymbol{T}_i\}_{i=1}^{M}$, we can compute a density map for points of trajectories as follows: compute the histogram of points of trajectories $\boldsymbol{H} = [H(x_1, .., x_D)]$ as

$$H(x_1, .., x_D) = |\{\boldsymbol{p}_{i,t} \in [x_1, x_1{+}1) \times .. \times [x_D, x_D{+}1)\}|,$$

where $|\cdot|$ calculates the cardinality of set; then a smoothed density map $\widehat{\boldsymbol{H}} = [\widehat{H}(x_1, ..., x_D)]$ is computed by Gaussian filtering $\widehat{\boldsymbol{H}} = \boldsymbol{G} * \boldsymbol{H}$, where $*$ is convolution and $\boldsymbol{G}$ is a Gaussian filter. Fig. 4 gives an example of the 2D case. For $\boldsymbol{p}_{i,t} \in [x_1, x_1 + 1) \times ... \times [x_D, x_D + 1)$, we select r as

$$r_{i,t} = (C/\widehat{H}(x_1, ..., x_D))^{1/D}, \tag{8}$$

where C is a constant. In our work, we reduce C from 200 to 10 linearly with the increase of iteration. (8) ensures that the number of points in the neighborhood to be relatively invariant to the change of position. As Fig. 4(b) shows, the point with high density (i.e., the red cross) has small neighborhood while the one with low density (i.e., the green cross) has large neighborhood. Given $r_{i,t}$, the bandwidths of kernels are set accordingly, where $\alpha = r_{i,t}/3$ and $\beta = \gamma = 2r_{i,t}/3$. Such a configuration ensures the influence of the points out of the neighborhood can be ignored.



(a) Traffic trajectories (b) Smoothed density map $\widehat{\boldsymbol{H}}$

Figure 4. (a) A set of traffic trajectories. (b) The smoothed density map. The two crosses represent two points of trajectories and their corresponding neighborhoods are shown as circles.

Finally, the significance of regularization term $\lambda = 1$ and the number of iteration $J = 7$ empirically in our work. In the experimental results, we will show that the performance of our algorithm is robust to the change of λ and the shrunk trajectories converge stably after 7 iterations.

4. Experiments

We use four data sets in our experiments: **T19:** the traffic trajectory data set in [17], which contains 1900 trajectories. The trajectories are clustered into 19 groups, each of which has 100 trajectories. The time length $T = 50$. The noise is moderate and there are few abnormal trajectories. **T11:** the traffic trajectory data set in [27], which contains 220 trajectories. The trajectories are clustered into 11 groups, each of which has 20 trajectories. The time length $T = 50$. The number of samples is small and the noise is serious. **T15:** the traffic trajectory data set in [11], which contains 1500 very noisy trajectories. The trajectories are clustered into 15 groups. The number of trajectories in a group is from 18 to 278, which is very unbalanced. **NY:** the pedestrian tracklet data in New York Train Station [31, 32], which contains over 40000 tracklets, rather than complete trajectories.

For demonstrating the effectiveness and the robustness of our approach, we also consider the noisy and the incomplete versions of the data sets (T19, T11 and T15 only). **Noisy case:** We add Gaussian noise $\mathcal{N}(0, \sigma^2)$ to the original trajectories. The standard deviation $\sigma = 0.03R,\ 0.05R$, respectively, where R is the dynamic range of trajectories. **Incomplete case:** Following [11], we select 20% trajectories randomly from each cluster and omit their first or last G points. G is set to be 10% (slight incompleteness) and 40% (serious incompleteness), respectively.

Following methods are tested. 1) *Similarity-measure-based methods*: clustering raw trajectories by K-means or spectral clustering with the Euclidean distance or the DTW distance ("**ED+Kmeans**", "**ED+SC**", "**DTW+Kmeans**", "**DTW+SC**"). 2) *Cluster-pattern-modeling methods*: the "**Heat-map**" method in [15]; the "**HMM**" based method in [17]; and the "**DPMM**" in [11]. 3) *Mean shift-based methods*: the mean shift-based clustering "**MS**" in [4]; the manifold blurring mean shift algorithm "**MBMS**" in [26]; our "**AMKS+Kmeans**" and "**FastAMKS+Kmeans**" approaches.

Table 1. Clustering Accuracy on The Data Set T19 (%)

Method	Clean	σ=0.03R	σ=0.05R	G=10%	G=40%
ED+Kmeans	88.9	88.2	86.8	83.2	73.4
DTW+Kmeans	92.1	88.7	83.3	88.2	77.7
ED+SC	84.2	84.2	84.2	81.6	72.2
DTW+SC	95.6	89.5	86.8	89.6	75.7
Heat-map [15]	95.0	91.8	88.5	91.8	67.9
HMM [17]	96.8	89.6	83.8	90.5	75.5
DPMM [11]	98.0	—	—	—	—
MS [4]	98.4	98.0	97.5	96.0	80.0
MBMS [26]	98.6	98.2	97.8	94.1	80.0
FastAMKS+Kmeans	**99.5**	**99.5**	**99.4**	**96.2**	**82.4**
AMKS+Kmeans	**99.5**	**99.5**	**99.4**	95.1	82.1

Table 2. Clustering Accuracy on The Data Set T11 (%)

Method	Clean	σ=0.03R	σ=0.05R	G=10%	G=40%
ED+Kmeans	90.5	86.4	85.8	87.2	80.3
DTW+Kmeans	90.0	87.7	86.9	87.3	74.1
ED+SC	86.3	86.3	86.3	85.9	76.4
DTW+SC	90.9	90.9	90.9	90.9	81.8
Heat-map [15]	91.8	85.4	84.9	88.0	75.0
HMM [17]	86.3	85.4	85.4	85.9	75.0
MS [4]	95.5	95.5	95.0	95.1	83.6
MBMS [26]	97.7	97.3	96.4	96.8	80.5
FastAMKS+Kmeans	**99.1**	**99.1**	98.6	**98.6**	**87.3**
AMKS+Kmeans	**99.1**	**99.1**	**99.1**	**98.6**	85.4

Table 3. Clustering Accuracy on The Data Set T15 (%)

Method	Clean	σ=0.03R	σ=0.05R	G=10%	G=40%
ED+Kmeans	82.6	78.8	77.0	80.8	72.3
DTW+Kmeans	83.2	80.0	78.8	81.1	71.8
ED+SC	85.0	81.2	81.2	84.4	76.1
DTW+SC	85.3	81.9	74.6	85.1	77.5
Heat-map [15]	82.0	78.9	71.8	82.0	**78.1**
HMM [17]	84.4	80.1	78.9	84.0	74.1
DPMM [11]	86.7	83.3	81.5	**86.1**	**78.1**
MS [4]	85.3	84.6	84.2	84.4	76.6
MBMS [26]	86.6	85.4	84.3	84.0	75.8
FastAMKS+Kmeans	**87.6**	**86.0**	**84.7**	84.4	**78.1**
AMKS+Kmeans	87.4	85.6	**84.7**	84.4	78.0

4.1. Trajectory Clustering

The Number of Clusters. When the number of clusters is not pre-determined, we can estimate the number of clusters by applying mean shift clustering algorithm to the shrunk trajectories. Specifically, we follow the strategy in [11]: after obtaining shrunk trajectories, we randomly select 30 subsets from trajectories. The number of trajectories in each subset is more than a half of the total number of shrunk trajectories. The mean shift clustering algorithm clusters the trajectories in each subset. The averaged number of the learned clusters is used to estimate the real number of clusters. Fig. 5 shows the experimental results on the three data sets, where the red squares correspond to the estimated number of clusters w.r.t. the real number of clusters. We can find that the estimated number of clusters fits the real number of clusters well in most situations.

Clustering Accuracy. We compare various trajectory clustering approaches on the three data sets. The real number of clusters is pre-determined. The experimental results

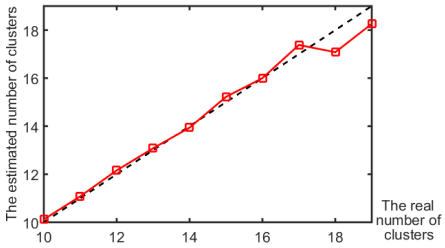


Figure 5. The estimated number of clusters v.s. the ground truth

of clustering accuracy are shown in Tables 1-3[2]. From Tables 1-3, we have following observations:

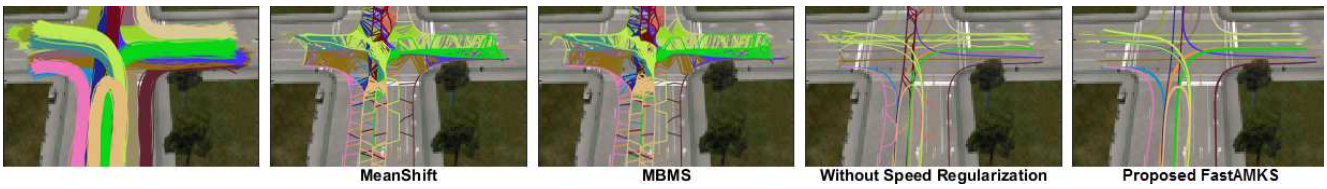
1) The results of our approach (AMKS+Kmeans and FastAMKS+Kmeans) is superior to other competitors in most situations. Especially in the noisy cases, since our approach is essentially related to manifold denoising, it is very robust to the noise — the clustering accuracy is almost unchanged with the increase of noise. Besides, an interesting result is the accelerated algorithm FastAMKS sometimes is equal to, even a little better than AMKS. In our opinion, that is because the initial structure of neighborhoods has few changes w.r.t iterations in the case of clean data. In the noisy or incomplete cases, the changes of the neighborhoods caused by shrinkage may introduce some errors. Therefore, the neighborhoods obtained from original trajectories may be more reliable.

2) The similarity-measure-based methods (ED+Kmeans, ED+SC, DTW+Kmeans, DTW+SC) achieve relatively low accuracies in datasets with larger variations (e.g., T11 and T15). This reveals that pre-defined similarity measures have limitations in adaptively handling different trajectory data. Comparatively, our approach has stronger capabilities in handing various trajectory data with the inclusion of adaptive multi-kernel estimation.

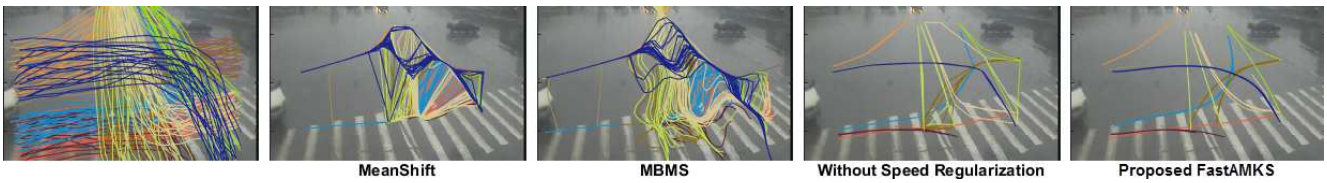
3) Our approach also has better or similar results to the cluster-pattern-modeling methods (Heat-map, HMM, DPMM). This further demonstrates the effectiveness of our framework. Moreover, note that the results of cluster-pattern-modeling methods are highly dependent on the cluster initialization results (e.g., the HMM based method [17] learns its HMM model based on a spectral pre-clustering result, the DPMM based method [11] learns its initial model from pre-selected representative trajectories). The results of these methods may be obviously decreased when their initial clustering results are less satisfactory. In contrast, our approach aims to enhance the discrimination among trajectories before clustering, and thus is more robust to the variation of initialization.

4) Our approach is also superior to the mean shift-based methods (MS, MBMS). This is because: (a) We apply an adaptive multi-kernel estimation to ensure the embedding of useful information from multiple trajectories and mul-

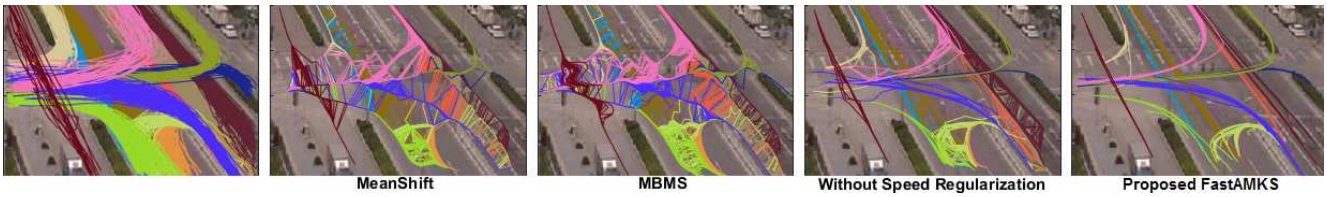
[2]Some experimental results of DPMM [11] are unavailable because of the lack of source code.



(a) T19: original trajectories and the shrunk trajectories obtained by MS [4], MBMS [25], adaptive multi-kernel estimation and our FastAMKS.

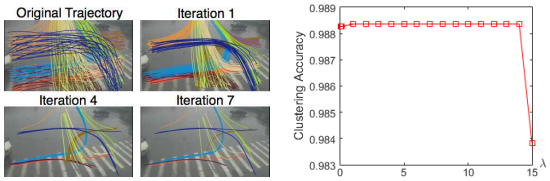


(b) T11: original trajectories and the shrunk trajectories obtained by MS [4], MBMS [25], adaptive multi-kernel estimation and our FastAMKS.



(c) T15: original trajectories and the shrunk trajectories obtained by MS [4], MBMS [25], adaptive multi-kernel estimation and our FastAMKS.

Figure 6. Visualization and comparison of shrunk trajectories. The color of trajectory indicates the cluster it belongs to.



(a) The shrunk trajectories w.r.t. the number of iteration. (b) λ v.s. clustering accuracy.

Figure 7. The robustness of our algorithm on the data set T11.

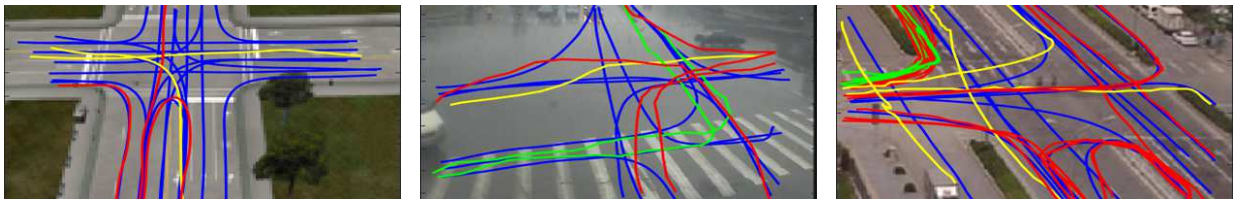
tiple aspects. (b) We leverage the information from other trajectories in the trajectory optimization process, so as to guarantee the smoothness and the discrimination of the optimized trajectories.

Stability of Trajectory. Besides the numerical results in the Tables and the examples in Fig. 1, we gives more visual results in Fig. 6. We compare the shrunk trajectories obtained by MS [4], MBMS [25], the adaptive multi-kernel-based estimation without speed-based regularization and our FastAMKS, respectively. The shrunk trajectories obtained by MS are very unstable. Many of shrunk trajectories contains discontinuous points so that the trajectories are shared by different clusters. MBMS improves the performance of MS a little — the shrunk trajectories are more smooth but still cannot preserve the shape of original ones. The instability of shrunk trajectories has negative influences on clustering accuracy. Replacing the Gaussian kernel used in MS and MBMS with our kernel (3), we can suppress the instability of shrunk trajectories greatly. However, without our speed-based regularization, there still exist some discontinuous points. Combining the adaptive multi-kernel-based estimation and the speed-based regularization, our FastAMKS algorithm obtains much more smooth and discriminative shrunk trajectories than its three competitors.

Robustness of Parameters. We also demonstrate the robustness of our FastAMKS algorithm to the number of iteration J and the significance of regularization term λ. The experimental results of the data set T11 are shown in Fig. 7. Specifically, Fig. 7(a) shows that the shrunk trajectories converges well after 7 iterations. Fig. 7(b) shows that the clustering accuracy is robust to the change of λ in a wide range from 0.01 to 14. Similar results can also be verified on the other data sets.

4.2. Anomaly Detection

One of useful applications in company with trajectory clustering is anomaly detection. The trajectories that are very different from their centers of clusters are regarded as abnormal trajectories. In our work, the mean of the shrunk trajectories belonging to the same cluster is regarded as the center of the cluster, denoted as $\{\boldsymbol{C}_k\}_{k=1}^{K}$ and shown in Fig. 8. Inspired by [14], we propose a simple anomaly detection method based on our clustering results and the excess kurtosis-based abnormality measure. For each original trajectory $\boldsymbol{T}_i$ belonging to the k-th cluster, we can compute the residual between $\boldsymbol{T}_i$ and the center of the cluster, de-



(a) The centers of clusters and anomaly in T19. (b) The centers of clusters and anomaly in T11. (c) The centers of clusters and anomaly in T15.

Figure 8. Visualization of centers of clusters and abnormal trajectories. The blue lines are centers of clusters obtained by our algorithm. The green lines are trajectories occupying the sidewalk. The yellow lines are trajectories changing lane unusually. The red lines are trajectories traveling in opposite direction.

(a) Original tracklets in NY. (b) Clustering results of shrunk tracklets. (c) The proposed graph and inferred paths.

Figure 9. In (b), given clustering results of shrunk tracklets, we can find the centric points of clusters and construct graph via detecting the density of points in the region between points. In (c), the proposed graph (red dots and lines) is shown. The colorful background shows the typcial semantic regions inferred in [31], and the yellow arrows shows the shortest paths connecting corresponding dots.

noted as $\boldsymbol{E}_{i,k} = \boldsymbol{C}_k - \boldsymbol{T}_i = [\boldsymbol{e}^1_{i,k}, ...\boldsymbol{e}^D_{i,k}] \in \mathbb{R}^{T\times D}$. Assuming that $\{\boldsymbol{e}^d_{i,k}\}^D_{d=1}$ are independent with each other, we compute the excess kurtosis of the residual, $\kappa_{i,k}$, as

$$\kappa_{i,k} = \frac{1}{D}\sum_{d=1}^{D}\kappa(\boldsymbol{e}^d_{i,k}) = \frac{1}{D}\sum_{d=1}^{D}\left(\frac{\mu^4(\boldsymbol{e}^d_{i,k})}{\sigma^4(\boldsymbol{e}^d_{i,k})} - 3\right), \quad (9)$$

where $\mu^4(\boldsymbol{e}^d_{i,k})$ and $\sigma(\boldsymbol{e}^d_{i,k})$ are the fourth moment and the standard deviation of $\boldsymbol{e}^d_{i,k}$. In any case, $-2 < \kappa < \infty$. If $\kappa < 0$, the distribution of the residual is broad and the peak is small. On the contrary, if $\kappa > 0$, the distribution of the residual is compact and the peak is obvious. In the case that $\kappa = 0$, the distribution is almost similar to normal distribution. Assume that the distribution of the residual between a normal trajectory and the center of its cluster is the Gaussian with zero mean and small variance. The trajectory with $\kappa < 0$ are likely to be anomaly.

Setting a threshold, we detect the trajectory whose excess kurtosis below the threshold as an abnormal trajectory. In our work, we set the threshold as -1.5. The results on T19, T11 and T15 are shown in Fig. 8, which verifies the feasibility of our method. The abnormal trajectories, e.g., unusual change of lane, occupation of sidewalk and travel in opposite direction, are found by our method.

4.3. Tracklet Clustering and Route Analysis

Our method can also be used to cluster tracklets and analyze routes in the scene. Applying our AMKS+Kmeans algorithm on the data set NY, we obtain shrunk tracklets and their clustering results. Compared with original tracklets in Fig. 9(a), the shrunk tracklets are aggregated tightly in various local regions. The clustering results in Fig. 9(b) show that our method clusters tracklets based on their local structures. Given the clustering results, we can analyze the routes in the scene effectively. For the shrunk tracklets in the same cluster, we compute the mean of points in the cluster, as the red dots in Fig. 9(b). These dots indicate the "popular" locations that many tracklets (pedestrians) pass. For each two dots, if there are dense points in the region between them, as the red ellipse in Fig. 9(b) shows, there exists an edge connecting them. As a result, we construct a graph connecting these dots. The shortest paths in the graph indicate popular routes on which there are a lot of pedestrians, as Fig. 9(c) shows. Comparing our paths with the semantic regions inferred in [31], we can find that a lot of them are similar, which verifies the rationality of our result.

5. Conclusion

In this paper, we propose an effective and robust trajectory clustering approach, which extends mean shift and manifold denoising algorithms and achieves outstanding results. Experimental results show that our approach can be used to cluster trajectories, detect anomaly and analyze scene, which is not only robust to the noise and the incompleteness of data but also insensitive to the changes of parameters. In the future, we plan to extend this analytic work to a predictive model. We will also explore other applications, e.g., action recognition and scene understanding.

Acknowledgement. This work is supported in part by NSF (DMS-1317424), NSFC (No. 61471235, U1201255), and Chinese National 973 Grants (2013CB329603).

References

[1] S. Ali and M. Shah. Floor fields for tracking in high density crowd scenes. In *ECCV*, pages 1–14. Springer, 2008. 1

[2] B. Cancela, A. Iglesias, M. Ortega, and M. Penedo. Unsupervised trajectory modelling using temporal information via minimal paths. In *CVPR*, pages 2553–2560. IEEE, 2014. 2

[3] L. Chen and R. Ng. On the marriage of lp-norms and edit distance. In *VLDB*, pages 792–803. VLDB Endowment, 2004. 2

[4] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(5):603–619, 2002. 2, 4, 5, 6, 7

[5] P. Dollár, V. Rabaud, and S. Belongie. Learning to traverse image manifolds. *NIPS*, 19:361, 2007. 2, 5

[6] D. Ellis, E. Sommerlade, and I. Reid. Modelling pedestrian trajectory patterns with gaussian processes. In *ICCV Workshops*, pages 1229–1234. IEEE, 2009. 2

[7] K. Fukunaga and L. Hostetler. The estimation of the gradient of a density function, with applications in pattern recognition. *Information Theory, IEEE Transactions on*, 21(1):32–40, 1975. 2

[8] A. Goh and R. Vidal. Segmenting motions of different types by unsupervised manifold clustering. In *CVPR*, pages 1–6. IEEE, 2007. 2

[9] D. Gong, X. Zhao, and G. Medioni. Robust multiple manifolds structure learning. In *ICML*, pages 321–328, 2012. 2, 5

[10] M. Hu, S. Ali, and M. Shah. Detecting global motion patterns in complex videos. In *ICPR*, pages 1–5. IEEE, 2008. 3

[11] W. Hu, X. Li, G. Tian, S. Maybank, and Z. Zhang. An incremental dpmm-based method for trajectory clustering, modeling, and retrieval. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(5):1051–1065, 2013. 1, 2, 5, 6

[12] N. Johnson and D. Hogg. Learning the distribution of object trajectories for event recognition. *Image and Vision computing*, 14(8):609–615, 1996. 2

[13] E. J. Keogh and M. J. Pazzani. Scaling up dynamic time warping for datamining applications. In *SIGKDD*, pages 285–289. ACM, 2000. 2

[14] K. Kim, D. Lee, and I. Essa. Gaussian process regression flow for analysis of motion trajectories. In *ICCV*, pages 1164–1171. IEEE, 2011. 2, 7

[15] W. Lin, H. Chu, J. Wu, B. Sheng, and Z. Chen. A heat-map-based algorithm for recognizing group activities in videos. *Circuits and Systems for Video Technology, IEEE Transactions on*, 23(11):1980–1992, 2013. 2, 5, 6

[16] S. Lloyd. Least squares quantization in pcm. *Information Theory, IEEE Transactions on*, 28(2):129–137, 1982. 2

[17] B. T. Morris and M. M. Trivedi. Trajectory learning for activity understanding: Unsupervised, multilevel, and long-term adaptive approach. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(11):2287–2301, 2011. 1, 2, 5, 6

[18] J. C. Nascimento, M. A. Figueiredo, and J. S. Marques. Trajectory classification using switched dynamical hidden markov models. *Image Processing, IEEE Transactions on*, 19(5):1338–1348, 2010. 2

[19] T. Nawaz, A. Cavallaro, and B. Rinner. Trajectory clustering for motion pattern extraction in aerial videos. In *ICIP*, pages 1016–1020. IEEE, 2014. 2

[20] P. Ochs, J. Malik, and T. Brox. Segmentation of moving objects by long term video analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 36(6):1187–1200, 2014. 1

[21] J. Park, Z. Zhang, H. Zha, and R. Kasturi. Local smoothing for manifold learning. In *CVPR*, volume 2, pages 452–459. IEEE Computer Society, 2004. 2, 5

[22] F. Shi, Z. Zhou, J. Xiao, and W. Wu. Robust trajectory clustering for motion segmentation. In *ICCV*, pages 3088–3095. IEEE, 2013. 1

[23] M. Vlachos, G. Kollios, and D. Gunopulos. Discovering similar multidimensional trajectories. In *ICDE*, pages 673–684. IEEE, 2002. 2

[24] H. Wang and C. Schmid. Action recognition with improved trajectories. In *ICCV*, pages 3551–3558. IEEE, 2013. 1

[25] J. M. Wang, D. J. Fleet, and A. Hertzmann. Gaussian process dynamical models for human motion. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(2):283–298, 2008. 1, 7

[26] W. Wang and M. A. Carreira-Perpinán. Manifold blurring mean shift algorithms for manifold denoising. In *CVPR*, pages 1759–1766. IEEE, 2010. 2, 3, 4, 5, 6

[27] W. Wang, W. Lin, Y. Chen, J. Wu, J. Wang, and B. Sheng. Finding coherent motions and semantic regions in crowd scenes: A diffusion and clustering approach. In *ECCV*, pages 756–771. Springer, 2014. 1, 5

[28] X. Wang, K. T. Ma, G. Ng, and E. Grimson. Trajectory analysis and semantic region modeling using nonparametric bayesian models. *International Journal of Computer Vision*, 96:287–321, 2011. 1

[29] H. Xu and H. Zha. Manifold based face synthesis from sparse samples. In *ICCV*, pages 2208–2215. IEEE, 2013. 2

[30] H. Xu, H. Zha, M. Davenport, et al. Manifold based dynamic texture synthesis from extremely few samples. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 3019–3026. IEEE, 2014. 2

[31] B. Zhou, X. Wang, and X. Tang. Random field topic model for semantic region analysis in crowded scenes from tracklets. In *CVPR*, pages 3441–3448. IEEE, 2011. 2, 5, 8

[32] B. Zhou, X. Wang, and X. Tang. Understanding collective crowd behaviors: Learning a mixture model of dynamic pedestrian-agents. In *CVPR*, pages 2871–2878. IEEE, 2012. 1, 5