Eurographics Conference on Visualization (EuroVis) 2013
B. Preim, P. Rheingans, and H. Theisel
(Guest Editors)

Volume 32 (2013), Number 3

Vector Field *k*-Means: Clustering Trajectories by Fitting Multiple Vector Fields

Nivan Ferreira[1], James T. Klosowski[2], Carlos E. Scheidegger[2], Cláudio T. Silva[1]

[1]Polytechnic Institute of New York University, New York, USA
[2]AT&T Labs Research, New Jersey, USA

Abstract

Scientists study trajectory data to understand trends in movement patterns, such as human mobility for traffic analysis and urban planning. In this paper, we introduce a novel trajectory clustering technique whose central idea is to use vector fields to induce a notion of similarity between trajectories, letting the vector fields themselves define and represent each cluster. We present an efficient algorithm to find a locally optimal clustering of trajectories into vector fields, and demonstrate how vector-field k-means can find patterns missed by previous methods. We present experimental evidence of its effectiveness and efficiency using several datasets, including historical hurricane data, GPS tracks of people and vehicles, and anonymous cellular radio handoffs from a large service provider.

Categories and Subject Descriptors (according to ACM CCS): I.5.3 [Pattern Recognition]: Clustering—Algorithms

1. Introduction

For many years, scientists have gathered and studied trajectory data to understand trends in movement patterns. Ecologists study animal movements to learn about population growth, social interactions, feeding and migratory patterns, etc. [BPAK04, GJL*09]. Meteorologists use trajectory data to help predict storm paths [Els03, CRG*07], and researchers from a wide variety of fields study human mobility to perform targeted advertising, predict traffic and commuting patterns, as well as data-driven urban planning [BDE09].

The recent ubiquity of GPS and RFID devices has caused a rapid increase in the amount of available trajectory data. These devices have been used to determine locations of animals, shipping containers, and vehicles. Even in the absence of explicit tracking devices, crowdsourcing can be used as an alternative to gather similar data [FLF*11]. Another option involves cellular phone handoffs: the traces of calls as they are handed from one cellphone tower to another [BCH*11, PXQ*11, BDE09]. This approach can greatly simplify and automate the data acquisition while still providing complete anonymity for individuals. In all such cases, due to the vast amount of data being collected, there is a great need for scalable and efficient techniques [GLW12].

The analysis of this kind of data is challenging not only because of its size, but also due to its complexity [RPN*08]. Trajectories are spatiotemporal in nature, involving geometric positions, directions, velocities, durations, and other characteristics specific to the entities being tracked. Hurricane tracks may include overall storm strength, wind speeds, or seasonality. Animal tracks may be influenced by their size, age, or gender. Incorporating these characteristics, when available, can help direct the trajectory analysis, but adds complexity.

Clustering techniques are general tools for data analysis that are also valuable for visualizing trajectory data by building summaries that reduce overplotting [TSAA12, WYCM11] and for visual analytics by revealing hidden patterns to be further investigated and visualized [RPN*08]. There are two main classes of algorithms for trajectory clustering. The first class corresponds to methods that embed the data in a metric space or define similarity notions and use ordinary point-based clustering techniques. The major drawback of these approaches is the difficulty in defining a similarity between trajectories, which results in undesired features [LHW07] or ignoring important components of the trajectory [GLW12]. The second class includes model-based approaches in which a generative model of the data is assumed and a clustering is indirectly obtained by estimating the model parameters. An example of this class is the work by Wei et al. [WYCM11], in which trajectories are approximated by fitting polynomials.

In this work, we present a model-based trajectory clustering approach that is based on vector field fitting. Intuitively,

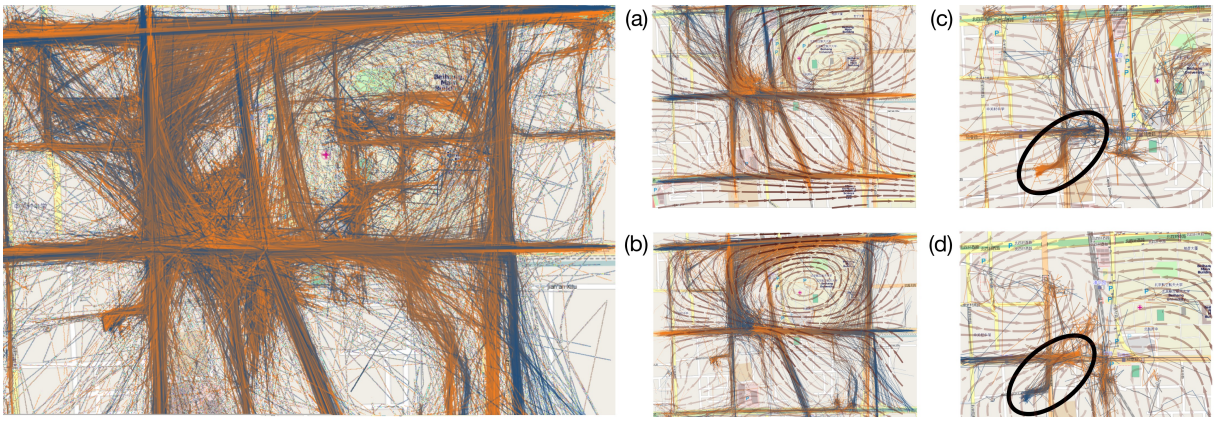


Figure 1: *The GeoLife Trajectories dataset clustered using vector field k-means. The original trajectories were cropped to a 10 block area in downtown Beijing. The orientation of each trajectory is represented by linearly interpolating from blue (start) to orange (end). This color scheme is used throughout the paper. We partition the data into four clusters ($k = 4$), and then subdivide each cluster resulting in 16 subclusters. Images (a) and (b) show two clusters from the first-level subdivision, and images (c) and (d) show two clusters from the second level. The first two vector fields show trajectories in patterns of faster vehicular traffic, while the latter appears to show pedestrian traffic moving to and from a lunch spot near the Microsoft Research Asia campus.*

the idea behind the method we call vector field *k*-means is to induce a similarity notion on the dataset by considering two trajectories similar if they can be approximated well by streamlines of a single vector field. This model solves the two main drawbacks of previous methods, namely that we can define a similarity that naturally encodes features of the trajectories together with their geometries. An example of our method used on a GPS track dataset is shown in Fig. 1.

One important consequence of our modeling approach that is not present in previous work is the ability to capture *global patterns* in the data that are not evident when considering only local information. For example, consider the synthetic trajectories in Fig. 2, which consists of two circular patterns each with 1000 trajectories. None of the trajectories actually form a complete circle, but vector field *k*-means is still able to recover the two separate circular patterns. Previous methods fail on this task tending to group both centers together, see Section 5.5. Furthermore, the obtained vector fields are a good summary of the clustered data and can be easily visualized.

Although the algorithm we provide for model fitting is reminiscent of *k*-means clustering, vector field *k*-means is an entirely novel formulation for trajectory clustering. It leverages techniques developed in the Computer Graphics and Visualization communities that have not been considered as tools for clustering in other communities. The most critical consequence of this is the ability to use partial data, such as the example in Fig. 2. Throughout the paper, the trajectories are colormapped to indicate orientation from blue (start) to orange (end). The vector field plots are colormapped such that darker arrows denote relatively larger magnitudes.

Vector field *k*-means is easy to state (Section 3) and implement (Section 4); it is efficient and scales well (Section 5). For implementations that are available, we provide direct experimental comparisons (Section 5.5). Lastly, when implementations are not available for direct comparison, we provide extensive discussion (Sections 2 and 6).

2. Related Work

Due to the growing rate at which mobility data is being collected, *computational movement analysis* is a very active research field, combining techniques and expertise from many fields, including GIS, information visualization, computational geometry, databases, and data mining [GLW12]. In this work, we focus on just one of the problems in movement analysis, that of extracting arbitrary movement patterns from trajectory data. In other words, given a large number of trajectories of moving objects, e.g. animals, people, or vehicles, we want to quickly identify underlying patterns that exist and that shed light on the global movement trends of the moving objects. Our approach for identifying these patterns is to perform trajectory clustering. As Kisilevich et al. [KMNR10] provide a thorough examination of many trajectory clustering techniques, we briefly review the most relevant methods here.

Rinzivillo et al. [RPN*08] designed a density-based progressive clustering technique that can utilize different distance functions at each step of their clustering. This allows analysis of objects with heterogeneous properties to be handled differently during the cluster refinement stages. Lee et al. [LHW07] also use density-based clustering, but believe that clustering whole trajectories may miss common sub-trajectories. They partition the trajectories into line segments based on a simplification algorithm and cluster these segments using the notions of neighborhood and density. The techniques of both Rinzivillo and Lee rely on the definition of

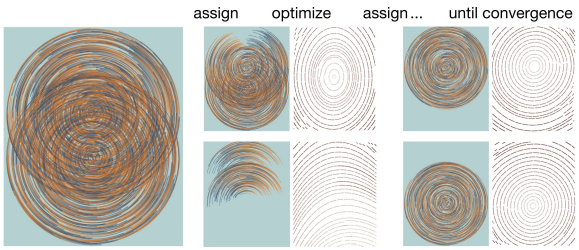


Figure 2: *An illustration of vector field k-means as it partitions 2000 synthetic trajectories into two clusters. The algorithm alternates between fitting the best possible vector fields from the current assignment ("optimize") and matching trajectories to the vector field which fits them best ("assign"). Although no trajectories form a complete circle, vector field k-means still recovers the two separate circular patterns.*

a distance measure between trajectories. This is known to be a difficult problem, in the sense that no proposed distance measure captures well all the attributes of trajectories [GLW12]. For example, both of these methods only use the geometry of the trajectory and therefore they cannot encode speed information, which might be relevant in cases like storm track analysis. Pelekis et al. [PKK*09] exploit local similarities of subtrajectories too, but they also study the effect of uncertainty (in measurement) in the original trajectory data.

Like Rinzivillo et al., our overall approach falls within the broader category of *visual and exploratory movement analysis*, which exploits humans' ability to visually detect patterns, and then steer the visualization and analysis to those regions of greatest interest. The research of Andrienko and Andrienko [AAW07, RPN*08, AAR*09] has focused on human-in-the-loop analysis systems, but has also included more general aggregation and visualization of movement data [AA08], and more recently the identification of important locations and events by analyzing movement data [AAH*11, AA11] and the visualization of trajectory attributes [TSAA12]. All these works could benefit from more efficient clustering methods since they rely on clustering for display purposes.

An important problem related to trajectory clustering is how to ultimately visualize trajectory data. Traditionally flow maps [PXY*05, VBS11, AA11] have been used to convey the amount of people and goods that moved between locations but without necessarily reporting the exact routes that were taken. More recently, there have been several compelling techniques based on density maps [WvdWvW09, SWvdW*11], kernel density estimation [DLH11], and 3D based visualizations [TSAA12]. Vector fields have been widely used in scientific visualization and even by some researchers doing trajectory clustering analysis to show speed and direction of animal movements [BPAK04] and wind [CRG*07]. In these cases, they have only been used to visualize the results, rather than as an integral part of the underlying clustering technique.

The related problem of deriving vector fields from trajectory datasets has been studied in different contexts. Jänicke et al. [JWC*11] defined a measure to evaluate vector field visualizations based on how well a computer vision algorithm could derive a vector field from an image containing visual representations (e.g., streamlines) of the vector field. This problem was independently investigated in the image processing community by Nascimento et al. [NFM09]. Although the model proposed by Nascimento et al. has some similarities with ours, we point out that it is not a clustering algorithm, but a way to model movement as vector fields. Furthermore, their model has a set of complex parameters to be estimated which brings higher computational costs, and more importantly increases the complexity for the human analyst. Thus their method is not suitable for our purposes. We propose a different (and simpler) modeling approach which has a smaller number of variables and outputs not only the vector fields (that in our case do represent distinct mobility patterns) but also a meaningful clustering of the input dataset, which makes our algorithm suitable for visual data analysis.

3. Vector Field K-Means Overview

Trajectories are modeled as paths of the form $\alpha : [t_0, t_1] \to \mathbb{R}^2$. We assume we are given a set of n trajectories $\mathscr{T} = \{\alpha_1, ..., \alpha_n\}$. Each trajectory is given as a sample, i.e., for each $i = 1, ..., n$, we are given a sequence of space-time points $\widehat{\alpha}_i = \{(\alpha_i(t_1^i), t_1^i), (\alpha_i(t_2^i), t_2^i), \ldots, (\alpha_i(t_{p_i}^i), t_{p_i}^i)\}$. We approximate each trajectory α_i with piecewise linear curves (constant velocity between two consecutive samples). This results in a polygonal line representation for each trajectory. For each α_i we denote the interval $[t_1^i, t_{p_i}^i]$ by I_i and by $|I_i|$ the time span for α_i, i.e., $|I_i| = t_{p_i}^i - t_1^i$. We call each portion of a trajectory between two samples a segment of the trajectory α_i. For each segment $s_j = [\alpha_i(t_j), \alpha_i(t_{j+1})]$ of α_i we define $\omega_{s_j} = \frac{t_{j+1} - t_j}{T}$, where $T = \sum_{\alpha_i \in \mathscr{T}} |I_i|$ is the total time span in the dataset.

In this paper, we consider vector fields as functions defined over a domain $\Omega \subset \mathbb{R}^2$ with values in $\mathbb{R}^2$. We discretize Ω as a regular triangle grid G with resolution R (R^2 vertices) and assume linear interpolation within each face of the grid for the reconstruction of the vector field. We assume all trajectories are contained in this grid and are tessellated so that each trajectory is comprised of segments that do not cross the boundaries of the domain triangles as in Fig. 3.

For the optimization problems described in this paper, the components of the vector field can be treated separately. Therefore, the presented formulas are written as if X were a scalar field. For each segment s_j of a trajectory α_i we denote by C_{s_j} the $2 \times R^2$ matrix that contains in the first and second rows respectively the barycentric coordinates of the first and second vertices of segment s_j (with 0 entries everywhere else); Fig. 3 illustrates the setting. We also define the vector b_{s_j} as the vector whose entries equal the value of the velocity vector of the segment s_j, i.e., $b_{s_j} = \frac{\alpha_i(t_{j+1}) - \alpha_i(t_j)}{|I_s|}$.

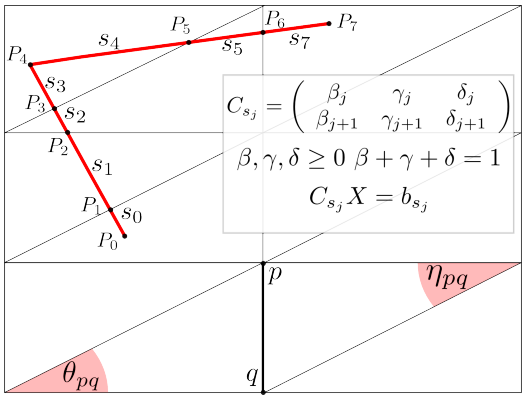


Figure 3: *Illustration of the trajectory tessellation and Laplacian matrix computation. The trajectories are tessellated so each segment is contained on a face of the grid. Each segment s_j determines a constraint in the form of a matrix C_{s_j}. The Laplacian matrix enforces our smoothness penalty.*

As briefly described before, our approach consists of capturing movement patterns by defining a vector field for which the trajectories are approximately integral lines, i.e., we attempt to separate the trajectories into a small number of clusters according to the best vector field that approximates them. Our assumption is that for every set of trajectories $\mathscr{T}$, there exists a set of *smooth vector fields* $X_j \in F, |F| = k$ that explains most of the mobility in the data, in the sense that each trajectory would be approximately tangent to one of the X_j. More formally, we define this as the problem of finding the vector fields X_j and the cluster assignment function $\Phi : \mathscr{T} \to \{1,\ldots,k\}$ such that for each $j = 1,\ldots,k$

$$\begin{cases} \Delta X_j = 0 \\ \alpha_i'(t) = X_j(\alpha_i(t)), \forall \alpha_i \in \Phi^{-1}(j) \text{ and } \forall t \in I_i \end{cases}$$

where Δ denotes the vector Laplace operator. The first equation directs the solution towards smooth vector fields; the second equation ensures that vector fields represent the trajectories well. It is clear that with this formulation the problem may have no solution. We propose instead to solve it in the least squares sense, by minimizing the following energy

$$E(X_1,\ldots,X_k,\Phi) = \sum_{j=1}^{k} \lambda_L \left|\left|\Delta X_j\right|\right|^2 + \frac{(1-\lambda_L)}{T} \sum_{\alpha_i \in \Phi^{-1}(j)} \int_{t_1^i}^{t_{p_i}^i} \left|\left|X_j(\alpha_i(t)) - \alpha_i'(t)\right|\right|^2 dt, \quad (1)$$

where T is the normalization factor (defined previously) and $\lambda_L > 0$ plays the role of a weighting factor: for small values of λ_L less weight is given to the smoothness of vector fields and therefore we get vector fields that match the given trajectories but are less smooth. For large values of λ_L we are giving relatively high priority to the smoothness of the vector field.

Algorithm 1 Vector Field K-Means Outline

Input: k: # of clusters, $\mathscr{T} = \{\alpha_1,\ldots,\alpha_n\}$: Array of curves
Output: $V = \{X_1,\ldots,X_k\}$, $\Phi : \mathscr{T} \to \{1,\ldots,k\}$

```
Φ ← Initialize(𝒯,k)
repeat
  for i = 1 to k do
    X_i ← fitVectorField(Φ^{-1}(i))
  end for
  for i = 1 to n do
    j_0 ← argmin_{j∈{1,2,...,k}} E''(X_j, α_i)
    Φ(α_i) ← j_0
  end for
until converge
```

4. Vector Field K-Means Fitting Process

As described previously, our model has two main elements: the vector fields and the assignment of trajectories to clusters. Therefore the model fitting problem consists of defining the vector fields $(X_1,...,X_k)$ and assigning each trajectory to a vector field (Φ), guided by Eq. 1. We propose a 2-level alternate optimization algorithm as the vector field k-means fitting process. It consists basically of the following steps:

(*) Given a candidate assignment of trajectories Φ, for each set $\Phi^{-1}(i), i = 1,\ldots,k$, we find the best-fitting vector fields *over those particular trajectories*, and

(**)Given a set of vector fields $\tilde{V} = \{X_1,\ldots,X_k\}$, we compute the best assignment function *for those particular vector fields* by picking, for each trajectory, the vector field that best approximates it.

Algorithm 1 contains the outline of vector field k-means. In this pseudo-code, the step $(*)$ corresponds to the *fitVectorField* routine. As we show below, we can formulate this step as a sparse linear system (in the general form of [SCOIT05]) whose solution can be computed essentially in linear time, and which gives us the smoothest, best-fitting vector field for a set of trajectories. The step $(**)$ corresponds to finding the vector field with smallest error with respect to a given trajectory. We further describe these components below.

4.1. Fitting Vector Fields

The *fitVectorField* routine is the central step of vector field k-means. It consists of an optimization problem with two types of constraints: value and smoothness, defined below. As depicted in Algorithm 1, in this step we are given as input a *fixed subset* $\mathscr{T}'$ of $\mathscr{T}$ and we want to minimize

$$E'(X,\mathscr{T}') = \lambda_L \left|\left|\Delta X\right|\right|^2 + \frac{(1-\lambda_L)}{T} \sum_{\alpha_i \in \mathscr{T}'} \int_{t_1^i}^{t_{p_i}^i} \left|\left|X(\alpha_i(t)) - \alpha_i'(t)\right|\right|^2 dt \quad (2)$$

We now use the grid G to write E' in matrix form. First, we use the well-known Laplacian matrix cotangent weights [VL08], denoted by L, to get a finite representation of the Laplace-Beltrami operator. In general, if we let p,q be a pair of adjacent vertices of the grid (see Fig. 3), then the entries of the cotangent Laplacian matrix L are given by

$$\bar{L}_{pq} = \frac{1}{2}(\cot\theta_{pq} + \cot\eta_{pq}),$$
$$\bar{L}_{p} = -\sum_{t \in Neighborhood(p)} \Delta_{pt}.$$

This formula allows us to compute the Laplacian of scalar functions defined over the grid by simply representing the values of the function on the vertices of the grid as a vector and multiplying the vector field by the Laplacian matrix. More concretely, by using the Laplacian matrix of G we can represent the first term of Eq. 2 in matrix form as $||\sqrt{\lambda_L}LX||^2$, where X is the finite-dimensional representation of the vector field. Before representing the second term of E' in matrix form we define for each segment s_j the matrix $\widetilde{C}_{s_j} = \sqrt{(1-\lambda_L)\omega_{s_j}}\Lambda C_{s_j}$ and the vector $\widetilde{b}_{s_j} = \sqrt{(1-\lambda_L)\omega_{s_j}}\Lambda b_{s_j}$, where Λ is a 2 by 2 matrix given by

$$\Lambda = \frac{1}{2\sqrt{6}}\begin{pmatrix} \sqrt{3}+1 & \sqrt{3}-1 \\ \sqrt{3}-1 & \sqrt{3}+1 \end{pmatrix}.$$

We denote by $\widetilde{C}$ the matrix that consists of stacking C_{s_j} for all segments s_j and denote by $\widetilde{b}_{s_j}$ the vector obtained by stacking all the $\widetilde{b}_{s_j}$ again for all s_j. A simple calculation shows that the second summand in Eq. 2 is given by $||\widetilde{C}X - \widetilde{b}||^2$. Therefore

$$E'(X,\mathscr{T}') = ||LX||^2 + \left|\left|\widetilde{C}X - \widetilde{b}\right|\right|^2$$

The minimization of this energy is a least-squares problem and can be solved by solving the corresponding system of normal equations: $(L^TL + \widetilde{C}^T\widetilde{C})X = \widetilde{C}^T\widetilde{b}$.

4.2. Assigning trajectories to vector fields

In the second phase of Algorithm 1, we assume we have the vector fields $\mathscr{V} = \{X_1,\ldots,X_k\}$ fixed. The goal is to build the next function Φ that assigns each trajectory to one of the k cluster centers, i.e. the vector fields $X_1,\ldots,X_k$. The assignment algorithm is trivial: for each trajectory α_i, we simply evaluate $E''(X_j,\alpha_i) = \int_{t_1^i}^{t_{p_i}^i} \left|\left|X_j(\alpha_i(t)) - \alpha_i'(t)\right|\right|^2 dt$, for each vector field X_j, and define the new assignment to be the minimizer for all the k possible choices.

4.3. Algorithm Initialization

We implemented a simple method to choose the initial vector fields and trajectory partitions that was effective in our experiments. The main idea is to try to have as diverse initial clusters as possible. The algorithm takes as inputs an array of curves and a number k of clusters to be created, and starts by choosing a trajectory α at random to be part of the first cluster. It uses the *fitVectorField* routine previously described to fit a single trajectory to the first vector field. The algorithm proceeds by fitting to the i-th vector field the trajectory that has the worst error among all previously fit vector fields. After computing k vector fields, we compute the assignment Φ by picking the best vector fields for each trajectory.

4.4. Computational Complexity and Convergence

As discussed in Section 4.2, the assignment step consists of a linear pass over the trajectory data, and for each of these, we need to find the vector field that minimizes the error. This can be implemented in $O(k|S(\mathscr{T})|)$, where $S(\mathscr{T})$ denotes the set of line segments that compose the trajectories in $\mathscr{T}$. For the fitting step, we have used a simple Unconstrained Conjugate Gradient algorithm as a linear system solver [NW99]. Therefore, the complexity of this step is given by $O(kN(R^2 + |S(\mathscr{T})|))$, where N denotes the maximum number of iterations of the Conjugate Gradient Method, R denotes the grid resolution corresponding to the multiplication by the Laplacian matrix, and $|S(\mathscr{T})|$ corresponds to the multiplication by the constraint matrix C. As we see in the experiments, good results can be obtained with relatively low values of R and hence the complexity is dominated by $kN|S(\mathscr{T})|$. The choice of the Conjugate Gradients solver was convenience; we could further optimize our implementation by using more sophisticated methods to solve systems of linear equations [NW99]. We note that each iteration of the main loop in Alg. 1 is guaranteed to decrease the energy E and thus no assignment can be repeated, from which we conclude that our algorithm converges in a finite number of steps.

5. Experiments and Evaluation

We now report the results of running vector field k-means. In our experiments, the algorithm was able to efficiently extract significant movement patterns across diverse datasets. We start with a small synthetic dataset and progressively increase the input sizes until we reach an example with over 370,000 very noisy trajectories. All running times (see Fig. 5) are from our prototype implementation: a single-threaded, single-process C++ application running on an Intel Core i7-960 desktop with 6GB of RAM. The total memory required by

Dataset	Trajectories	Resolution	k	Optimize	Assign
Synthetic	2000	3	2	1.730s	0.074s
Atlantic	1415	5	7	8.076s	0.335s
Beijing Wide	45563	5	4	110.99s	4.265s
Beijing Campus	12883	10	16	124.35s	2.195s
CDR	37435	4	4	201.24s	7.597s
CDR Large	372601	4	4	2497s	75.24s

Figure 5: *Experimental results: For each dataset, we report the number of trajectories, the grid resolution, the number of clusters (k), and the total running times (in seconds) for the vector field fitting (optimize) and trajectory assignments.*

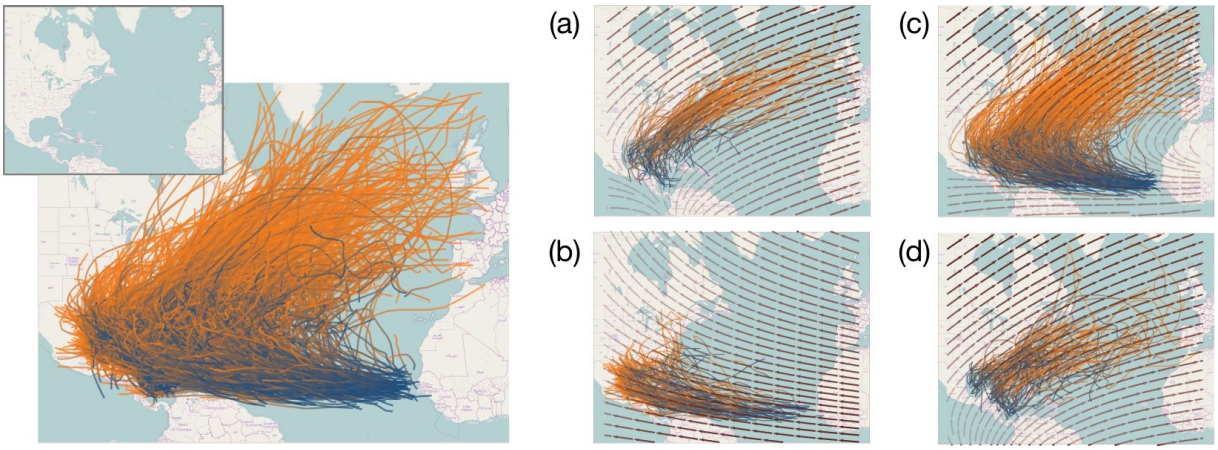


Figure 4: *Left: 1415 Atlantic tropical storms (from the HURDAT dataset) used as input. Right: four trajectory clusters and their corresponding vector fields showing relative speed. Clusters (b) and (c) contain the Cape Verde-type cyclones, and separate them according to whether they dissipate in North America (b) or turn back to the Atlantic Ocean (c). Clusters (a) and (d) show storms developing in the Gulf of Mexico. We observed faster-moving storms to be in cluster (a), and slower-moving ones in (d).*

our application remained under 1GB for all reported experiments. We ran the algorithm until convergence, although the optimization could be stopped earlier when the number of trajectories that change from one cluster to another is small.

5.1. Synthetic Data

In this dataset there exists two overlapping circulatory movement patterns. Each trajectory covers a partial, randomly selected section of the circle at a random distance from the center. We sampled 1000 trajectories from each overlapping pattern. As we show in Fig. 2, vector field *k*-means recovers the two overlapping patterns perfectly, which illustrates that our method does not create clusters by selecting representative trajectories at all: its vector fields fit *all* circular trajectories equally well. Our implementation converged in 30 iterations, in less than 2 seconds, using a 3x3 grid.

5.2. Atlantic Hurricanes

HURDAT is a hurricane tracking dataset maintained by the National Hurricane Center (NHC) [hur12]. The dataset contains trajectories of the 1415 Atlantic tropical storms between 1861 and 2011. It contains not only position and time information, but also sustained surface wind speeds and sea-level pressure. The data are recorded with a resolution of 6 hours.

In Fig. 4, vector field *k*-means separates what looks like a fairly uniform set of trajectories. One of the clusters neatly captures Cape Verde hurricanes which tend to make landfall in North America, while two other clusters show storms that originate in the Caribbean and Gulf of Mexico. Upon closer inspection, this separation of two similar looking clusters is due to the more chaotic trajectories of one of the clusters, which result in a generally lower-velocity vector field.

5.3. GeoLife GPS Trajectory Dataset

The GeoLife GPS dataset consists of 17,621 trajectories recorded by Microsoft Research at Beijing. The trajectories are GPS tracks of 178 users from April 2007 to October 2011 [ZXM10]. Although the dataset encompasses trajectories across the globe, we focus on two regions around Beijing. As the raw trajectories are unsegmented, some lasting for days, we split trajectories whenever the time between two samples is larger than 2.5 times the median time between samples. We then reduced the sampling rate by only keeping measurements at least 2 minutes apart.

Fig. 6 shows a first run of vector field *k*-means on the GeoLife dataset in which the algorithm was able to find general movement trends within the trajectories. Three of these are clear directional patterns heading west, north, and south. We speculate these to be mainly commuting patterns, since the remaining cluster (subfigure c) consists essentially of trajectories inside the city's road network. Although we believe that with the chosen grid resolution (5x5) vector field *k*-means cannot reliably resolve the patterns in that cluster (and hence the vector field is not very informative), the large density of trajectories around a relatively small area in the cluster suggested further analysis centered in that region.

Fig. 1 shows an exploration we performed on that narrower region of Beijing. In this case we used a grid of resolution 15. By looking at the darkness of the arrows in Fig. 1 we see that regions of different speeds are found. For example, traffic over the streets (Fig. 1 (a) and (b)) is much faster than the speed of people going to and from the speculative lunch spot location (Fig. 1 (c) and (d)), which suggests that people walk to that destination.

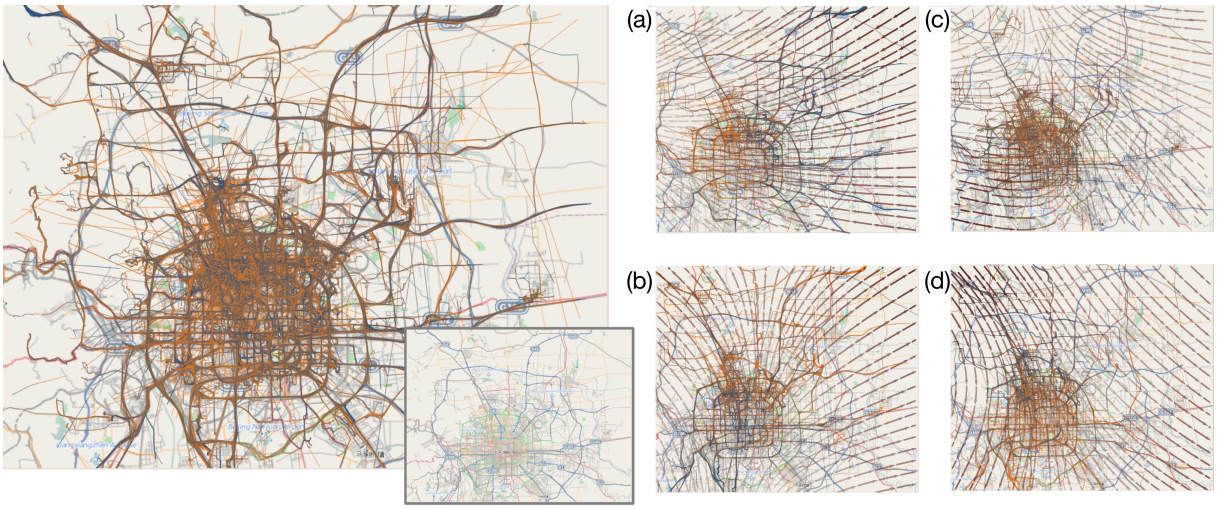


Figure 6: *Large-scale movement patterns around Beijing, from the GeoLife Trajectories dataset. Clusters (a), (b) and (d) appear to depict travel in and out of the city through the surrounding highways. Cluster (c) has much slower speeds and its trajectories are tightly packed around a small region. Upon inspection, we found this to area to contain the Microsoft Research Asia campus.*

5.4. Call Detail Record Dataset

We collected anonymized Call Detail Records (CDR) from the cellular network of a large U.S. communications service provider. For each phone call, CDRs provide us with the sequence of cellular antenna locations, known as handoffs, used during that call. We collected over 370,000 calls during one week in 2011 for approximately 300 antenna located near a suburban city with 20,000 residents. Our goal was to capture handoffs related to vehicular traffic.Given the sensitivity of CDRs, we took several steps to ensure privacy. The data was collected and anonymized by a third party not involved in the data analysis. Unlike other studies [BCH*11] our dataset cannot associate multiple calls made by the same individuals: each call is independent. As a result, all of our input trajectories (i.e.handoff patterns), are only partial trajectories for any individual. As we show in Figs. 9 and 10, although the handoffs are quite noisy, we can still recover movement patterns clearly related to the highway traffic around the city.

5.5. Comparison

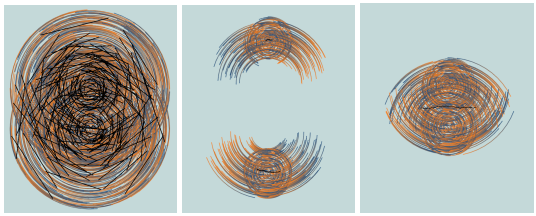
We compare our algorithm with TraClus, by Lee et al. [LHW07], a density-based algorithm that is one of the main references in trajectory clustering. Roughly speaking, the algorithm consists of two steps: trajectory simplification into line segments and segment clustering. The algorithm also has two main parameters: a distance threshold ε used to define neighborhoods for each segment, and a density lower bound *MinLns* that is used to find neighborhoods that define clusters. Unlike vector field *k*-means, TraClus does not incorporate the time information, thus trajectory speed information is lost.

We used the C++ implementation of their algorithm that is available on the author's webpage (http://dm.kaist.ac.kr/jaegil/#Publications). In the following experiments, we use the heuristic proposed in their paper (and also part of the author's implementation) to select the parameter values. Whenever this heuristic fails to provide the parameter estimates, we adjusted the parameters by manually searching over a range of values. We reiterate that as vector field *k*-means and TraClus mine different patterns, it is difficult to say that one method is always better than the other, but we do investigate what kind of patterns TraClus is not able to find that vector field *k*-means is, and vice versa.

We first present the performance of TraClus on the synthetic discussed in Sec. 5.1. The results are shown in Fig. 7. Note that because TraClus utilizes local features while clustering, it is unable to capture the overall global structure of this dataset. More specifically, when the neighborhood size ε is very small, TraClus considers each segment as its own cluster, while making ε larger groups the segments of the two centers of the circular patterns together, as shown in Fig. 7.

Our results of running TraClus on the HURDAT dataset, shown in Fig. 8, are very similar to the ones obtained by the authors [LHW07] though not exact because our input contained more trajectories. In this case, TraClus detects nine clusters in four seconds. The representative trajectory for each of these clusters is highlighted in black, but are not as informative as the vector fields of Fig. 4. In addition, their partitioning step precludes TraClus from generating clusters of Atlantic crossing storms that then proceed up the U.S. east coast. That information has been lost by their approach.

We also performed experiments using the TraClus algorithm and the GeoLife dataset (used in Fig. 1), In this case,



(a) (b) (c)

Figure 7: *TraClus experiments using the synthetic dataset. In (a) 268 clusters are found with $\varepsilon = 0.03$ and $MinLns = 2$. In (b), with $\varepsilon = 0.23$ and $MinLns = 140$ TraClus detects two clusters (drawn here separated for clarity), but clearly merges portions of the two circular patterns. Slight variations on the parameters ($\varepsilon = 0.25$ and $MinLns = 160$) causes TraClus to merge the two cluster into one as seen in (c). These results were obtained in 0.8, 1.6, and 6.5 seconds respectively.*

TraClus failed to produce any meaningful clusters. Using the parameters reported by the TraClus heuristics ($\varepsilon = 20$ and $MinLns = 12454$), the algorithm finds no clusters, after 54 seconds of computation. Changing the parameters to $\varepsilon = 20$ and $MinLns = 1000$, the algorithm reports a single cluster which does not represent any meaningful pattern, after a little more than 3.6 hours of computation.

6. Discussion

In this section, we discuss issues related to parameter selection, advantages and limitations of vector field *k*-means, as well as possible extensions of our algorithm.

6.1. Parameter selection

Although we can select both grid resolution R and the weight given to the Laplacian regularization λ_L, the two parameters are not independent. The eigenvectors of the Laplacian are naturally interpreted as equivalent to the fundamental frequencies on the mesh, exactly like sines and cosines are the fundamental frequencies on a circle [VL08]. The corresponding eigenvalues are the squares of the frequencies themselves, which means that as we increase λ_L, we give larger weights to the eigenvectors corresponding to high-frequency signals, and the system tends towards lower-frequency results. A similar effect is achieved by reducing R, which directly band-limits the signal on the vector field. Using this dependency between these parameters, we can virtually eliminate one of them. As a result, we fix λ_L to be 0.05 and interactively select R in our experiments. The resolution parameter R thus controls the amount of detail in the derived vector fields: by increasing R more complex behavior can be modeled and smaller features have more influence in the clustering. Decreasing R enables the algorithm to filter smaller, possibly insignificant, features and capture global trends. As with most parameters, a clear trade-off exists.

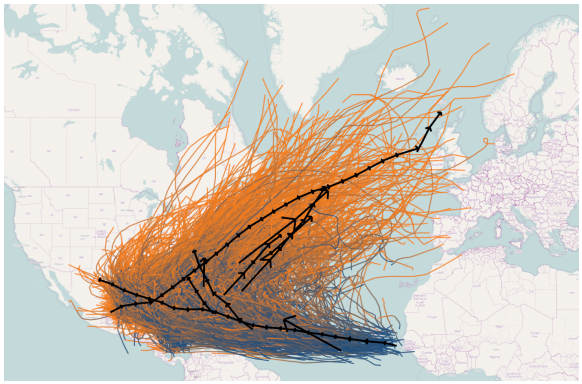


Figure 8: *TraClus computed nine clusters for the HURDAT dataset. Cluster representative trajectories are in black. Notice that because TraClus starts by subdividing trajectories, no cluster from TraClus captures the pattern of trajectories found by vector field k-means in Figure 4 (c).*

Picking an appropriate number of clusters remains an open problem even in the case of traditional *k*-means, and we offer no substantive contributions on that matter. Many proposed methods try to attack this problem (see [FW12] and references therein), however no definitive algorithm solves this problem optimally for all applications in general settings. Still, we stress that as far as performance is concerned, vector field *k*-means compares quite favorably to results reported in the literature, and thus it is much easier to keep a human analyst in the loop and make cluster count an interactive procedure with vector field *k*-means than with previous methods.

6.2. Advantages

Our model aims at a balance between richness and expressivity of features, and simplicity of implementation and analysis. We believe this is a significant advantage over the current methods for trajectory clustering. As mentioned earlier, by representing the cluster centers as vector fields and using those as a means to define similarity between trajectories, we can eliminate expensive computations of metrics for trajectories and the computations of the centroid trajectory as well [GLW12]. Another advantage is our ability to capture global patterns that are not apparent at the local level in highly noisy trajectories or even partial trajectories.

Vector field *k*-means is also highly parallelizable. Our prototype does not take advantage of this and includes no significant optimizations, but it is obvious that separate components of vector fields can be computed in parallel, and that many of the intermediate matrices in the linear solvers can be reused from one iteration to the next. We expect these to further increase the performance, and allow vector field *k*-means to handle even larger datasets.

Figure 9: *The anonymized call detail records for over 370,000 cell phone calls produced noisy trajectories around a suburban city. Two of the four clusters computed are shown.*

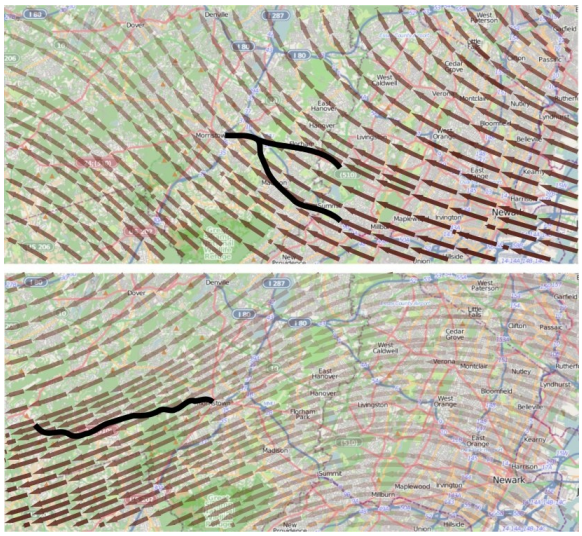


Figure 10: *Despite the noisy trajectories, we recovered clear movement patterns related to highway (bold black lines) traffic. The two vector fields correspond to the clusters in Fig. 9.*

6.3. Limitations

The choice of the initial clusters impact the results achieved by vector field k-means, as it converges to a local minimum and we currently do not offer any guarantees that our result is close to the global optimum. Many techniques have been proposed to choose good initial centers for the clusters in the regular k-means case, one of the most important being the k-means++ approach by Arthur et al. [AV07], which proves a $\log k$ approximation factor. However it is not yet obvious how to extend this technique to vector field k-means results.

Although vector field k-means is able to naturally encode some attributes such as direction and speed, which are not well handled by previous methods, the notion of spatial distance is not directly represented, which may result in clusters with different behaviors in different spatial regions.

The vector fields derived by our method are steady, and in general, trajectories with self-intersections can lead to artifacts in steady vector fields near these areas. However, our results show that vector field k-means is able to mine interesting patterns even in the presence of self-intersections.

6.4. Future Work

For clarity and simplicity, we described the vector field k-means algorithm in Section 3 in 2D, but one could extend it to higher dimensions. One would need to involve the region of interest in a simplicial grid of dimension d, in which we assume linear interpolation inside each simplex of the grid. This extension could be used to derive time dependent vector fields, by creating a 3D grid and setting the constraints on the interior of the tetrahedral decomposition. This would allow for better treatment of self-intersecting trajectories. Most of the steps would be straightforward to carry through; although we know of no good 3D equivalent to the cotangent weight of the 2D Laplacian. It is also unknown if our algorithm's performance characteristics will remain in a higher-dimensional extension.

Another interesting research direction is to apply vector field k-means using user input. Conceptually, a user could visually define the vector field using vector field design techniques [CKW*12] or even some sample trajectories. The algorithm would then retrieve all trajectories in the database that follow those patterns and whose errors are small enough.

7. Conclusions

We have introduced a novel trajectory clustering technique that brings together ideas from visualization, data clustering, and scalar field design. Vector field k-means can find global patterns, handle partial trajectories, scale to large datasets, and is simple to implement. The algorithm opens many possibilities for modeling and user interfaces design, notably for querying and exploring trajectory datasets.

Acknowledgments

We thank the OpenStreetMap project [HW08] for providing the maps used in many figures in this paper. This work was supported in part by the National Science Foundation (CNS-1229185, CNS-1153503, IIS-0905385, IIS-1142013, and AGS 0835821), the Department of Energy.

References

[AA08] ANDRIENKO G., ANDRIENKO N.: Spatio-temporal aggregation for visual analysis of movements. In *IEEE Symposium on Visual Analytics Science and Technology* (2008), pp. 51–58. 3

[AA11] ANDRIENKO N., ANDRIENKO G.: Spatial generalization and aggregation of massive movement data. *IEEE Trans. Visualization and Computer Graphics 17*, 2 (2011), 205–219. 3

[AAH*11] ANDRIENKO G., ANDRIENKO N., HURTER C., RINZIVILLO S., WROBEL S.: From movement tracks through events to places: Extracting and characterizing significant places from mobility data. In *IEEE Symposium on Visual Analytics Science and Technology* (2011), pp. 159–168. 3

[AAR*09] ANDRIENKO G., ANDRIENKO N., RINZIVILLO S., NANNI M., PEDRESCHI D., GIANNOTTI F.: Interactive visual clustering of large collections of trajectories. In *IEEE Symposium on Visual Analytics Science and Technology* (2009), pp. 3–10. 3

[AAW07] ANDRIENKO G., ANDRIENKO N., WROBEL S.: Visual analytics tools for analysis of movement data. *SIGKDD Explorations 9*, 2 (2007), 38–46. 3

[AV07] ARTHUR D., VASSILVITSKII S.: k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms* (2007), Society for Industrial and Applied Mathematics, pp. 1027–1035. 9

[BCH*11] BECKER R. A., CACERES R., HANSON K., LOH J. M., URBANEK S., VARSHAVSKY A., VOLINSKY C.: Route classification using cellular handoff patterns. In *13th International Conference on Ubiquitous Computing* (2011), pp. 123–132. 1, 7

[BDE09] BAYIR M., DEMIRBAS M., EAGLE N.: Discovering spatiotemporal mobility profiles of cellphone users. In *IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks & Workshops* (2009), pp. 1–9. 1

[BPAK04] BRILLINGER D., PREISLER H., AGER A., KIE J.: An exploratory data analysis (EDA) of the paths of moving animals. *Statistical Planning and Inference 122*, 2 (2004), 43–63. 1, 3

[CKW*12] CHEN G., KWATRA V., WEI L., HANSEN C., ZHANG E.: Design of 2d time-varying vector fields. *IEEE Trans. Visualization and Computer Graphics 18*, 10 (2012), 1717–1730. 9

[CRG*07] CAMARGO S. J., ROBERTSON A. W., GAFFNEY S. J., SMYTH P., GHIL M.: Cluster analysis of typhoon tracks. *Climate 20*, 14 (2007), 3635–3676. 1, 3

[DLH11] DAAE LAMPE O., HAUSER H.: Interactive visualization of streaming data with kernel density estimation. In *IEEE Pacific Visualization Symposium* (2011), pp. 171–178. 3

[Els03] ELSNER J.: Tracking hurricanes. *Bulletin of the American Meteorological Society 84*, 3 (2003), 353–356. 1

[FLF*11] FERREIRA N., LINS L., FINK D., KELLING S., WOOD C., FREIRE J., SILVA C.: BirdVis: Visualizing and understanding bird populations. *IEEE Transactions on Visualization and Computer Graphics 17*, 12 (2011), 2374–2383. 1

[FW12] FANG Y., WANG J.: Selection of the number of clusters via the bootstrap method. *Comput. Stat. Data Anal. 56*, 3 (Mar. 2012), 468–477. 8

[GJL*09] GRUNDY E., JONES M. W., LARAMEE R. S., WILSON R. P., SHEPARD E. L.: Visualisation of sensor data from animal movement. In *Computer Graphics Forum* (2009), vol. 28, Wiley Online Library, pp. 815–822. 1

[GLW12] GUDMUNDSSON J., LAUBE P., WOLLE T.: Computational movement analysis. *Springer Handbook of Geographic Information* (2012), 725–741. 1, 2, 3, 8

[hur12] HURDAT: The national hurricane center's north atlantic hurricane database, Feb 2012. URL: http://www.aoml.noaa.gov/hrd/data_sub/re_anal.html. 6

[HW08] HAKLAY M., WEBER P.: Openstreetmap: User-generated street maps. *Pervasive Computing, IEEE 7*, 4 (2008), 12–18. 9

[JWC*11] JÄNICKE H., WEIDNER T., CHUNG D., LARAMEE R. S., TOWNSEND P., CHEN M.: Visual reconstructability as a quality metric for flow visualization. In *Computer Graphics Forum* (2011), vol. 30, Wiley Online Library, pp. 781–790. 3

[KMNR10] KISILEVICH S., MANSMANN F., NANNI M., RINZIVILLO S.: Spatio-temporal clustering. *Data Mining and Knowledge Discovery Handbook* (2010), 855–874. 2

[LHW07] LEE J., HAN J., WHANG K.: Trajectory clustering: a partition-and-group framework. In *ACM SIGMOD International Conference on Management of Data* (2007), pp. 593–604. 1, 2, 7

[NFM09] NASCIMENTO J. C., FIGUEIREDO M. A. T., MARQUES J. S.: Trajectory analysis in natural images using mixtures of vector fields. In *IEEE International Conference on Image Processing* (2009), pp. 4353–4356. 3

[NW99] NOCEDAL J., WRIGHT S.: *Numerical optimization*. Springer Verlag, 1999. 5

[PKK*09] PELEKIS N., KOPANAKIS I., KOTSIFAKOS E., FRENTZOS E., THEODORIDIS Y.: Clustering trajectories of moving objects in an uncertain world. In *Ninth IEEE International Conference on Data Mining* (2009), pp. 417–427. 3

[PXQ*11] PU J., XU P., QU H., CUI W., LIU S., NI L.: Visual analysis of people's mobility pattern from mobile phone data. In *Proceedings of Visual Information Communication-International Symposium* (2011), pp. 13:1–13:10. 1

[PXY*05] PHAN D., XIAO L., YEH R., HANRAHAN P., WINOGRAD T.: Flow map layout. In *IEEE Symposium on Information Visualization* (2005), pp. 219–224. 3

[RPN*08] RINZIVILLO S., PEDRESCHI D., NANNI M., GIANNOTTI F., ANDRIENKO N., ANDRIENKO G.: Visually driven analysis of movement data by progressive clustering. *Information Visualization 7*, 3-4 (2008), 225. 1, 2, 3

[SCOIT05] SORKINE O., COHEN-OR D., IRONY D., TOLEDO S.: Geometry-aware bases for shape approximation. *IEEE Trans. Visualization and Computer Graphics 11*, 2 (2005), 171–180. 4

[SWvdW*11] SCHEEPENS R., WILLEMS N., VAN DE WETERING H., ANDRIENKO G., ANDRIENKO N., VAN WIJK J. J.: Composite density maps for multivariate trajectories. *IEEE Trans. Visual. and Computer Graphics 17*, 12 (2011), 2518–2527. 3

[TSAA12] TOMINSKI C., SCHUMANN H., ANDRIENKO G., ANDRIENKO N.: Stacking-based visualization of trajectory attribute data. *IEEE Trans. Visualization and Computer Graphics 18*, 12 (2012), 2565–2574. 1, 3

[VBS11] VERBEEK K., BUCHIN K., SPECKMANN B.: Flow map layout via spiral trees. *IEEE Transactions on Visualization and Computer Graphics 17*, 12 (2011), 2536–2544. 3

[VL08] VALLET B., LÉVY B.: Spectral geometry processing with manifold harmonics. *Computer Graphics Forum (Proceedings Eurographics)* (2008). 5, 8

[WvdWvW09] WILLEMS N., VAN DE WETERING H., VAN WIJK J. J.: Visualization of vessel movements. *Computer Graphics Forum (Proceedings of EuroVis) 28*, 3 (2009), 959–966. 3

[WYCM11] WEI J., YU H., CHEN J., MA K.: Parallel clustering for visualizing large scientific line data. In *IEEE Symposium on Large Data Analysis and Visualization* (2011), pp. 47–55. 1

[ZXM10] ZHENG Y., XIE X., MA W.: Geolife: A collaborative social networking service among user, location and trajectory. *IEEE Data Engineering Bulletin 33*, 2 (2010), 32–40. 6