CENTRAL TRAJECTORIES[*]

Marc van Kreveld,[§] *Maarten Löffler,*[§] *and Frank Staals*[§]

ABSTRACT. An important task in trajectory analysis is partitioning a set of trajectories into clusters: groups of similar trajectories. The clusters are often summarized by a single representative trajectory and an associated size of each cluster. We study the problem of computing a suitable representative of such a set of similar trajectories. To this end we define a *central trajectory* $\mathcal{C}$, which consists of pieces of the input trajectories, switches from one entity to another only if they are within a small distance of each other, and such that at any time t, the point $\mathcal{C}(t)$ is as central as possible. We measure centrality in terms of the radius of the smallest disk centered at $\mathcal{C}(t)$ enclosing all entities at time t, and discuss how the techniques can be adapted to other measures of centrality. We first study the problem in $\mathbb{R}^1$, where we show that an optimal central trajectory $\mathcal{C}$ representing n trajectories, each consisting of τ edges, has complexity $\Theta(\tau n^2)$ and can be computed in $O(\tau n^2 \log n)$ time. We then consider trajectories in $\mathbb{R}^d$ with $d \geq 2$, and show that the complexity of $\mathcal{C}$ is at most $O(\tau n^{5/2})$ and can be computed in $O(\tau n^3)$ time.

1   Introduction

A *trajectory* is a model for the movement of an entity in the plane, or more generally in $\mathbb{R}^d$, for example a function that maps time to location. Trajectory data is obtained by tracking the movements of e.g. animals [9, 13, 21], hurricanes [28], traffic [25], or other moving entities [16] over time. Large amounts of such data have recently been collected in a variety of research fields. As a result, there is a great demand for tools and techniques to analyze trajectory data.

One important task in trajectory analysis is *clustering*: subdividing a large collection of trajectories into groups of "similar" ones. This problem has been studied extensively, and many different techniques are available [10, 19, 20, 24, 29]. Once a suitable clustering has been determined, the result needs to be stored or prepared for further processing. Storing the whole collection of trajectories in each cluster is often not feasible, because follow-up analysis tasks may be computationally expensive. Instead, we wish to represent each cluster by a signature: the number of trajectories in the cluster, together with a *representative* trajectory which should capture the defining features of all trajectories in the cluster.

Figure 1:   (a) Every trajectory has a peculiarity that is not representative for the set. (b) Taking, for example, the pointwise average of a set of trajectories may result in one that ignores context.

Representative trajectories are also useful for visualization purposes. Displaying large amounts of trajectories often leads to visual clutter. Instead, if we show only a small number of representative trajectories, this reduces the visual clutter, and allows for more effective data exploration. The original trajectories can still be shown if desired, using the details-on-demand principle in information visualization [27].

Representative trajectories.   When choosing a representative trajectory for a group of similar trajectories, the first obvious choice would be to pick one of the trajectories in the group. However, one can argue that no single element in a group may be a good representative, e.g. because each individual trajectory has some prominent feature that is not shared by the rest (see Fig. 1(a)), or no trajectory is sufficiently in the middle all the time. On the other hand, it is desirable to output a trajectory that does consist of *pieces* of input trajectories, because otherwise the representative trajectory may display behaviour that is not present in the input, e.g. because of contextual information that is not available to the algorithm (see Fig. 1(b)).

To determine what a good representative trajectory of a group of similar trajectories is, we identify two main categories: *time-dependent* and *time-independent* representatives. Trajectories are typically collected as a discrete sequence of time-stamped locations. By linearly interpolating the locations we obtain a continuous piecewise-linear curve as the image of the function. Depending on the application, we may be interested in the curve with attached time stamps (say, when studying a flock of animals that moved together) or in just the curve (say, when considering hikers that took the same route, but possibly at different times and speeds).

When time is not important, one can select a representative based directly on the geometry or topology of the set of curves [11, 22]. When time is important, we would like to have the property that at each time t our representative point $\mathcal{C}(t)$ is a good representative of the set of points $P(t)$. To this end, we may choose any static representative point of a point set, for which many examples are available: the Fermat-Weber point (which minimizes the sum of distances to the points in P), the center of mass (which minimizes the sum of squared distances), or the center of the smallest enclosing circle (which minimizes the distance to the farthest point in P).

Central trajectories.   In this work, we focus on time-dependent measures based on static concepts of centrality. We choose the distance to the farthest point, but discuss in Section 4 how our results can be adapted to other measures.

Ideally, we would output a trajectory $\mathcal{C}$ such that at any time t, $\mathcal{C}(t)$ is the point (entity) that is closest to its farthest entity. Unfortunately, when the entities move in $\mathbb{R}^d$ for $d > 1$, this may cause discontinuities. See Fig. 2 for a simple example. Such discontinuities are unavoidable: if we insist that the output trajectory consists of pieces of input trajectories *and* is continuous, then in general, there will be no opportunities to switch from one trajectory to another,

Figure 2: The point $\mathcal{C}(t)$ minimizing the distance to the furthest other point will jump (from r to q), thus its trajectory is discontinuous.

and we are effectively choosing one of the input trajectories again. At the same time, we do not want to output a trajectory with arbitrarily large discontinuities. An acceptable compromise is to allow discontinuities, or *jumps*, but only over small distances, controlled by a parameter ε. We note that this problem of discontinuities shows up for time-independent representatives for entities moving in $\mathbb{R}^d$, with $d \geq 3$, as well, because the traversed curves generally do not intersect.
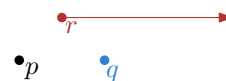
Related work.   Buchin et al. [11] consider the problem of computing a *median* trajectory for a set of trajectories without time information. Their method produces a trajectory that consists of pieces of the input. Recently, Ahn et al. [5] developed algorithms to compute a middle curve based on the discrete Fréchet distance. Note that they consider only the vertices of the curves. Agarwal et al. [1] consider trajectories with time information and compute a representative trajectory that follows the median (in $\mathbb{R}^1$) or a point of high *depth* (in $\mathbb{R}^2$) of the input entities. The resulting trajectory does not necessarily stay close to the input trajectories. They give exact and approximate algorithms. Durocher and Kirkpatrick [17, 18] observe that a trajectory minimizing the sum of distances to the other entities is *unstable*, in the sense that arbitrarily small movement of the entities may cause an arbitrarily large movement in the location of the representative entity. They proceed to consider alternative measures of centrality, and define the *Steiner center* and the *projection median*, which they prove are more stable. Basu et al. [7] extend this latter concept to higher dimensions. Agarwal et al. [2] study how to maintain the diameter and the width of a set of moving points, and Demaine et al. [14] present a data structure to maintain the smallest enclosing disk. Note that the center of the enclosing disk, or the mid point of the diametral pair may be an arbitrary point, rather than one of the input points. In both cases, the number of combinatorial changes during the motion is high ($\Omega(n^2)$). Therefore, Agarwal and Har-Peled [3] study approximating several measures of extent.

Problem description.   We are given a set $\mathcal{X}$ of n entities, each moving along a piecewise linear trajectory in $\mathbb{R}^d$ consisting of τ edges, and some distance threshold ε. We assume that all trajectories have their vertices at the same times, i.e. times $t_0, .., t_\tau$. Fig. 3(a) shows an example.   For an entity σ, let $\sigma(t)$ denote the position of σ at time t. With slight abuse of notation we write σ for both entity σ and its trajectory. At a given time t, we denote the distance from σ to the entity farthest away from σ by $D_\sigma(t) = \max_{\psi \in \mathcal{X}} \|\sigma(t)\psi(t)\|$, where $\|pq\|$ denotes the Euclidean distance between points p and q in $\mathbb{R}^d$. Fig. 3(b) illustrates the pairwise distances and resulting D functions for five example trajectories. For ease of exposition, we assume that the trajectories are in general position: that is, no three trajectories intersect in the same point, and no two pairs of entities are at distance ε from each other at the same time.

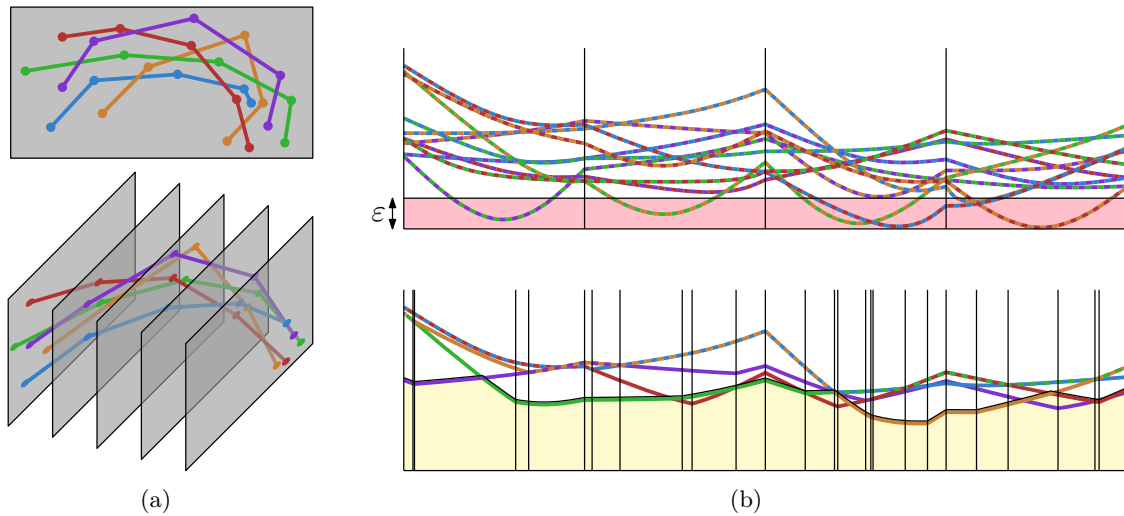(a)   (b)

Figure 3: (a) Two views of five moving entities and their trajectories. (b) On the top the pairwise distances between the entities as a function over time. On the bottom the functions D_σ, and in yellow the area representing $\mathcal{D}(\mathcal{C})$.
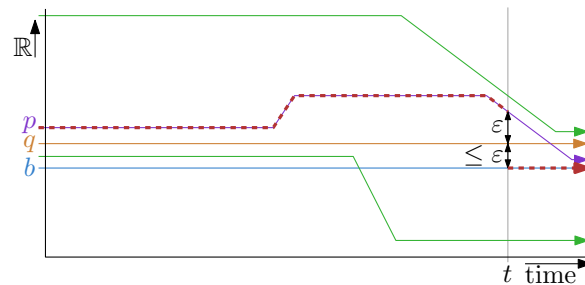

Figure 4: A set of entities moving in $\mathbb{R}^1$ that shows that $\mathcal{C}$ may make large jumps in an arbitrary short time interval $[t, t+\delta]$. Initially, the central trajectory $\mathcal{C}$ (red, dashed) follows entity p. When p (purple) is at distance ε from q (orange) at time t, $\mathcal{C}$ almost immediately jumps to b (blue).

A *trajectoid* is a function that maps time to the set of entities $\mathcal{X}$, with the restriction that at discontinuities the distance between the entities involved is at most ε. Intuitively, a trajectoid corresponds to a concatenation of pieces of the input trajectories in such a way that two consecutive pieces match up in time, and the end point of the former piece is within distance ε from the start point of the latter piece. In Fig. 3(b), a trajectoid may switch between a pair of entities when their pairwise distance function lies in the bottom strip of height ε. More formally, for a trajectoid $\mathcal{T}$ we have that

- at any time t, $\mathcal{T}(t) = \sigma$ for some $\sigma \in \mathcal{X}$, and
- at every time t where $\mathcal{T}$ has a discontinuity, that is, $\mathcal{T}$ *jumps* from entity σ to entity ψ, we have that $\|\sigma(t)\psi(t)\| \leq \varepsilon$.

Note that this definition still allows for a series of jumps within an arbitrarily short time

interval $[t, t + \delta]$, essentially simulating a jump over distances larger than ε. See Fig. 4. To make the formulation cleaner, we slightly weaken the second condition and allow a trajectoid to have discontinuities with a distance larger than ε, provided that such a large jump can be realized by a sequence of small jumps, each of distance at most ε. More formally, two entities σ and ψ are ε-*connected* at time t if there is a sequence $\sigma = \sigma_0, .., \sigma_k = \psi$ of entities such that for all i, subsequent entities σ_i and σ_{i+1} are within distance ε of each other at time t. We can then replace the second condition by

- at every time t where $\mathcal{T}$ has a discontinuity, that is, $\mathcal{T}$ jumps from entity σ to entity ψ, the entities σ and ψ are ε-connected.

When it is clear from the context, we write $\mathcal{T}(t)$ instead of $\mathcal{T}(t)(t)$ to mean the location of entity $\mathcal{T}(t)$ at time t. We now wish to compute a trajectoid $\mathcal{C}$ that minimizes the function

$$\mathcal{D}(\mathcal{T}) = \int_{t_0}^{t_\tau} D_{\mathcal{T}}(t)\, \mathrm{d}t.$$

So, at any time t, all entities lie in a disk of radius $D_{\mathcal{C}}(t)$ centered at $\mathcal{C}(t)$.

Outline and results. We first study computing a central trajectory minimizing $\mathcal{D}$ for entities that move in $\mathbb{R}^1$. In Section 2 we show that the worst-case complexity of a central trajectory in $\mathbb{R}^1$ is $\Theta(\tau n^2)$, and that we can compute one in $O(\tau n^2 \log n)$ time. We then extend our approach to entities moving in $\mathbb{R}^d$, for any constant d, in Section 3. For this case, we prove that the maximal complexity of a central trajectory $\mathcal{C}$ is $O(\tau n^{5/2})$. Computing $\mathcal{C}$ takes $O(\tau n^3)$ time and requires $O(\tau n^2 \log n)$ working space. We briefly discuss various extensions to our approach in Section 4.

Even though we do not expect this to happen in practice, the worst-case complexity of our central trajectories can be significantly higher than the input size. If this occurs, we can use traditional line simplification algorithms like Imai and Iri [23] to simplify the resulting central trajectory. This gives us a representative that still is always close —for instance within distance 2ε— to one of the input trajectories. Alternatively, we can use dynamic-programming combined with our methods to enforce the output trajectory to have at most k vertices, for any k, and always be on the input trajectories. Computing such a central trajectory is more expensive than our current algorithms, however. Furthermore, enforcing a low output complexity may not be necessary. For example, in applications like visualization, the number of trajectories shown often has a larger impact visual clutter than the length or complexity of the individual trajectories. It may be easier to follow a single trajectory that has many vertices than to follow many trajectories that have fewer vertices each.

2   Entities moving in $\mathbb{R}^1$

Let $\mathcal{X}$ be the set of entities moving in $\mathbb{R}^1$. The trajectories of these entities can be seen as polylines in $\mathbb{R}^2$: we associate time with the horizontal axis, and $\mathbb{R}^1$ with the vertical axis (see Fig. 5(a)). We observe that the distance between two points p and q in $\mathbb{R}^1$ is simply their absolute difference, that is, $\|pq\| = |p - q|$.
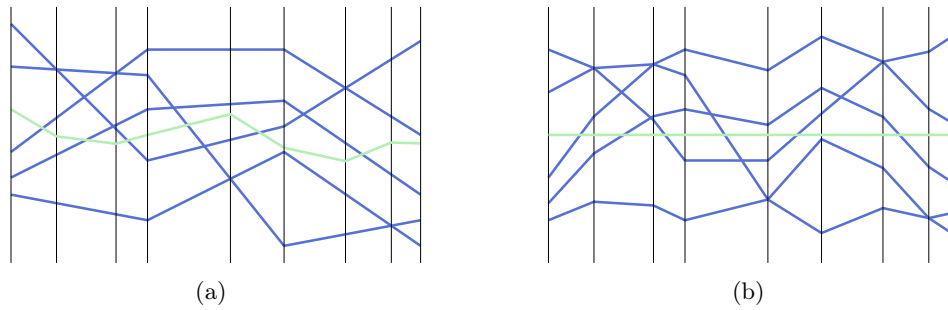
Figure 5: (a) A set of trajectories and the ideal trajectory I. The breakpoints in the ideal trajectory partition time into $O(n\tau)$ intervals. (b) The trajectories after transforming I to a horizontal line.

Let I be the *ideal* trajectory, that is, the trajectory that minimizes $\mathcal{D}$ but is not restricted to lie on the input trajectories. It follows that at any time t, $I(t)$ is simply the average of the highest entity $\mathcal{U}(t)$ and the lowest entity $\mathcal{L}(t)$. We further subdivide each time interval $J_i = [t_i, t_{i+1}]$ into *elementary intervals*, such that I is a single line segment inside each elementary interval.

Lemma 1. *The ideal trajectory I has complexity $\tau(n+2)$.*

Proof. The ideal trajectory I changes direction when $\mathcal{U}(t)$ or $\mathcal{L}(t)$ changes. During a single interval $[t_i, t_{i+1}]$ all entities move along lines, so $\mathcal{U}$ and $\mathcal{L}$ are the upper and lower envelope of a set of n lines. So by standard point-line duality, $\mathcal{U}$ and $\mathcal{L}$ correspond to the upper and lower hull of n points. The summed complexity of the upper and lower hull is at most $n+2$.
□

We assume without loss of generality that within each elementary interval I coincides with the x-axis. Note that we could realize this explicitly by transforming each trajectory σ to a trajectory $\sigma'(t) = \sigma(t) - I(t)$, see Fig. 5(b) for an illustration. To simplify the description of the proofs and algorithms, we also assume that the entities never move parallel to the ideal trajectory, that is, there are no horizontal edges.

Lemma 2. *$\mathcal{C}$ is a central trajectory in $\mathbb{R}^1$ if and only if it minimizes the function*

$$\mathcal{D}'(\mathcal{T}) = \int_{t_0}^{t_\tau} |\mathcal{T}(t)| \, \mathrm{d}t.$$

Proof. A central trajectory $\mathcal{C}$ is a trajectoid that minimizes the function

$$\mathcal{D}(\mathcal{T}) = \int_{t_0}^{t_\tau} D_{\mathcal{T}}(t) \, \mathrm{d}t = \int_{t_0}^{t_\tau} \max_{\psi \in \mathcal{X}} \|\mathcal{T}(t)\psi(t)\| \, \mathrm{d}t = \int_{t_0}^{t_\tau} \max_{\psi \in \mathcal{X}} |\mathcal{T}(t) - \psi(t)| \, \mathrm{d}t$$

$$= \int_{t_0}^{t_\tau} \max\{|\mathcal{T}(t) - \mathcal{U}(t)|, |\mathcal{T}(t) - \mathcal{L}(t)|\} \, \mathrm{d}t.$$

Since $(\mathcal{U}(t) + \mathcal{L}(t))/2 = 0$, we have that $|\mathcal{T}(t) - \mathcal{U}(t)| > |\mathcal{T}(t) - \mathcal{L}(t)|$ if and only if $\mathcal{T}(t) < 0$. So, we split the integral, depending on $\mathcal{T}(t)$, giving us

$$\mathcal{D}(\mathcal{T}) = \int_{t_0 \leq t \leq t_\tau \wedge \mathcal{T}(t) \geq 0} \mathcal{T}(t) - \mathcal{L}(t) \, \mathrm{d}t + \int_{t_0 \leq t \leq t_\tau \wedge \mathcal{T}(t) < 0} \mathcal{U}(t) - \mathcal{T}(t) \, \mathrm{d}t$$

$$= \int_{t_0 \leq t \leq t_\tau \wedge \mathcal{T}(t) \geq 0} \mathcal{T}(t) \, \mathrm{d}t - \int_{t_0 \leq t \leq t_\tau \wedge \mathcal{T}(t) \geq 0} \mathcal{L}(t) \, \mathrm{d}t +$$

$$\int_{t_0 \leq t \leq t_\tau \wedge \mathcal{T}(t) < 0} \mathcal{U}(t) \, \mathrm{d}t - \int_{t_0 \leq t \leq t_\tau \wedge \mathcal{T}(t) < 0} \mathcal{T}(t) \, \mathrm{d}t.$$

We now use that $-\int_{\mathcal{T}(t) < 0} \mathcal{T}(t) = \int_{\mathcal{T}(t) < 0} |\mathcal{T}(t)|$, and that $-\int \mathcal{L}(t) = \int \mathcal{U}(t)$ (since $(\mathcal{U}(t) + \mathcal{L}(t))/2 = 0$). After rearranging the terms we then obtain

$$\mathcal{D}(\mathcal{T}) = \int_{t_0 \leq t \leq t_\tau \wedge \mathcal{T}(t) \geq 0} \mathcal{T}(t) \, \mathrm{d}t + \int_{t_0 \leq t \leq t_\tau \wedge \mathcal{T}(t) < 0} |\mathcal{T}(t)| \, \mathrm{d}t +$$

$$\int_{t_0 \leq t \leq t_\tau \wedge \mathcal{T}(t) \geq 0} \mathcal{U}(t) \, \mathrm{d}t + \int_{t_0 \leq t \leq t_\tau \wedge \mathcal{T}(t) < 0} \mathcal{U}(t) \, \mathrm{d}t$$

$$= \int_{t_0 \leq t \leq t_\tau} |\mathcal{T}(t)| \, \mathrm{d}t + \int_{t_0 \leq t \leq t_\tau} \mathcal{U}(t) \, \mathrm{d}t.$$

The last term is independent of $\mathcal{T}$, so we have $\mathcal{D}(\mathcal{T}) = \mathcal{D}'(\mathcal{T}) + c$, for some $c \in \mathbb{R}$. The lemma follows. □

By Lemma 2 a central trajectory $\mathcal{C}$ is a trajectoid $\mathcal{T}$ that minimizes $\mathcal{D}'$. Observe that since I coincides with the x-axis, $\mathcal{D}'(\mathcal{T})$ corresponds to the area $\mathcal{D}'(\mathcal{T})$ between $\mathcal{T}$ and the ideal trajectory I. Hence, we can focus on finding a trajectoid that minimizes $\mathcal{D}'(\mathcal{T})$.

2.1 Complexity

Lemma 3. *For a set of n trajectories in $\mathbb{R}^1$, each with vertices at times $t_0, .., t_\tau$, a central trajectory $\mathcal{C}$ may have worst case complexity $\Omega(\tau n^2)$.*

Proof. We describe a construction for the entities that shows that within a single time interval $J = [t_i, t_{i+1}]$ the complexity of $\mathcal{C}$ may be $\Omega(n^2)$. Repeating this construction τ times gives us $\Omega(\tau n^2)$ as desired.

Within J the entities move linearly. So we construct an arrangement $\mathcal{A}$ of lines that describes the motion of all entities. We place $m = n/3$ lines such that the upper envelope of $\mathcal{A}$ has linear complexity. We do the same for the lower envelope. We position these lines such that the ideal trajectory I—which is the average of the upper and lower envelope—makes a vertical "zigzagging" pattern (see Fig. 6). The remaining set H of m lines are almost horizontal (i.e. we can perturb them by an arbitrarily small amount to avoid degeneracies). Two consecutive lines are placed at (vertical) distance at most ε. We place all lines such that they all intersect I. It follows that $\mathcal{C}$ jumps $\Omega(n^2)$ times between the lines in H (as is shown in Fig. 6). The lemma follows. □
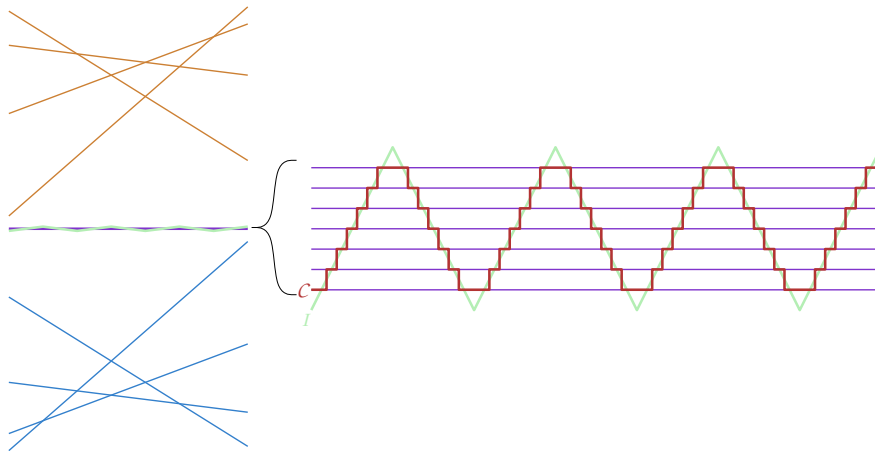
Figure 6: Lower bound construction that shows that $\mathcal{C}$ (red) may have quadratic complexity. The ideal trajectory I is shown in green.

Recall that two entities σ and ψ are ε-connected at time t if there is a sequence $\sigma = \sigma_0, .., \sigma_k = \psi$ of entities such that for all i, subsequent entities σ_i and σ_{i+1} are within distance ε of each other at time t. A subset $\mathcal{X}' \subseteq \mathcal{X}$ of entities is ε-connected at time t if all entities in $\mathcal{X}'$ are pairwise ε-connected at time t. The set $\mathcal{X}'$ is ε-connected during an interval I, if they are ε-connected at any time $t \in I$. We now observe:

Observation 4. *$\mathcal{C}$ can jump from entity σ to ψ at time t if and only if σ and ψ are ε-connected at time t.*

At any time t, we can partition $\mathcal{X}$ into maximal sets of ε-connected entities. The central trajectory $\mathcal{C}$ must be in one of such maximal sets $\mathcal{X}'$: it uses the trajectory of an entity $\sigma \in \mathcal{X}'$ (at time t), if and only if σ is the entity from $\mathcal{X}'$ closest to I. More formally, let $f_\sigma(t) = |\sigma(t) - I(t)|$ (and note that, since I coincides with the x-axis this simplifies to $f_\sigma(t) = |\sigma(t)|$ for the transformed trajectories), and let $\mathcal{L}(\mathcal{F}) = \min_{f \in \mathcal{F}} f$ denote the lower envelope of a set of functions $\mathcal{F}$.

Observation 5. *Let σ be an entity, and let $\mathcal{X}'$ be a maximal set of entities containing σ that is ε-connected during interval J, and assume that $\mathcal{C} \in \mathcal{X}'$ during J. For any time $t \in J$, we have that $\mathcal{C}(t) = \sigma(t)$ if and only if f_σ is on the lower envelope of the set $\mathcal{F}' = \{f_\psi \mid \psi \in \mathcal{X}'\}$ at time t, that is, $f_\sigma(t) = \mathcal{L}(\mathcal{F}')(t)$.*

Let $\mathcal{X}_1, .., \mathcal{X}_m$ denote a collection of maximal sets of entities that are ε-connected during time intervals $J_1, .., J_m$, respectively. Let $\mathcal{F}_i = \{f_\sigma \mid \sigma \in \mathcal{X}_i\}$, and let $\mathcal{L}_i$ be the lower envelope $\mathcal{L}(\mathcal{F}_i)$ of $\mathcal{F}_i$ restricted to interval J_i. A lower envelope $\mathcal{L}_i$ has a break point at time t if $f_\sigma(t) = f_\psi(t)$, for $\sigma, \psi \in \mathcal{X}_i$. There are two types of break points: (i) $\sigma(t) = \psi(t)$, or (ii) $\sigma(t) = -\psi(t)$. At events of type (i) the transformed trajectories of σ and ψ intersect. At events of the type (ii), σ and ψ are equally far from I, but on different sides of I. Let $B = \{(t, \sigma, \psi) \mid \mathcal{L}_i(t) = f_\sigma(t) = f_\psi(t) \wedge i \in \{1, .., m\}\}$ denote the collection of break points from all lower envelopes $\mathcal{L}_1, .., \mathcal{L}_m$.

Lemma 6. *Consider a triplet $(t, \sigma, \psi) \in B$. There is at most one lower envelope $\mathcal{L}_i$ such that t is a break point in $\mathcal{L}_i$.*

Proof. Assume by contradiction that t is a break point in both $\mathcal{L}_i$ and $\mathcal{L}_j$. At any time t, an entity can be in at most one maximal set $\mathcal{X}_\ell$. So if $\mathcal{X}_i$ and $\mathcal{X}_j$ share either entity σ or ψ, then the intervals J_i and J_j are disjoint. It follows t cannot lie in both intervals, and thus cannot be a break point in both $\mathcal{L}_i$ and $\mathcal{L}_j$. Contradiction. $\qquad\square$

Lemma 7. *Let $\mathcal{A}$ be an arrangement of n lines, describing the movement of n entities during an elementary interval J. If there is a break point $(t, \sigma, \psi) \in B$, with $t \in J$, of type (ii), then $\sigma(t)$ and $\psi(t)$ lie on the boundary $\partial \mathcal{Z}$ of the zone $\mathcal{Z}$ of I in $\mathcal{A}$.*

Proof. Let $\mathcal{X}_j$ be the maximal ε-connected set containing σ and ψ, and assume without loss of generality that $f_\sigma(t) = \sigma(t) = -\psi(t) = f_\psi(t)$. Now, assume by contradiction that σ is not on $\partial \mathcal{Z}$ at time t (the case that $\psi(t)$ is not on $\partial \mathcal{Z}$ is symmetric). This means that there is an entity ρ with $0 \le \rho(t) < \sigma(t)$. If $\rho \in \mathcal{X}_j$, this contradicts that $f_\sigma(t)$ was on the lower envelope of $\mathcal{X}_j$ at time t. So ρ is not ε-connected to σ at time t. Hence, their distance is at least ε. We then have $\sigma(t) > \rho(t) + \varepsilon > \varepsilon$. It now follows that σ and ψ cannot be ε-connected at time t: the distance between σ and ψ is bigger than ε so they are not directly connected, and f_σ and f_ψ are on $\mathcal{L}_j$, so there are also no other entities in $\mathcal{X}_j$ through which they can be ε-connected. If σ and ψ are not both in $\mathcal{X}_j$, they cannot contribute to a break point of $\mathcal{L}_j$. Contradiction. $\qquad\square$

Lemma 8. *Let $\mathcal{A}$ be an arrangement of n lines, describing the movement of n entities during an elementary interval J. The total number of break points $(t, \sigma, \phi) \in B$, with $t \in J$, of type (ii) is at most $6.5n$.*

Proof. By Lemma 6 all break points can be charged to exactly one set $\mathcal{X}_j$. From Lemma 7 it follows that break points of type (ii) involve only entities whose lines in $\mathcal{A}$ participate in the zone of I.

Figure 7: The jumps of $\mathcal{L}$ (dashed arrows) involving edges e and g.

Let E be the set of edges of $\partial \mathcal{Z}$. We have that $|E| \le 5.5n$ [8, 26]. We now split every edge that intersects I, at the intersection point. Since every line intersects I at most once, this means the number of edges in E increases to $6.5n$. For every pair of edges (e, g), that lie on opposite sides of I, there is at most one time t where a lower envelope $\mathcal{L} = \mathcal{L}_j$, for some j, has a break point of type (ii).

Consider a break point of type (ii), that is, a time t such that $\mathcal{L}$ switches (jumps) from an entity σ to an entity ψ, with σ and ψ on opposite sides of I. Let $e \in E$ and $g \in E$ be the edges containing $\sigma(t)$ and $\psi(t)$, respectively. If the arriving edge g has not been charged before, we charge the jump to g. Otherwise, we charge it to e. We continue to show that every edge in E is charged at most once. Since E has at most $6.5n$ edges, the number of break points of type (ii) is also at most $6.5n$.
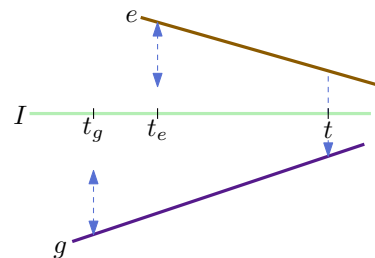
We now show that either e or g has not been charged before. Assume, by contradiction, that both e and g have been charged before time t, at times t_e and t_g, respectively. Consider the case that $t_g < t_e$ (see Fig. 7). At time t_e, the lower envelope $\mathcal{L}$ jumps from an edge h onto e or vice versa. Since there is a jump involving edge g at time t_g and one at time t it follows that at time t_e, g is the closest edge in E opposite to e. Hence, $h = g$. This means we jump twice between e and g. Contradiction. The case $t_e < t_g$ is symmetrical and the case $t_e = t_g$ cannot occur. It follows that e or g was not charged before time t, and thus all edges in E are charged at most once. $\qquad\square$

Lemma 9. *The total complexity of all lower envelopes $\mathcal{L}_1, .., \mathcal{L}_m$ on $[t_i, t_{i+1}]$ is $O(n^2)$.*

Proof. The break points in the lower envelopes are either of type (i) or of type (ii). We now show that there are at most $O(n^2)$ break points of either type.

The break points of type (i) correspond to intersections between the trajectories of two entities. Within interval $[t_i, t_{i+1}]$ the entities move along lines, hence there are at most $O(n^2)$ such intersections. By Lemma 6 all break points can be charged to exactly one set $\mathcal{X}_i$. It follows that the total number of break points of type (i) is $O(n^2)$.

To show that the number of events of the second type is at most $O(n^2)$ as well we divide $[t_i, t_{i+1}]$ in $O(n)$ elementary intervals such that I coincides with the x-axis. By Lemma 8 each such elementary interval contains at most $O(n)$ break points of type (ii). $\qquad\square$

Theorem 10. *Given a set of n trajectories in $\mathbb{R}^1$, each with vertices at times $t_0, .., t_\tau$, a central trajectory $\mathcal{C}$ has worst case complexity $O(\tau n^2)$.*

Proof. A central trajectory $\mathcal{C}$ is a piecewise linear function. From Observation 5 it now follows that $\mathcal{C}$ has a break point at time t only if (a) two subsets of entities become ε-connected or ε-disconnected, or (b) the lower envelope of a set of ε-connected entities has a break point at time t. Within a single time interval $J_i = [t_i, t_{i+1}]$ there are at most $O(n^2)$ times when two entities are at distance exactly ε. Hence, the number of events of type (a) during interval J_i is also $O(n^2)$. By Lemma 9 the total complexity of all lower envelopes of ε-connected sets during J_i is $O(n^2)$. Hence, the number of break points of type (b) within interval J_i is also $O(n^2)$. The theorem follows. $\qquad\square$

2.2   Algorithm

We now present an algorithm to compute a trajectoid $\mathcal{C}$ minimizing $\mathcal{D}'$. By Lemma 2 such a trajectoid is a central trajectory. The basic idea is to construct a weighted (directed acyclic) graph that represents a set of trajectoids containing $\mathcal{C}$. We can then find $\mathcal{C}$ by computing a minimum weight path in this graph.

The graph that we use is a weighted version of the Reeb graph that Buchin et al. [12] use to model the trajectory grouping structure. We review their definition here. The *Reeb graph* $\mathcal{R}$ is a directed acyclic graph. Each edge $e = (u, v)$ of $\mathcal{R}$ corresponds to a maximal subset of entities $C_e \subseteq \mathcal{X}$ that is ε-connected during the time interval $[t_u, t_v]$. We refer to such a subset C_e as a *component*. The vertices represent times at which the sets of ε-connected entities

change, that is, the times at which two entities σ and ψ are at distance ε from each other and the set containing σ merges with or splits from the set containing ψ. See Fig. 8. for an illustration.

By Observation 4 a central trajectory $\mathcal{C}$ can jump from σ to ψ if and only if σ and ψ are ε-connected, that is, if σ and ψ are in the same component C_e of edge e. From Observation 5 it follows that on each edge e, $\mathcal{C}$ uses only the trajectories of entities σ for which f_σ occurs on the lower envelope of the functions $\mathcal{F}_e = \{f_\sigma \mid \sigma \in C_e\}$. Hence, we can then express the cost for $\mathcal{C}$ using edge e by

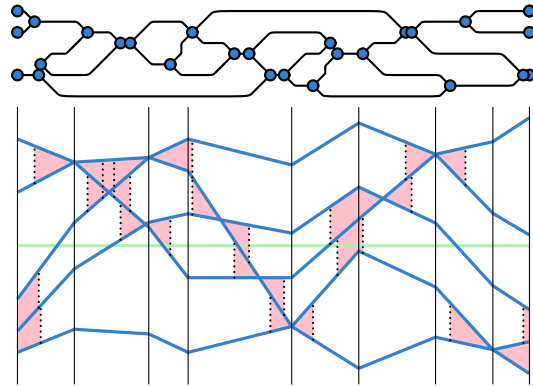$$\omega_e = \int_{t_u}^{t_v} \mathcal{L}(\mathcal{F}_e)(t)\,\mathrm{d}t.$$

Figure 8: The Reeb graph for the moving entities from Fig. 5(b). The pink areas indicate that two entities are at distance $\leq \varepsilon$.

Thus, a central trajectory $\mathcal{C}$ follows a path in the Reeb graph $\mathcal{R}$. In other words, the set of trajectoids represented by $\mathcal{R}$ contains a trajectoid minimizing $\mathcal{D}'$. We can compute a central trajectory by finding a minimum weight path in $\mathcal{R}$ from a source to a sink.

Analysis. First we compute the Reeb graph as defined by Buchin et al. [12]. This takes $O(\tau n^2 \log n)$ time. Second we compute the weight ω_e for each edge e. The Reeb graph $\mathcal{R}$ is a DAG, so once we have the edge weights, we can use dynamic programming to compute a minimum weight path in $O(|\mathcal{R}|) = O(\tau n^2)$ time. So all that remains is to compute the edge weights ω_e. For this, we need the lower envelope $\mathcal{L}_e$ of each set $\mathcal{F}_e$ on the interval J_e. To compute the lower envelopes, we need the ideal trajectory I, which we can compute in $O(\tau n \log n)$ time by computing the lower and upper envelope of the trajectories in each time interval $[t_i, t_{i+1}]$.

Lemma 9 implies that the total complexity of all lower envelopes is $O(\tau n^2)$. To compute them we have two options. We can simply compute the lower envelope from scratch for every edge of $\mathcal{R}$. This takes $O(\tau n^2 \cdot n \log n) = O(\tau n^3 \log n)$ time. Instead, for each time interval $J_i = [t_i, t_{i+1}]$, we compute the arrangement $\mathcal{A}$ representing the straightened trajectories on the interval J_i, and use it to trace $\mathcal{L}_e$ in $\mathcal{A}$ for every edge e of $\mathcal{R}$.

Using a standard sweep line algorithm, an arrangement of m line segments can be built in $O((m + A) \log m)$ time, where A is the output complexity. We have $O(n^2)$ line segments: $n + 2$ per entity. Since each pair of trajectories intersects at most once during J_i, we have that $A = O(n^2)$. Thus, we can build $\mathcal{A}$ in $O(n^2 \log n)$ time. The arrangement $\mathcal{A}$ represents all break points of type (i), of all functions f_σ. We now compute all pairs of points in $\mathcal{A}$ corresponding to break points of type (ii). We do this in $O(n^2)$ time by traversing the zone of I in $\mathcal{A}$.

We can then trace the lower envelopes through $\mathcal{A}$: for each edge $e = (u, v)$ in the Reeb graph with $J_e \subseteq J_i$, we start at the point $\sigma(t_u)$, $\sigma \in C_e$, that is closest to I, and follow the edges in $\mathcal{A}$ corresponding to $\mathcal{L}_e$, taking care to jump when we encounter break points of
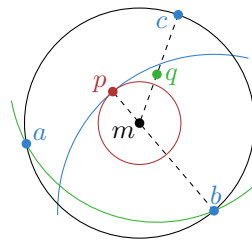
Figure 9: Point p is closest to the ideal point m, however the smallest enclosing disk centered at q is smaller than that of p.

type (ii). Our lower envelopes are all disjoint (except at endpoints), so we traverse each edge in $\mathcal{A}$ at most once. The same holds for the jumps. We can avoid costs for searching for the starting point of each lower envelope by tracing the lower envelopes in the right order: when we are done tracing $\mathcal{L}_e$, with $e = (u, v)$, we continue with the lower envelope of an outgoing edge of vertex v. If v is a split vertex where σ and ψ are at distance ε, then the starting point of the lower envelope of the other edge is either $\sigma(t_v)$ or $\psi(t_v)$, depending on which of the two is farthest from I. It follows that when we have $\mathcal{A}$ and the list of break points of type (ii), we can compute all lower envelopes in $O(n^2)$ time. We conclude:

Theorem 11. *Given a set of n trajectories in $\mathbb{R}^1$, each with vertices at times $t_0, .., t_\tau$, we can compute a central trajectory $\mathcal{C}$ in $O(\tau n^2 \log n)$ time using $O(\tau n^2)$ space.*

3   Entities moving in $\mathbb{R}^d$

In the previous section, we used the ideal trajectory I, which minimizes the distance to the farthest entity, ignoring the requirement to stay on an input trajectory. The problem was then equivalent to finding a trajectoid that minimizes the distance to the ideal trajectory. In $\mathbb{R}^d$, with $d > 1$, however, this approach fails, as the following example shows.

Observation 12. *Let P be a set of points in $\mathbb{R}^2$. The point in P that minimizes the distance to the ideal point (i.e., the center of the smallest enclosing disk of P) is not necessarily the same as the point in P that minimizes the distance to the farthest point in P.*

Proof. See Fig. 9. Consider three points a, b and c at the corners of an equilateral triangle, and two points p and q close to the center m of the circle through a, b and c. Now p is closer to m than q, yet q is closer to b than p, and q is as far from a as from b.   □

3.1   Complexity

It follows from Lemma 3 that the complexity of a central trajectory for entities moving in $\mathbb{R}^d$ is at least $\Omega(\tau n^2)$. In this section, we prove that the complexity of $\mathcal{C}$ within a single time interval $[t_i, t_{i+1}]$ is at most $O(n^{5/2})$. Thus, the complexity over all τ time intervals is $O(\tau n^{5/2})$.

Recall that D_σ is a function expressing the distance from σ to the entity farthest away from σ. Let $\mathcal{F}$ denote the collection of functions D_σ, for $\sigma \in \mathcal{X}$. We partition time into intervals $J_1', .., J_{k'}'$ such that in each interval J_i' all functions D_σ restricted to J_i' are *simple*, that is, they contain no breakpoints, and thus consist of just one piece. We now show that each function D_σ consists of at most $\tau(2n-1)$ pieces, and thus the total number of intervals is at most $O(\tau n^2)$. See Fig. 3(b) for an illustration.

Lemma 13. *Each function D_σ is piecewise hyperbolic and consists of at most $\tau(2n-1)$ pieces.*

Proof. Consider a time interval $J_i = [t_i, t_{i+1}]$. For any entity ψ and any time $t \in J_i$, the function $\|\sigma(t)\psi(t)\| = \sqrt{at^2 + bt + c}$, with $a, b, c \in \mathbb{R}$, is hyperbolic in t. Each pair of such functions can intersect at most twice. During J_i, D_σ is the upper envelope of these functions, so it consists of $\lambda_2(n)$ pieces, where λ_s denotes the maximum complexity of a Davenport-Schinzel sequence of order s [4]. We have $\lambda_2(n) = 2n - 1$, so the lemma follows. $\square$

Lemma 14. *The total number of intersections of all functions in $\mathcal{F}$ is at most $O(\tau n^3)$.*

Proof. Fix a pair of entities σ, ψ. By Lemma 13 there are at most $\tau(2n-1)$ time intervals J, such that D_σ restricted to J is simple. The same holds for D_ψ. So, there are at most $\tau(4n-2)$ intervals in which both D_σ and D_ψ are simple (and hyperbolic). In each interval D_σ and D_ψ intersect at most twice. $\square$

We again observe that $\mathcal{C}$ can only jump from one entity to another if they are ε-connected. Hence, Observation 4 holds for entities moving in $\mathbb{R}^d$ as well. As before, this means that at any time t, we can partition $\mathcal{X}$ into maximal sets of ε-connected entities. Let $\mathcal{X}' \ni \sigma$ be a maximal subset of ε-connected entities at time t. This time, a central trajectory $\mathcal{C}$ uses the trajectory of entity σ at time t, if and only if σ is the entity from $\mathcal{X}'$ whose function $\mathcal{D}_\sigma$ is minimal. Hence, if we define $f_\sigma = D_\sigma$, then Observation 5 holds again as well.

Consider all $m' = O(n^2)$ intervals $J_1', .., J_{m'}'$ that together form $[t_j, t_{j+1}]$. We subdivide these intervals at points where the distance between two entities is exactly ε. Let $J_1, .., J_m$ denote the set of resulting intervals. Since there are $O(n^2)$ times at which two entities are at distance exactly ε, we still have $O(n^2)$ intervals. Note that for all intervals J_i and all entities σ, f_σ is simple and totally defined on J_i.

In each interval J_i, a central trajectory $\mathcal{C}$ uses the trajectories of only one maximal set of ε-connected entities. Let $\mathcal{X}_i'$ be this set, let $\mathcal{F}_i' = \{f_\sigma \mid \sigma \in \mathcal{X}_i'\}$ be the set of corresponding functions, and let $\mathcal{L}_i$ be the lower envelope of $\mathcal{F}_i'$, restricted to interval J_i. We now show that the total complexity of all these lower envelopes is $O(n^{5/2})$. It follows that the maximal complexity of $\mathcal{C}$ in J_i is at most $O(n^{5/2})$ as well.

Lemma 15. *Let J be an interval, let $\mathcal{F}$ be a set of hyperbolic functions that are simple and totally defined on J, and let k denote the complexity of the lower envelope $\mathcal{L}$ of $\mathcal{F}$ restricted to J. Then the interval J contains $\Omega(k^2)$ intersections of functions in $\mathcal{F}$ that do not lie on $\mathcal{L}$.*

Proof. Let $\mathcal{L} = L_1, .., L_k$ denote the pieces of the lower envelope, ordered from left to right. Consider any subsequence $\mathcal{L}' = L_1, ..., L_i$ of the pieces. The functions in $\mathcal{F}$ are all hyperbolic, so every pair of functions intersect at most twice. Therefore $\mathcal{L}'$ consists of at most $\lambda_2(|\mathcal{F}|) = 2|\mathcal{F}| - 1$ pieces. Hence, $i \leq 2|\mathcal{F}| - 1$. The same argument gives us that there must be at least $\ell_i = \lfloor (i+1)/2 \rfloor$ distinct functions of $\mathcal{F}$ contributing to $\mathcal{L}'$.

Figure 10: The function f (blue) that defines segment L_i of $\mathcal{L}$ intersects with at least $\ell_{i-2} = \lfloor (i-1)/2 \rfloor$ other functions from $\mathcal{F}_i$ in interval J.

Consider a piece $L_i = [a, b]$ such that a is the first time that a function f contributes to the lower envelope. That is, a is the first time such that $f(t) = \mathcal{L}(t)$. Clearly, there are at least ℓ_k such pieces. Furthermore, there are at least ℓ_{i-2} distinct functions corresponding to the pieces $L_1, .., L_{i-2}$. Let $\mathcal{F}_i$ denote the set of those functions.

All functions in $\mathcal{F}$ are continuous and totally defined, so they span time interval J. It follows that all functions in $\mathcal{F}_i$ must intersect f at some time after the start of interval J, and before time a. Since a was the first time that f lies on $\mathcal{L}$, all these intersection points do not lie on $\mathcal{L}$. See Fig. 10. In total we have at least

$$\sum_{i=2}^{\ell_k} \ell_{i-2} = \sum_{i=2}^{\lfloor (k+1)/2 \rfloor} \lfloor (i-1)/2 \rfloor = \sum_{i=1}^{\lfloor (k+1)/2 \rfloor - 1} \lfloor i/2 \rfloor = \Omega(k^2)$$

such intersections. $\qquad\square$

Lemma 16. *Let $\mathcal{F}_1, .., \mathcal{F}_m$ be a collection of m sets of partial functions, let n be the total number of functions, and let $J_1, .., J_m$ be a collection of intervals such that:*

- *the total number of intersections between functions in $\mathcal{F}_1, .., \mathcal{F}_m$ is at most $O(n^3)$,*
- *for any two intersecting intervals J_i and J_j, $\mathcal{F}_i$ and $\mathcal{F}_j$ are disjoint, and*
- *for every set $\mathcal{F}_i$, all functions in $\mathcal{F}_i$ are simple, hyperbolic, and totally defined on J_i.*

Let $\mathcal{L}_i$ denote the lower envelope of $\mathcal{F}_i$ restricted to J_i. The total complexity of the lower envelopes $\mathcal{L}_1, .., \mathcal{L}_m$ is $O((m + n^2)\sqrt{n})$.

Proof. Let k_i denote the complexity of the lower envelope $\mathcal{L}_i$. An interval J_i is *heavy* if $k_i > \sqrt{n}$ and *light* otherwise. Clearly, the total complexity of all light intervals is at most $O(m\sqrt{n})$. What remains is to bound the complexity of all heavy intervals.

Relabel the intervals such that $J_1, .., J_h$ are the heavy intervals. By Lemma 15 we have that in each interval J_i, there are at least ck_i^2 intersections involving the functions $\mathcal{F}_i$, for some $c \in \mathbb{R}$.

Since for every pair of intervals J_i and J_j that overlap the sets $\mathcal{F}_i$ and $\mathcal{F}_j$ are disjoint, we can associate each intersection with at most one interval. There are at most $O(n^3)$ intersections in total, thus we have $c'n^3 \geq \sum_{i=1}^m ck_i^2 \geq \sum_{i=1}^h ck_i^2$, for some $c' \in \mathbb{R}$. Using that
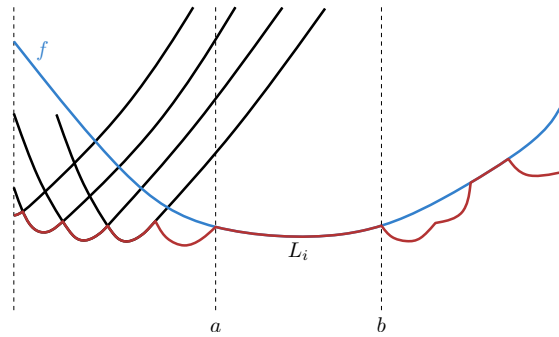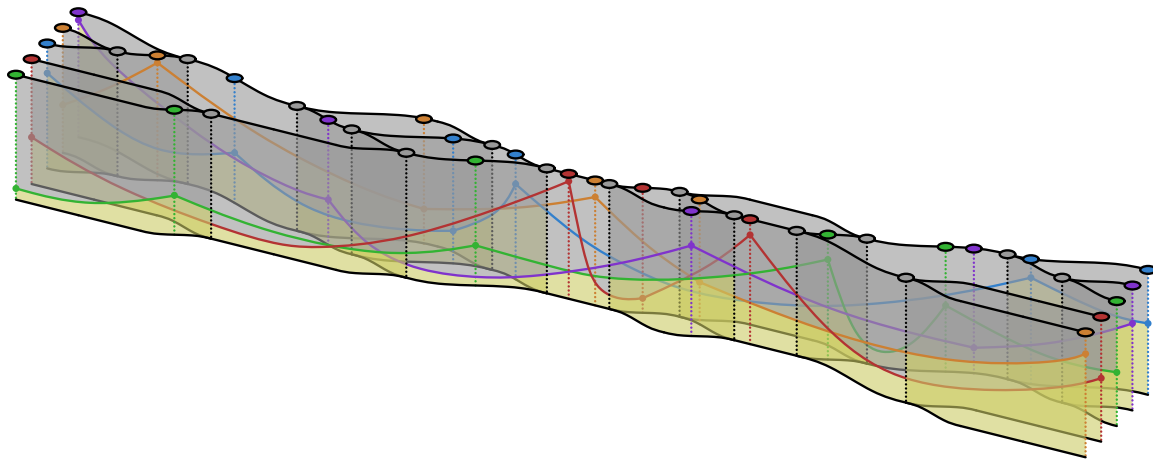
Figure 11: The modified Reeb graph $\mathcal{R}$ for five entities moving during a single interval, and the corresponding functions f_σ for each entity σ.

for all heavy intervals $k_i > \sqrt{n}$ we obtain

$$c'n^3 \geq \sum_{i=1}^{h} ck_i^2 \geq \sum_{i=1}^{h} c\sqrt{n}k_i = c\sqrt{n} \sum_{i=1}^{h} k_i.$$

It follows that the total complexity of the heavy intervals is $\sum_{i=1}^{h} k_i \leq c'n^3/c\sqrt{n} = O(n^2\sqrt{n})$. $\hfill\square$

By Lemma 14 we have that the number of intersections between functions in $\mathcal{F}$ in time interval $[t_j, t_{j+1}]$ is $O(n^3)$. Hence, the total number of intersections over all functions in all sets $\mathcal{F}'_i$ is also $O(n^3)$. All functions in each set $\mathcal{F}'_i$ are simple and totally defined on J_i, and all intervals $J_1, .., J_m$ are pairwise disjoint, so we can use Lemma 16. It follows that the total complexity of $\mathcal{L}'_1, .., \mathcal{L}'_m$ is at most $O(n^{5/2})$. Thus, in a single time interval the worst case complexity of $\mathcal{C}$ is also at most $O(n^{5/2})$. We conclude:

Theorem 17. *Given a set of n trajectories in $\mathbb{R}^d$, each with vertices at times $t_0, .., t_\tau$, a central trajectory $\mathcal{C}$ has worst case complexity $O(\tau n^{5/2})$.*

3.2 Algorithm

We use the same global approach as in Section 2.2: we represent a set of trajectoids containing an optimal solution by a graph, and then compute a minimum weight path in this graph. The graph that we use, is a slightly modified Reeb graph. We split an edge e into two edges at time t if there is an entity $\sigma \in C_e$ such that $D_\sigma = f_\sigma$ has a break point at time t. All functions f_σ, with $\sigma \in C_e$, are now simple and totally defined on J_e. This process adds a total of $O(\tau n^2)$ degree-two vertices to the Reeb graph. Let $\mathcal{R}$ denote the resulting Reeb graph (see Fig. 11).

To find all the times where we have to insert vertices, we explicitly compute the functions D_σ. This takes $O(\tau n \lambda_2(n) \log n) = O(\tau n^2 \log n)$ time, where λ_s denotes the

maximum complexity of a Davenport-Schinzel sequence of order s [4], since within each time interval $[t_i, t_{i+1}]$ each D_σ is the upper envelope of a set of n functions that intersect each other at most twice. After we sort these break points in $O(\tau n^2 \log n)$ time, we can compute the modified Reeb graph $\mathcal{R}$ in $O(\tau n^2 \log n)$ time [12].[1]

Next, we compute all weights ω_e, for each edge e. This means we have to compute the lower envelope $\mathcal{L}_e$ of the functions $\mathcal{F}_e = \{f_\sigma \mid \sigma \in C_e\}$ on the interval J_e. All these lower envelopes have a total complexity of at most $O(\tau n^{5/2})$:

Lemma 18. *The total complexity of the lower envelopes for all edges of the Reeb graph is* $O(\tau n^{5/2})$.

Proof. We consider each time interval $J_i = [t_i, t_{i+1}]$ separately. Let $\mathcal{R}_i$ denote the Reeb graph restricted to J_i. We now show that for each $\mathcal{R}_i$, the total complexity of all lower envelopes $\mathcal{L}_e$ of edges e in $\mathcal{R}_i$ is $O(n^2\sqrt{n})$. The lemma then follows.

By Lemma 14, the total number of intersections of all functions $\mathcal{F}_e$, with e in $\mathcal{R}_i$, is $O(n^3)$. Each set $\mathcal{F}_e$ corresponds to an interval J_e, on which all functions in $\mathcal{F}_e$ are simple and totally defined. Furthermore, at any time, every entity is in at most one component C_e. So, if two intervals J_e and $J_{e'}$ overlap, the sets of entities C_e and $C_{e'}$, and thus also the sets of functions $\mathcal{F}_e$ and $\mathcal{F}_{e'}$ are disjoint. It follows that we can apply Lemma 16. Since $\mathcal{R}_i$ has $O(n^2)$ edges the total complexity of all lower envelopes is $O(n^2\sqrt{n})$. $\square$

We again have two options to compute all lower envelopes: either we compute all of them from scratch in $O(\tau n^2 \cdot \lambda_2(n) \log n) = O(\tau n^3 \log n)$ time, or we use a similar approach as before. For each time interval, we compute the arrangement $\mathcal{A}$ of all functions $\mathcal{F}$, and then trace $\mathcal{L}_e$ in $\mathcal{A}$ for every edge e. For n^2 functions that pairwise intersect at most twice, the arrangement can be built in $O(n^2 \log n + A)$ time, where A is the output complexity [6]. The complexity of $\mathcal{A}$ is $O(n^3)$, so we can construct it in $O(n^3)$ time. As before, every edge is traversed at most once so tracing all lower envelopes $\mathcal{L}_e$ takes at most $O(n^3)$ time. It follows that we can compute all edge weights in $O(\tau n^3)$ time, using $O(n^3)$ working space.

Computing a minimum weight path takes $O(\tau n^2)$ time, and uses $O(\tau n^2)$ space as before. Thus, we can compute $\mathcal{C}$ in $O(\tau n^3)$ time and $O(n^3 + \tau n^2)$ space.

Reducing the required working space.   We can reduce the amount of working space required to $O(n^2 \log n + \tau n^2)$ as follows. Consider computing the edge weights in the time interval $J = [t_i, t_{i+1}]$. Interval J is subdivided into $O(n^2)$ smaller intervals $J_1, .., J_m$ as described in Section 3.1. We now consider groups of r consecutive intervals. Let J be the union of r consecutive intervals, we compute the arrangement $\mathcal{A}$ of the functions $\mathcal{F}$, restricted to time interval J. Since every interval J_i has at most $O(n^2)$ intersections $\mathcal{A}$ has worst case complexity $O(rn^2)$. Thus, at any time we need at most $O(rn^2)$ space to store the arrangement. In total this takes $O(\sum_{i=1}^{n^2/r} (n_i \log n_i + A_i))$ time, where n_i is the number of functions in the ith group of intervals, and A_i is the complexity of the arrangement in group i. The total complexity of all arrangements is again $O(n^3)$. Since we cut each function D_σ into an additional $O(n^2/r)$ pieces, the total number of functions is $O(n^3/r + n^2)$. Hence, the

---

[1] The algorithm to compute the Reeb graph is presented for entities moving in $\mathbb{R}^2$ in [12], but it immediately extends to entities moving in $\mathbb{R}^d$.

total running time is $O((n^3/r)\log n + n^3)$. We now choose $r = \Theta(\log n)$ to compute all edge weights in $[t_i, t_{i+1}]$ in $O(n^3)$ time and $O(n^2 \log n)$ space. We conclude:

Theorem 19. *Given a set of n trajectories in $\mathbb{R}^d$, each with vertices at times $t_0, .., t_\tau$, we can compute a central trajectory $\mathcal{C}$ in $O(\tau n^3)$ time using $O(n^2 \log n + \tau n^2)$ space.*

4   Extensions

We now briefly discuss how our results on entities moving in $\mathbb{R}^d$, $d > 1$, can be extended in various directions.

Other measures of centrality. We based our central trajectory on the center of the smallest enclosing disk of a set of points. Instead, we could choose other static measures of centrality, such as the Fermat-Weber point, which minimizes the sum of distances to the other points, or the center of mass, which minimizes the sum of squared distances to the other points. In both cases we can use the same general approach as described in Section 3.

Let $\hat{D}_\sigma^2(t) = \sum_{\psi \in \mathcal{X}} \|\sigma(t)\psi(t)\|^2$ denote the sum of the squared Euclidean distances from σ to all other entities at time t. This function $\hat{D}_\sigma^2$ is piecewise quadratic in t, and consists of (only) $O(\tau)$ pieces. It follows that the total number of intersections between all functions $\hat{D}_\sigma^2$, $\sigma \in \mathcal{X}$, is at most $O(\tau n^2)$. We again split the domain of these functions into elementary intervals. The Reeb graph $\mathcal{R}$ representing the ε-connectivity of the entities still has $O(\tau n^2)$ vertices and edges. Each vertex of $\mathcal{C}$ corresponds either to an intersection between two functions $\hat{D}_\sigma^2$ and $\hat{D}_\psi^2$, or to a jump, occurring at a vertex of $\mathcal{R}$. It now follows that $\mathcal{C}$ has complexity $O(\tau n^2)$.

To compute a central trajectory, we compute a shortest path in the (weighted) Reeb graph $\mathcal{R}$. To compute the weights we again construct the arrangement of all curves $\hat{D}_\sigma^2$, and trace the lower envelope $\mathcal{L}_e$ of the curves associated to each edge $e \in \mathcal{R}$. This takes $O(\tau n^2 \log n)$ time in total.

Next, we consider the sum of Euclidean distances $\hat{D}_\sigma(t) = \sum_{\psi \in \mathcal{X}} \|\sigma(t)\psi(t)\|$. For $d = 1$, we have $\|\sigma(t)\psi(t)\| = |\sigma(t) - \psi(t)|$. When n is odd, it then follows that a trajectoid that minimizes $\hat{D}_\sigma$ coincides with the $\lceil n/2 \rceil$ level of the trajectory arrangement $\mathcal{A}$. When n is even an optimal trajectoid stays between the $\lfloor n/2 \rfloor$ level $\lceil n/2 \rceil$ level of $\mathcal{A}$. In both cases, we can then show that the complexity is bounded by the complexity of these levels: $O(\tau n^{4/3})$ [15]. For $d \geq 2$, the sum of Euclidean distances is a sum of square roots, and cannot be represented analytically in an efficient manner. Hence, we cannot efficiently compute a central trajectory for this measure.

Similarly, depending on the application, we may prefer a different way of integrating over time. Instead of the integral of D, we may, for example, wish to minimize $\max_t D_{\mathcal{T}}(t)$ or $\int D_{\mathcal{T}}^2(t) \, \mathrm{d}t$ (over all trajectoids $\mathcal{T}$). Again, the same general approach still works, but now, after constructing the Reeb graph, we compute the weights of each edge differently.

Minimizing the distance to the Ideal Trajectory I. We saw that for entities moving in $\mathbb{R}^1$, minimizing the distance from $\mathcal{C}$ to the farthest entity is identical to minimizing the

distance from $\mathcal{C}$ to the ideal trajectory I (which itself minimizes the distance to the farthest entity, but is not constrained to lie on an input trajectory). We also saw that for entities moving in $\mathbb{R}^d$, $d > 1$, these two problems are not the same. So, a natural question is whether we can also minimize the distance to I in this case. It turns out that, at least for $\mathbb{R}^2$, we can again use our general approach, albeit with different complexities.

Demaine et al. [14] show that for entities moving along lines[2] in $\mathbb{R}^2$ the ideal trajectory I has complexity $O(n^{3+\delta})$ for any $\delta > 0$. It follows that the function $\check{D}_\sigma(t) = \|I(t)\sigma(t)\|$ is a piecewise hyperbolic function with at most $O(\tau n^{3+\delta})$ pieces. The total number of intersections between all functions $\check{D}_\sigma$, for $\sigma \in \mathcal{X}$, is then $O(\tau n^{5+\delta})$. Similar to Lemma 16, we can then show that all lower envelopes in $\mathcal{R}$ together have complexity $O(\tau n^{4+\delta})$. We then also obtain an $O(\tau n^{4+\delta})$ bound on the complexity of a central trajectory $\mathcal{C}$ minimizing the distance to I.

To compute such a central trajectory $\mathcal{C}$ we again construct $\mathcal{R}$. To compute the weight of an edge $e \in R$ it is now more efficient to recompute the lower envelope $\mathcal{L}_e$ from scratch. This takes $O(\tau n^3 \cdot n \log n) = O(\tau n^4 \log n)$ time, whereas constructing the entire arrangement may take $O(\tau n^{5+\delta})$ time.

We note that the $O(n^{3+\delta})$ bound on the complexity of I by Demaine et al. [14] is not known to be tight. The best known lower bound is only $\Omega(n^2)$. So, a better upper bound for this problem immediately also gives a better bound on the complexity of $\mathcal{C}$.

Relaxing the input pieces requirement. We require each piece of the central trajectory to be part of one of the input trajectories, and allow small jumps between the trajectories. This is necessary, because in general there may be no intersections between the trajectories. Another interpretation of this dilemma is to relax the requirement that the output trajectory stays on an input trajectory at all times, and just require it to be *close* (within distance ε) to an input trajectory at all times. In this case, no discontinuities in the output trajectory are necessary.

We can model this by replacing each point entity by a disk of radius ε. The goal is then to compute a path that stays within the union of disks at all times, and minimizes $\mathcal{D}$. We now observe that if at time t the ideal trajectory I is contained in the same component of ε-disks as $\mathcal{C}$, the central trajectory will follow I. If I lies outside of the component, $\mathcal{C}$ travels on the border of the ε-disk (in the component containing $\mathcal{C}$) minimizing $D(t)$. In terms of the distance functions, this behavior again corresponds to following the lower envelope of a set of functions. We can thus identify the following types of break points of $\mathcal{C}$: (i) break points of I, (ii) breakpoints in one of the lower envelopes $\mathcal{L}_1, .., \mathcal{L}_m$ corresponding to the distance functions of the entities in each component, and (iii) break points at which $\mathcal{C}$ switches between following I and following a lower envelope $\mathcal{L}_j$. There are at most $O(\tau n^{3+\delta})$ break points of type (i) [14], and at most $O(\tau n^2 \sqrt{n})$ of type (ii). The break points of type (iii) correspond to intersections between I and the manifold that we get by tracing the ε-disks over the trajectory. The number of such intersections is at most $O(\tau n^{4+\delta})$. Hence, in this case $\mathcal{C}$ has complexity $O(\tau n^{4+\delta})$. We can thus get an $O(\tau n^{5+\delta} \log n)$ algorithm by computing the lower envelopes from scratch.

---

[2]Or, more generally, along a curve described by a low degree polynomial.

References

[1] P. K. Agarwal, M. de Berg, J. Gao, L. J. Guibas, and S. Har-Peled. Staying in the middle: Exact and approximate medians in $\mathbb{R}^1$ and $\mathbb{R}^2$ for moving points. In *CCCG*, pages 43–46, 2005.

[2] P. K. Agarwal, L. J. Guibas, J. Hershberger, and E. Veach. Maintaining the extent of a moving point set. *Discrete & Computational Geometry*, 26(3):353–374, 2001.

[3] P. K. Agarwal and S. Har-Peled. Maintaining approximate extent measures of moving points. In *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '01, pages 148–157, Philadelphia, PA, USA, 2001. Society for Industrial and Applied Mathematics.

[4] P. K. Agarwal and M. Sharir. Davenport-Schinzel sequences and their geometric applications. In J.-R. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*, pages 1–47. Elsevier Science Publishers B.V. North-Holland, Amsterdam, 2000.

[5] H.-K. Ahn, H. Alt, M. Buchin, E. Oh, L. Scharf, and C. Wenk. A middle curve based on discrete Fréchet distance. In E. Kranakis, G. Navarro, and E. Chávez, editors, *Proc. LATIN 2016: Theoretical Informatics: 12th Latin American Symposium*, pages 14–26. Springer, 2016.

[6] N. M. Amato, M. T. Goodrich, and E. A. Ramos. Computing the arrangement of curve segments: Divide-and-conquer algorithms via sampling. In *Proc. 11th Annual ACM-SIAM Sympos. on Discrete Algorithms*, volume 9, pages 705–706, 2000.

[7] R. Basu, B. Bhattacharya, and T. Talukdar. The projection median of a set of points in $\mathbb{R}^d$. *Discrete & Computational Geometry*, 47(2):329–346, 2012.

[8] M. Bern, D. Eppstein, P. Plassman, and F. Yao. Horizon theorems for lines and polygons. In J. Goodman, R. Pollack, and W. Steiger, editors, *Discrete and Computational Geometry: Papers from the DIMACS Special Year*, volume 6 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 45–66. American Mathematical Society, Association for Computing Machinery, Providence, RI, 1991.

[9] P. Bovet and S. Benhamou. Spatial analysis of animals' movements using a correlated random walk model. *J. Theoretical Biology*, 131(4):419–433, 1988.

[10] K. Buchin, M. Buchin, J. Gudmundsson, M. Löffler, and J. Luo. Detecting commuting patterns by clustering subtrajectories. *International Journal of Computational Geometry & Applications*, 21(03):253–282, 2011.

[11] K. Buchin, M. Buchin, M. van Kreveld, M. Löffler, R. I. Silveira, C. Wenk, and L. Wiratma. Median trajectories. *Algorithmica*, 66(3):595–614, 2013.

[12] K. Buchin, M. Buchin, M. van Kreveld, B. Speckmann, and F. Staals. Trajectory grouping structure. *Journal of Computational Geometry*, 6(1), 2015.

[13] C. Calenge, S. Dray, and M. Royer-Carenzi. The concept of animals' trajectories from a data analysis perspective. *Ecological Informatics*, 4(1):34 – 41, 2009.

[14] E. D. Demaine, S. Eisenstat, L. J. Guibas, and A. Schulz. Kinetic minimum spanning circle. In *Proceedings of the Fall Workshop on Computational Geometry*, 2010.

[15] T. K. Dey. Improved bounds on planar k-sets and k-levels. In *Proceedings 38th Annual Symposium on Foundations of Computer Science*, pages 156–161, 1997.

[16] S. Dodge, R. Weibel, and E. Forootan. Revealing the physics of movement: Comparing the similarity of movement characteristics of different types of moving objects. *Computers, Environment and Urban Systems*, 33(6):419–434, 2009.

[17] S. Durocher and D. Kirkpatrick. The Steiner centre of a set of points: Stability, eccentricity, and applications to mobile facility location. *International Journal of Computational Geometry & Applications*, 16(04):345–371, 2006.

[18] S. Durocher and D. Kirkpatrick. The projection median of a set of points. *Computational Geometry*, 42(5):364 – 375, 2009.

[19] S. Gaffney, A. Robertson, P. Smyth, S. Camargo, and M. Ghil. Probabilistic clustering of extratropical cyclones using regression mixture models. *Climate Dynamics*, 29(4):423–440, 2007.

[20] S. Gaffney and P. Smyth. Trajectory clustering with mixtures of regression models. In *Proc. 5th ACM SIGKDD Internat. Conf. Knowledge Discovery and Data Mining*, pages 63–72, 1999.

[21] E. Gurarie, R. D. Andrews, and K. L. Laidre. A novel method for identifying behavioural changes in animal movement data. *Ecology Letters*, 12(5):395–408, 2009.

[22] S. Har-Peled and B. Raichel. The Fréchet distance revisited and extended. In *Proc. of the Twenty-seventh Annual Symposium on Computational Geometry*, SoCG '11, pages 448–457, New York, NY, USA, 2011. ACM.

[23] H. Imai and M. Iri. Polygonal approximations of a curve âĂŞ formulations and algorithms. In *Computational Morphology*, pages 71–86. Elsevier Science, 1988.

[24] J. Lee, J. Han, and K. Whang. Trajectory clustering: a partition-and-group framework. In *Proc. ACM SIGMOD International Conference on Management of Data*, pages 593–604, 2007.

[25] X. Li, X. Li, D. Tang, and X. Xu. Deriving features of traffic flow around an intersection from trajectories of vehicles. In *Proc. 18th International Conference on Geoinformatics*, pages 1–5. IEEE, 2010.

[26] J. Pach and P. K. Agarwal. *Combinatorial Geometry*. John Wiley & Sons, New York, NY, 1995.

[27] B. Shneiderman. The eyes have it: a task by data type taxonomy for information visualizations. pages 336–343, 1996.

[28] A. Stohl. Computation, accuracy and applications of trajectories – a review and bibliography. *Atmospheric Environment*, 32(6):947 – 966, 1998.

[29] M. Vlachos, D. Gunopulos, and G. Kollios. Discovering similar multidimensional trajectories. In *Proc. 18th Internat. Conf. Data Engineering*, pages 673–684, 2002.