Minimizing Walking Length in Map Matching

Amin Gheibi, Anil Maheshwari, Jörg-Rüdiger Sack

Minimizing Walking Length in Map Matching

Amin Gheibi*, Anil Maheshwari*, and Jörg-Rüdiger Sack *

School of Computer Science, Carleton University, Ottawa, ON, Canada
[agheibi,anil,sack]@scs.carleton.ca

Abstract. In this paper, we propose a geometric algorithm for a map matching problem. More specifically, we are given a planar graph, H, with a straight-line embedding in a plane, a directed polygonal curve, T, and a distance value $\varepsilon > 0$. The task is to find a path, P, in H, and a parameterization of T, that minimize the sum of the length of walks on T and P whereby the distance between the entities moving along P and T is at most ε, at any time during the walks. It is allowed to walk forwards and backwards on T and edges of H. We propose an algorithm with $\mathcal{O}(mn(m+n)\log(mn))$ time complexity and $\mathcal{O}(mn(m+n))$ space complexity, where m (n, respectively) is the number of edges of H (of T, respectively). As we show, the algorithm can be generalized to work also for weighted non-planar graphs within the same time and space complexities.

1 Introduction

Trajectory data are often obtained from global positioning system (GPS) devices. Such devices have accuracy limitations due to noise, sampling intervals, or poor signals (e.g., inside buildings) thus raw spatial trajectories tend not to be accurate. Under the assumption that the travel captured by the trajectory was following edges of a map (stored as a graph) the map matching problem arises. It asks to find a path on the map that "corresponds well" to the given trajectory. Map matching arises in different contexts and is a necessary step in preprocessing raw data before data mining [1]. A variety of approaches have been used to solve the map matching problem (e.g. geometric, probabilistic methods, fuzzy logic, neural networks). In [2], Chen et al. discussed recent map matching algorithms when a trajectory is obtained from low-frequency GPS data of vehicles driving on a road network. Ruan et al. [3] studied indoor map matching technology based on personal motion states. In [4], Asakura et al. proposed a pedestrian-oriented map matching algorithm in the context of disasters. In this context, refugees have battery-driven mobile GPS terminals and move to shelters at walking speed. They stated that in order to reduce battery consumption (which is vital in this context), they chose a geometric approach in which computation resources are less utilized when compared e.g., with probabilistic methods.

* Research supported by Natural Sciences and Engineering Research Council of Canada

In this paper, we focus on geometric approaches. We assume that a map is given as a planar graph via a straight-line embedding in a plane. Therefore, a path in the graph corresponds to a polygonal curve in the plane. A trajectory is given as a directed polygonal curve from a starting point to an ending point. The objective is to find a path in the map which is most similar to the given trajectory.

To measure similarity, [1] and [7] observe that methods which consider global features of the input trajectories achieve more accurate results than local approaches. The Fréchet distance is a global similarity measure between curves, see e.g., the seminal paper [6]. Commonly, the Fréchet distance is illustrated as follows: Suppose a person wants to walk along one curve and his/her dog on another; the person is keeping the dog at a leash. Both person and dog walk, from starting point to ending points along their respective curves. The standard Fréchet distance is the minimum leash length required without either person or dog needing to backtrack. The weak Fréchet distance is a variant of the standard Fréchet distance in which backtracking on one or both curves is allowed. Alt and Godau [6] proposed algorithms to compute the standard and weak Fréchet distances in $\mathcal{O}(n^2 \log n)$ time, where n is the maximum number of segments in the input polygonal curves. Har-Peled and Raichel [15] showed that the weak Fréchet distance can be computed in quadratic time.

In [5], Alt et al. discussed the map matching problem set in the context of the standard Fréchet distance. I.e., their algorithm finds a path in the planar graph with minimum Fréchet distance to the given trajectory. The time complexity of their algorithm is $\mathcal{O}(mn \log^2 mn)$ where m (n, respectively) is the number of edges in the input planar graph (the input polygonal curve, respectively). Brakatsoulas et al. [7], extended the map matching algorithm of [5] for the weak Fréchet distance. The time complexity of their algorithm is $\mathcal{O}(mn \log mn)$. In [8], Chen et al. proposed a $(1 + \varepsilon)$-approximation algorithm for the map matching problem when the similarity measure is the standard Fréchet distance and input model is more "realistic". They assumed that the input polygonal curve is c-packed and the input graph is ϕ-low density in $\mathbb{R}^d$ (see Section 2 of [8]).

In [10], Gheibi et al. studied a natural optimization problem on the weak Fréchet distance, called the *minimum backward Fréchet distance (MBFD)* problem. There, the task is to determine a pair of walks for a given input leash length such that the total length of backtracking on both input polygonal curves is minimized. The cost of backtracking could represent, for example, the cost of moving against a flow, or the cost for a moving entity (e.g., a human, a humanoid robot) to move backwards because of the entity's physiology [9]. They proposed an algorithm solving this problem within time complexity $\mathcal{O}(n^2 \log n)$ and space complexity $\mathcal{O}(n^2)$, where n is the maximum number of segments in the polygonal curves. In [11], the weighted variant of the MBFD problem is solved in $\mathcal{O}(n^3)$ time. In this variant, each edge of the input polygonal curves has an associated non-negative weight to capture different costs for backward movement.

In this paper, we study the map matching problem when the similarity measure is the MBFD. More specifically, as input, we are given: a planar graph, H,

with a straight-line embedding in a plane, a directed polygonal curve, T, and a distance $\varepsilon > 0$. As motions, both forward and backward motions along T and the edges of H are allowed. The objective is to find a path, P, in H, and a parameterization of T, that minimize the sum of the walk lengths along T and P while keeping a leash length of at most ε. We restrict the start and end point of P to be at a vertex of H. However, P may partially contain an edge of H. The difference between this problem setting and that optimization setting of [5] and [7], is that here the total walking length along T and P is minimized while in the other settings the leash length is minimized. The optimization problem introduced in this paper, can also be used to track objects moving on road networks. To ensure high-quality tracking, the mobile tracker must remain within a distance of ε to the moving object, at all time. To minimize energy consumption, the tracker wants to minimizes walking distance. This type of scenario has been discussed in the context of wireless networks (see [12] and [13]).

Figure 1 shows an example of an embedding of a planar graph, H, in $\mathbb{R}^2$, a polygonal curve, T, and a length ε. The dog walks on T from $T(0)$ to $T(4)$ and the person chooses a path in H, from one vertex of H to another vertex. Two points, a and b, are determined on $\langle v_4, v_5\rangle$ and $\langle v_3, v_4\rangle$ respectively. A path in H, and a walk on T, that minimize the walking lengths on H and T, are as follows: the dog starts at $T(0)$ and continues on T. The person starts at v_1 and walks on the edges $\langle v_1, v_3\rangle$, $\langle v_3, v_4\rangle$, and $\langle v_4, v_5\rangle$. They move together in a forward direction until the dog reaches the end of the second segment of T and the person reaches the point a on $\langle v_4, v_5\rangle$. Then, the dog continues to move forwards until the end of the third segment of T is reached, while the person moves backwards from a to v_4 and then to b. At the final step, they move forwards again together until the dog reaches the end of T and the person reaches v_5. We show the path in the graph by the sequence of its vertices, $P^* = [v_1, v_3, v_4, a, v_4, b, v_4, v_5]$.

The structure of this paper is as follows. In Section 2, we discuss preliminaries and define the problem formally. In Section 3, we propose a polynomial time algorithm for the map matching problem introduced. Then, in Section 4, we develop an algorithm with improved time and space complexities. In Section 5, we sketch a solution to a weighted problem variant. Finally, in Section 6, we conclude the paper.

2 Preliminaries and Definitions

A geometric path in $\mathbb{R}^2$ is a sequence of points in 2D Euclidean space, $\mathbb{R}^2$. A polygonal curve, or a discrete geometric path, is a geometric path, sampled by a finite sequence of points (called vertices), which are connected by line segments (called edges) in order. Let $T : [0, n] \to \mathbb{R}^2$ be a polygonal curve with n segments. A vertex of T is denoted by $T(i)$, $i = 0, \ldots, n$. Let $H = \langle V_H, E_H\rangle$ be a planar graph with a straight-line embedding in $\mathbb{R}^2$ where V_H (E_H, respectively) is the set of vertices (edges, respectively) of H. In this paper, the geometric embedding of H is crucial and we simply refer to the straight-line embedding of the graph

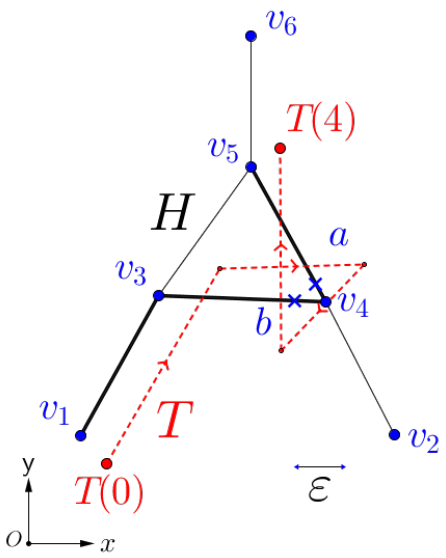

Fig. 1. An embedding of a planar graph, H, a polygonal curve, T, and a length ε are given. The path $P^* = [v_1, v_3, v_4, a, v_4, b, v_4, v_5]$, in H, is a part of a solution to the map matching problem instance. The edges of H that P^* lies on, are illustrated in bold.

in $\mathbb{R}^2$ as H. A path, P, in H, is a polygonal curve $P : [0, 1] \to H$, such that $P \subset H$ and $P(0), P(1) \in V_H$.

A parameterization of a polygonal curve, $T : [0, n] \to \mathbb{R}^2$, is a continuous function $f : [0, 1] \to [0, n]$, where $f(0) = 0$ and $f(1) = n$. Note that a parameterization is corresponding to a walk on T and the interval $[0, 1]$ is representing time during the walk.

Walking Length. Let f be a parameterization of a polygonal curve, T. Let $\mathcal{D}_f \subseteq [0, 1]$ be the closure of the set of times in which $f(t)$ is decreasing (i.e., the movement is backward). The walking length of T is defined by Formula 1, where $\|.\|$ is the Euclidean norm and $(.)'$ is derivative.

$$\mathcal{L}_f(T) := ||T|| + 2 \int_{t \in \mathcal{D}_f} ||(T(f(t)))'|| dt \tag{1}$$

Note that if f is monotone (i.e., there is no backward movements on T), then $\mathcal{L}_f(T) = ||T||$.

Problem Definition. Suppose H, T and a length, $\varepsilon > 0$, are given. The objective is to find a path in H and a parameterization of T such that sum of the length of P and the walking length of T is minimized (Formula 2). We consider only paths in the graph and parameterizations of T that guarantee to maintain the leash length at most ε, during the walks.

$$\mathcal{M}^{\varepsilon}(H,T) := \inf_{P \subset H, f} \{||P|| + \mathcal{L}_f(T)\} \tag{2}$$

Deformed free-space surface. The free-space diagram is a structure, used to decide whether the Fréchet distance between two polygonal curves is upper bounded by a given ε [6]. In [5], Alt et al. introduced a 3D structure, called free-space surface, to solve the decision version of their map matching problem. Here, we use both free-space diagram and free-space surface. However, we modify them slightly, to fit our problem setting.

Let $P : [0,1] \to H$ be a path in H with $k+1$ vertices, $[p_0, p_1, \ldots, p_k]$. The *free-space diagram* is the rectangle $[0,1] \times [0,1]$, partitioned into n columns and k rows. It consists of nk parameter cells $C^{x,y}$, for $x = 1, ..., n$ and $y = 1, ..., k$. Cell $C^{x,y}$ is the result of the product of two sub-intervals of $[0,1]$ that are mapped to edge $\overrightarrow{T(x-1)T(x)}$ of T and edge $\overrightarrow{p_{y-1}p_y}$ of P, respectively. We call a point $(t_1, t_2) \in [0,1]^2$ white if $d(T(f(t_2)), P(t_1)) \leq \varepsilon$, where d is the Euclidean distance; otherwise, we call it black. It has been shown that the set of all white points inside a cell $C^{x,y}$ is determined by the intersection of an ellipse with $C^{x,y}$. This set is called the free-space region of that cell. The boundaries of a cell and its corresponding ellipse intersect at most eight times. These intersection points form at most four intervals of white points on the boundary of the cell (i.e., at most one interval per side of the cell). Note that two adjacent cells have the same interval on the shared side between the cells. The union of all cells' free-spaces is the free-space (or white-space) of the diagram; it is denoted by W_P. The complement of W_P is the forbidden-space (or black-space) of the diagram and is denoted by B_P. We stretch/compress the columns and rows of the free-space diagram, such that their widths and heights are equal to the lengths of the corresponding segments of T and P, respectively. The resulting diagram is called the *deformed free-space diagram* and is denoted by $\mathcal{F}_{\varepsilon}(T,P)$. In Figure 2, the free-space diagram $\mathcal{F}_{\varepsilon}(T,P^*)$ is drawn, where $P^* = [v_1, v_3, v_4, a, v_4, b, v_4, v_5]$ is a path in H, denoted as a sequence of its vertices.

Note that if the path P contains only a single vertex of H, $v_i \in V_H$, then $\mathcal{F}_{\varepsilon}(T,P)$ is a line segment and its length is equal to the Euclidean length of T. We call this 1D free-space diagram, $\mathcal{F}_i$, the *deformed free-space line* of v_i. We denote the left endpoint of $\mathcal{F}_i$ (i.e., the endpoint corresponding to $T(0)$) by s_i and the right endpoint of $\mathcal{F}_i$ (i.e., the endpoint corresponding to $T(n)$) by t_i. If P contains only an edge, $\langle v_i, v_j \rangle \in E_H$, of H, then $\mathcal{F}_{\varepsilon}(T,P)$ has only one row. We call this row the *deformed free-space face* of $\langle v_i, v_j \rangle$, and denote it by $\mathcal{F}_i^j$.

Note that $\mathcal{F}_i^j$ and $\mathcal{F}_j^k$ have $\mathcal{F}_j$ in common. Therefore, gluing $\mathcal{F}_i^j$ and $\mathcal{F}_j^k$ along $\mathcal{F}_j$ produces a conforming surface. Thus, we can construct the *deformed free-space surface* as follows. We first lay out the straight-line embedding of H in the xy-plane. For each edge $\langle v_i, v_j \rangle \in E_H$, we lay out $\mathcal{F}_i^j$, orthogonal to the xy-plane, along z axis, such that $\mathcal{F}_i$ ($\mathcal{F}_j$, respectively) is on top of v_i (v_j, respectively) and s_i (s_j, respectively) is in the xy-plane. Note that $\mathcal{F}_i^j$ is stretched along z axis from the plane $z = 0$ to the plane $z = \|T\|$. Suppose $Adj(v_j)$ is the set of all vertices $v_k \in V_H$ such that $\langle v_j, v_k \rangle \in E_H$. We glue $\mathcal{F}_i^j$ to $\mathcal{F}_j^k$ along $\mathcal{F}_j$, where

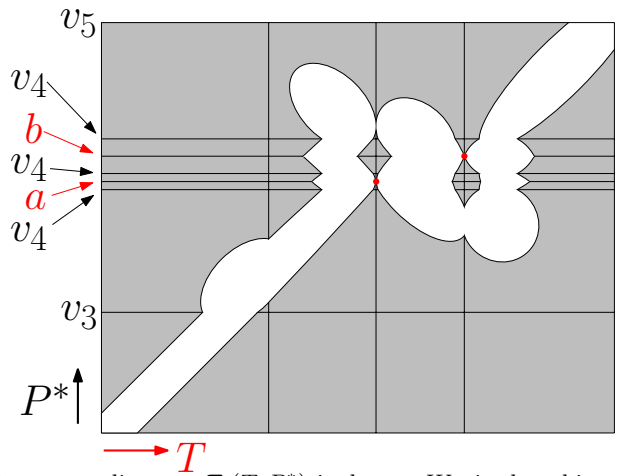

Fig. 2. The free-space diagram $\mathcal{F}_\varepsilon(T, P^*)$ is drawn. W_P is the white area and B_P is the gray area.

$v_k \in Adj(v_j)$. Also, we glue $\mathcal{F}_i^j$ to $\mathcal{F}_h^i$ along $\mathcal{F}_i$, where $v_i \in Adj(v_h)$. The result is a conforming $3D$ surface between two planes, $z = 0$ and $z = \|T\|$, called deformed free-space surface and is denoted by $\mathcal{S} = H \times [0, \|T\|]$. Note that s_i is on the plane $z = 0$ and t_i is on the plane $z = \|T\|$, $i = 1, \ldots, |V_H|$. The union of the white-space (black-space, respectively) of all faces of $\mathcal{S}$ is called the *white-surface* (*black-surface*, respectively) and is denoted by $\mathcal{W}$ ($\mathcal{B}$, respectively). For the given planar graph H, the polygonal curve T, and the length ε in Figure 1, the corresponding deformed free-space surface is shown in Figure 3, from two points of views. Since the white-space of each cell of any $\mathcal{F}_i^j$ is convex, for simplicity, we just draw the white-space intervals on the boundary of the cells. In this figure, the red dashed polygonal curve is a path on the white-surface $\mathcal{W}$, from s_1 to t_5, that realizes $P^* = [v_1, v_3, v_4, a, v_4, b, v_4, v_5]$, in H, and a parameterization of T, that is an optimal solution to our problem setting. It intersects the following free-space faces sequentially: $\mathcal{F}_1^3$, $\mathcal{F}_3^4$, $\mathcal{F}_4^5$, $\mathcal{F}_3^4$, $\mathcal{F}_4^5$.

3 Algorithm

In this section, we first transform the map matching problem to a shortest path problem on a weighted graph, $\mathcal{G} = \langle V, E \rangle$; this yields a polynomial time algorithm. Before discussing the construction of $\mathcal{G}$, we introduce a set of Steiner points on the boundary of the cells of $\mathcal{S}$.

Steiner Points. We position Steiner points so as to create intervals on the boundary of the cells of $\mathcal{S}$. There are two types of intervals, Type 1 and Type 2. We classify the Steiner points based on the type of the intervals that they belong to. We denote the set of Type 1 (Type 2, respectively) Steiner points by S_1 (S_2, respectively).

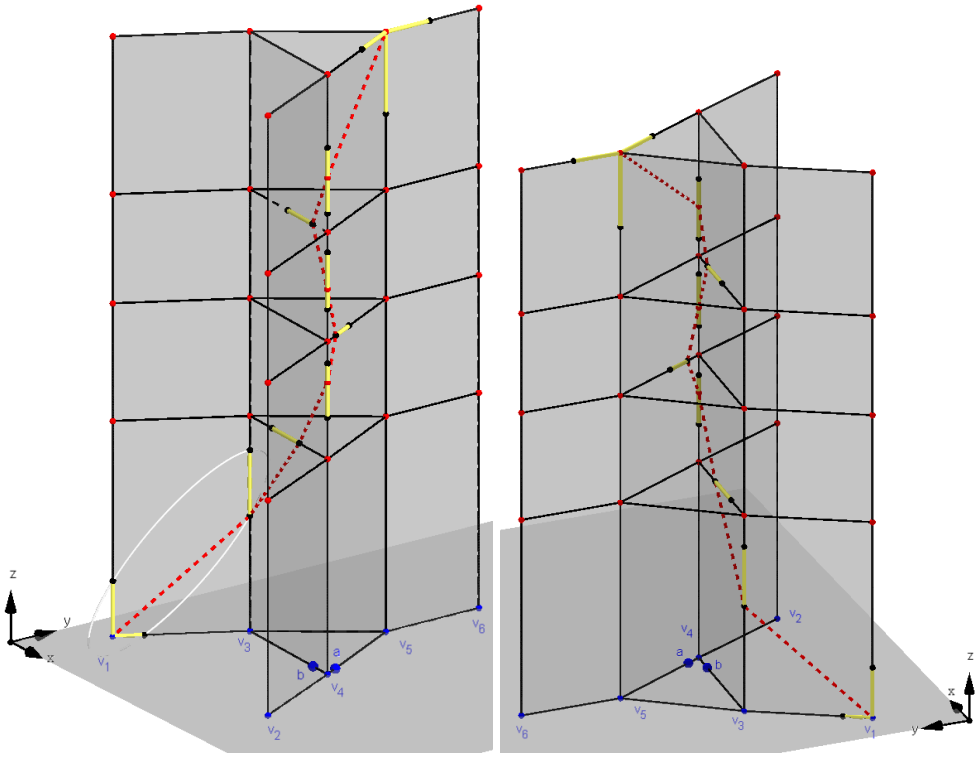

Fig. 3. The free-space surface for the example of Figure 1 is drawn from two different viewpoints in $3D$. The yellow line segments show the intervals on the cell boundaries. The red dashed polygonal curve is a path on the white-surface that realizes an optimal solution to our problem setting.

Type 1. We say an interval is Type 1, if it lies completely in a plane, $z = c$, parallel to the xy-plane, where c is a constant. Each deformed free-space face, $\mathcal{F}_i^j$, may have $n+1$ Type 1 intervals, $\mathcal{FI}_i^j(\ell)$, $\ell = 0, \ldots, n$, shared between its cells (where n is the number of edges in T). For each interval $\mathcal{FI}_i^j(\ell)$, we project the endpoints of $\mathcal{FI}_i^j(\ell)$ orthogonally to all $\mathcal{FI}_i^j(k)$, $k \neq \ell$. If the line segment from an endpoint of $\mathcal{FI}_i^j(\ell)$ to its projection on $\mathcal{FI}_i^j(k)$ lies in the free-space of $\mathcal{F}_i^j$ and the projection point is not identical with an endpoint of $\mathcal{FI}_i^j(k)$, then we take the projection point as a Type 1 Steiner point (see Figure 4). The set of all Steiner points, obtained by the projections on $\mathcal{F}_i^j$, for all $\langle v_i, v_j \rangle \in E_H$, is denoted by S_1.

Type 2. We say an interval is Type 2 if it lies completely on a deformed free-space line. As we mentioned in Section 2, a plane $z = c$ corresponds to a point on the given trajectory T. The intersection of $z = c$ and $\mathcal{S}$ is an instance of H, denoted by H_c. Note that some part (possibly empty) of H_c is in $\mathcal{W}$. Let $z = h_j$ be the corresponding plane of $T(j)$, a vertex of T. The part of the deformed

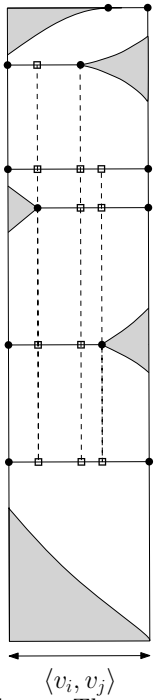

Fig. 4. The free-space face $\mathcal{F}_i^j$ is drawn. The endpoints of the intervals, $\mathcal{FI}_i^j(\ell)$, are shown by points and the Type 1 Steiner points are shown by squares.

free-space surface, $\mathcal{S}$, between the two parallel planes, $z = h_{j-1}$ and $z = h_j$, $j = 1, \ldots, n$, corresponds to edge $\overrightarrow{T(j-1)T(j)}$ of T. We denote this part of $\mathcal{S}$ by $\mathcal{T}_{j-1}^j = H \times [h_{j-1}h_j]$. In $\mathcal{T}_{j-1}^j$, $j = 1, \ldots, n$, there is at most one Type 2 interval per vertex $v_i \in V_H$. We denote these Type 2 intervals by $\mathcal{TI}_{j-1}^j(i)$, $i = 1, \ldots, |V_H|$. Suppose $z = c$ is the plane that is passing through an endpoint, p, of $\mathcal{TI}_{j-1}^j(i)$. Let the intersection of $z = c$ with $\mathcal{TI}_{j-1}^j(k)$, $k \neq i$, be q_k. Note that both p and q_k are on the graph H_c. Then, if q_k is not an endpoint of $\mathcal{TI}_{j-1}^j(k)$ and there is a path, from p to q_k, in H_c, that is in $\mathcal{W}$, then q_k is a Type 2 Steiner Point. The set of all Type 2 Steiner points is denoted by S_2. An example is given in Figure 5. Suppose it is $\mathcal{T}_{j-1}^j$, for $j = 1$. In this example, there are four yellow intervals, $\mathcal{TI}_{j-1}^j(1)$, $\mathcal{TI}_{j-1}^j(3)$, $\mathcal{TI}_{j-1}^j(5)$, and $\mathcal{TI}_{j-1}^j(6)$. The black points show the interval endpoints and red points show the Type 2 Steiner points. For simplicity, only two, out of eight planes, are drawn. The plane z_3 (z_6, respectively) is passing through an endpoint of $\mathcal{TI}_{j-1}^j(3)$ ($\mathcal{TI}_{j-1}^j(6)$, respectively). The intersections of z_3 with $\mathcal{TI}_{j-1}^j(5)$ and $\mathcal{TI}_{j-1}^j(6)$ are Steiner points. However, the intersection of z_6 with $\mathcal{TI}_{j-1}^j(3)$ is not a Steiner point.

Constructing Graph. Now, we explain the construction of $\mathcal{G} = \langle V, E \rangle$. Recall that the white-surface (the white-space of $\mathcal{S}$) is denoted by $\mathcal{W}$. The vertices of $\mathcal{W}$ are the end points of the intervals on the boundary of the cells in $\mathcal{S}$ (at most

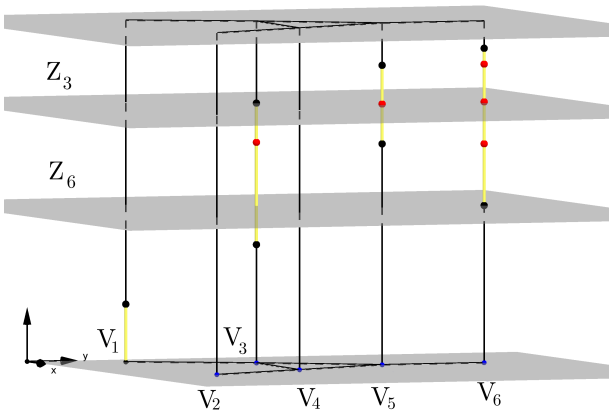

Fig. 5. An example of $\mathcal{T}^j_{j-1}$, for $j = 1$, is drawn. In this example, there are four intervals that are shown by yellow color. The black balls show the interval endpoints and red balls show the Type 2 Steiner points.

4 intervals may exist per cell). We denote the set of vertices of $\mathcal{W}$ by $V_{\mathcal{W}}$. The set of vertices, V, of $\mathcal{G}$, is $V = V_{\mathcal{W}} \cup S_1 \cup S_2$. Note that V contains all s_i and t_i if they are in $\mathcal{W}$. Every two vertices, $v_1, v_2 \in V$, that are on the boundary of a cell, are linked by two directed edges in E, from v_1 to v_2, $\langle v_1, v_2\rangle$, and vice versa, $\langle v_2, v_1\rangle$. The weight of an edge $e = \langle v_1, v_2\rangle \in E$, is its length in the L_1 metric, $|e|_1$.

Obtain an Optimal Solution. In order to have an optimal solution, at least one s_i and one t_j, $i, j = 1, \ldots, |V_H|$, must be in $\mathcal{W}$. The main steps of the algorithm are as follows:

- Find all the vertices in V_H that are in ε distance of $T(0)$ ($T(n)$, respectively), $v_{i_1}, \ldots, v_{i_{k_1}}$ ($v_{j_1}, \ldots, v_{j_{k_2}}$, respectively).
- Add an extra node, s', to $\mathcal{G}$, and add k_1 extra directed edges, $\langle s', s_{i_1}\rangle, \ldots, \langle s', s_{i_{k_1}}\rangle$, to E. The weight of these k_1 edges are set to zero. Analogously, add another extra node, t', to $\mathcal{G}$, and add k_2 extra directed edges, $\langle t_{j_1}, t'\rangle, \ldots, \langle t_{j_{k_2}}, t'\rangle$, to E. The weight of these k_2 edges are also set to zero.
- Find a shortest path, from s' to t', in $\mathcal{G}$. Note that if there is no path from s' to t' in $\mathcal{G}$, then there is no solution for the given leash length.
- Remove s' and t' from the head and tail of the shortest path. The remaining path is from one s_i to one t_j. It gives an optimal solution to our problem setting.

Note that, a vertex of $\mathcal{G}$ (except s' and t') is also represented by a point in $\mathcal{W}$. Therefore, the geometric embedding of a path, from one s_i to one t_j, in $\mathcal{G}$, is constructed by connecting the consecutive vertices of the path in $\mathcal{W}$ by line segments.

Observation 1 *Let Π be a path in the white-space, $\mathcal{W}$, of a deformed free-space surface, $\mathcal{S}$, from one s_i to one t_j. Π realizes a path, $P : [0, 1] \to H$, in H, and a*

parameterization, $f : [0, 1] \to [0, n]$, of T, that maintain the leash length at most ε, for all $t \in [0, 1]$.

Constructing a path in H. We can construct a path P in H, from the given path Π in $\mathcal{W}$, as follows. As we mentioned earlier in this section, we have two types of intervals on the boundary of the cells, Type 1 and Type 2. A Type 1 interval lies completely on a plane, $z = c$, parallel to the xy-plane. A Type 2 interval lies completely on a deformed free-space line, $\mathcal{F}_i$. The path Π intersects a sequence of intervals (of both types). The path P in H is constructed by processing the intervals in this sequence. For each interval in this sequence, if it is Type 2 interval, on $\mathcal{F}_i$, then we append v_i to the tail of P. If it is Type 1, then the intersection point, q, of Π and that interval, is appended to the tail of P, as a vertex of P. Note that q may not be a vertex of H. However, it is a point on an edge of H. At the end, we connect the consecutive vertices in P by straight line segments.

Correctness. To establish the correctness, we use norms in two spaces: (1) the Euclidean space of the embedding of the input graph and the polygonal curve, called the input space, (2) the deformed free-space surface, called the configuration space. In the input space, we denote the Euclidean length of a polygonal curve T by $\|T\|$. We also defined walking length of T, $\mathcal{L}_f(T)$, based on a parameterization f. Note that if f is a monotone parameterization, then $\mathcal{L}_f(T) = \|T\|$. In configuration space, a path from an s_i to a t_j in $\mathcal{W}$, is also denoted by its vertices, $\Pi : \langle s_i = p_1, p_2, \ldots, p_k = t_j \rangle$. The length, $|.|_1$, of each segment of Π is calculated by the L_1 metric. The length of a path, $|\Pi|_1$, is the sum of the length of its segments.

Lemma 1 is at the heart of the correctness proof. This section is concluded by a corollary to Lemma 1 and Observation 1, that is, in order to find a solution for our problem setting, it suffices to find a shortest path from s' to t' in $\mathcal{G}$.

Lemma 1 *For any path $\Pi : \langle s_i = p_1, p_2, \ldots, p_{k_1} = t_j \rangle$ in $\mathcal{W}$, there is a path $\Pi' : \langle s_i = p'_1, p'_2, \ldots, p'_{k_2} = t_j \rangle$ in $\mathcal{W}$ such that $\Pi' \subset \mathcal{G}$ and $|\Pi'|_1 \leq |\Pi|_1$.*

Proof. The path Π intersects a sequence, SF, of deformed free-space faces. Every two consecutive faces in SF share a deformed free-space line. Therefore, we can unfold the free-space faces in the sequence, along the shared free-space lines. The result is a $2D$ free-space diagram, denoted by $\mathcal{F}_\varepsilon(SF)$. W.l.o.g., we can assume that $\mathcal{F}_\varepsilon(SF)$ is axis aligned in $\mathbb{R}^2$. The path Π is also unfolded into a $2D$ path in the white-space, $\mathcal{W}_{SF}$, of $\mathcal{F}_\varepsilon(SF)$. Note that unfolding does not change the length of a path. As an example, in Figure 6a, the result of unfolding the faces that are intersected by the red dashed polygonal curve in Figure 3, is shown.

Let $\Pi_{opt} : \langle s_i = q_1, q_2, \ldots, q_{k_3} = t_j \rangle$ be a L_1 shortest path, from s_i to t_j, in $\mathcal{W}_{SF}$. Then, $|\Pi_{opt}|_1 \leq |\Pi|_1$. To prove the lemma, it suffices to show that there is a path $\Pi' \subset \mathcal{G} = \langle V, E \rangle$, from s_i to t_j, in $\mathcal{W}_{SF}$, such that $|\Pi'|_1 = |\Pi_{opt}|_1$.

We know that the vertices of Π_{opt} are endpoints of some intervals on the boundary of the cells of $\mathcal{F}_\varepsilon(SF)$ [14] (the well known rubber band property of shortest paths). Therefore, the vertices of Π_{opt} are in V (i.e., the set of vertices

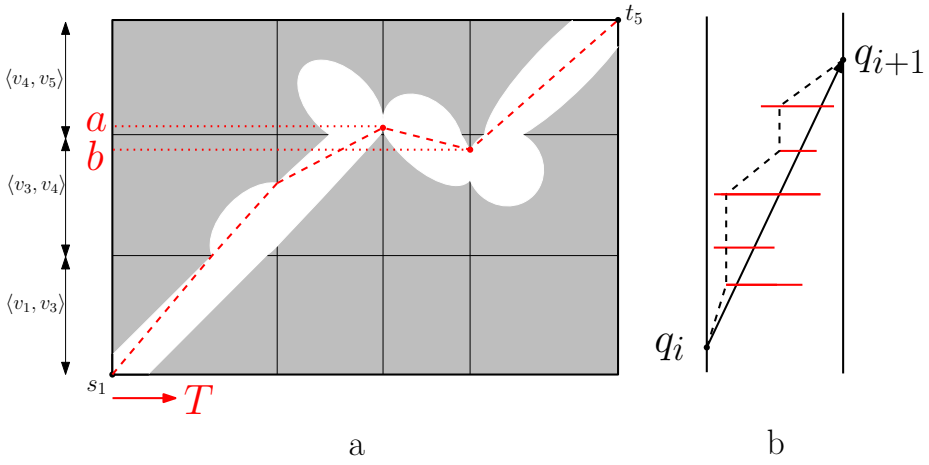

Fig. 6. a) The result of unfolding the sequence of deformed free-space faces that are intersected by the red dashed polygonal curve in Figure 3. It is a $2D$ free-space diagram, $\mathcal{F}_\varepsilon(SF)$. The red dashed polygonal curve is shown after unfolding. b) Illustration of case 1 in the proof of Lemma 1.

of $\mathcal{G}$). Thus, it is sufficient to show that for each edge, $\overrightarrow{q_iq_{i+1}}$, of Π_{opt}, there is a path, $\pi_{q_iq_{i+1}}$, from q_i to q_{i+1}, in $\mathcal{G}$, that $|\overrightarrow{q_iq_{i+1}}|_1 = |\pi_{q_iq_{i+1}}|_1$. Two cases arise depending on whether $\overrightarrow{q_iq_{i+1}}$ lies completely within a row (or a column) of $\mathcal{F}_\varepsilon(SF)$, or not. We need a definition before discussing these cases. We assume that $\mathcal{F}_\varepsilon(SF)$ is an axis-aligned rectangle in a $2D$ Cartesian coordinate system, where the x-axis corresponds to T (Figure 6a). We say a path $\Pi \in \mathcal{W}_{SF}$ is x-monotone (y-monotone, respectively), if any vertical (horizontal, respectively) line intersects it at most ones. Π is said to be xy-monotone, if it is both x- and y-monotone.

Case 1. In this case, $\overrightarrow{q_iq_{i+1}}$ lies completely within a column (or a row) of $\mathcal{F}_\varepsilon(SF)$. Here, we discuss the case when it lies within a column (see Figure 6b); the arguments are analogous for case of a row. W.l.o.g we assume that $\overrightarrow{q_iq_{i+1}}$ is xy-increasing. The other cases are symmetric. Edge $\overrightarrow{q_iq_{i+1}}$ intersects a sequence of horizontal intervals, I_z, within a column. They are sorted based on their y coordinates. We construct $\pi_{q_iq_{i+1}}$ sequentially and always denote the last vertex appended to $\pi_{q_iq_{i+1}}$ by π_{last}. Initially, $\pi_{q_iq_{i+1}}$ contains only q_i and $\pi_{last} = q_i$. The sequence of intervals are processed sequentially. Suppose we processed interval I_z and now we want to process I_{z+1}. We project orthogonally from π_{last} to I_{z+1}. If the projection point exists (i.e., the perpendicular line from π_{last} to I_{z+1} intersects I_{z+1}), then append the projection point on I_{z+1} to $\pi_{q_iq_{i+1}}$ and update π_{last}. Otherwise, the closest endpoint of I_{z+1} to π_{last} is appended to $\pi_{q_iq_{i+1}}$ and we update π_{last}. When all intervals, I_z, have been processed, q_{i+1} is appended to $\pi_{q_iq_{i+1}}$.

Since the sorted list of intervals withing a column are traversed by $\pi_{q_iq_{i+1}}$ sequentially, the path $\pi_{q_iq_{i+1}}$ is y-monotone. Also, by construction, each vertex

of $\pi_{q_iq_{i+1}}$ either has the same x as its preceding vertex in $\pi_{q_iq_{i+1}}$ (i.e., it is the result of the orthogonal projection) or its x is greater than its preceding vertex's x (since the orthogonal projection does not exist and $\overrightarrow{q_iq_{i+1}}$ is xy-increasing inside the white-space). Therefore, the path $\pi_{q_iq_{i+1}}$ is x-monotone. Thus, the path $\pi_{q_iq_{i+1}}$ is xy-monotone. We know that the L_1 length of two xy-monotone paths that have the same starting and ending points, are equal. Therefore, $|\overrightarrow{q_iq_{i+1}}|_1 = |\pi_{q_iq_{i+1}}|_1$.

Now, we prove that $\pi_{q_iq_{i+1}} \subset \mathcal{G} = \langle V, E \rangle$. It suffices to show that each vertex of $\pi_{q_iq_{i+1}}$ is in V and between every two consecutive vertices of $\pi_{q_iq_{i+1}}$ there is an edge in E. Each vertex of $\pi_{q_iq_{i+1}}$ is either the result of the orthogonal projection or an endpoint of an interval. Therefore, each vertex is either a Steiner point or a vertex of the white-surface. In both cases, the vertex is in V. In addition, between every two consecutive vertices of $\pi_{q_iq_{i+1}}$ there is an edge in E because every two consecutive vertices of $\pi_{q_iq_{i+1}}$ lie on the boundary of a cell and, by the construction of $\mathcal{G}$, all members of V that lie on the boundary of a cell are linked by edges in E.

Case 2. In [11], Section 4, Lemma 4, it is proved that if $\overrightarrow{q_iq_{i+1}}$ does not lie completely within a row and within a column of $\mathcal{F}_\varepsilon(SF)$, then there is a xy-monotone path $\pi'_{q_iq_{i+1}}$, from q_i to q_{i+1}, such that its edges lie completely within a row and within a column of $\mathcal{F}_\varepsilon(SF)$. For each edge of $\pi'_{q_iq_{i+1}}$, we apply case 1. Then, we concatenate the resulting xy-monotone paths for edges of $\pi'_{q_iq_{i+1}}$, to obtain $\pi_{q_iq_{i+1}}$. Since, xy-monotone paths for edges of $\pi'_{q_iq_{i+1}}$ are in $\mathcal{G}$ (as we proved in Case 1), the resulting path, $\pi_{q_iq_{i+1}}$, is a xy-monotone path in $\mathcal{G}$. Therefore, $|\overrightarrow{q_iq_{i+1}}|_1 = |\pi_{q_iq_{i+1}}|_1$.

Corollary 1 *For any pair of s_i and t_j, if t_j is reachable from s_i by a path in $\mathcal{W}$, then there is a path from s_i to t_j, in $\mathcal{G}$, that is a L_1 shortest path in $\mathcal{W}$.*

Corollary 2 *A shortest path in $\mathcal{G}$, from s' to t', yields an optimal solution for our problem setting.*

Proof. Let Π'_{opt} be a shortest path in $\mathcal{G}$, from s' to t'. We remove s' and t' from the head and tail of Π'_{opt}. The result, Π_{ij}, is a shortest path from s_i to t_j. Therefore, among all possible shortest paths $\Pi_{k\ell}$, for s_k and t_ℓ, $k, \ell = 1, \ldots, |V_H|$, the pair (s_i, t_j) has a shortest L_1 shortest path, Π_{ij}. By Corollary 1, $\Pi_{ij} \subset \mathcal{G}$ is a L_1 shortest path in $\mathcal{W}$. Each point on Π_{ij} is corresponding to a point, p, on H and a point, q, on T, such that the Euclidean distance of p and q is less than ε. By Observation 1, Π_{ij}, is corresponding to a path, P, in H, from $v_i \in V_H$ to $v_j \in V_H$, and a parameterization, f, of T. The summation of the Euclidean length of P, $\|P\|$, and the walking length of T, $\mathcal{L}_f(T)$, is equal to the L_1 length of Π_{ij}. Since Π_{ij} is a shortest L_1 shortest path, P and f minimize the matching cost, $\mathcal{M}^\varepsilon(H, T)$ (Equation 2).

Theorem 1 *Let H be a planar graph with a straight-line embedding in a plane, T be a directed polygonal curve, and $\varepsilon > 0$ be a distance. A path, $P : [0, 1] \to H$, between two vertices of H, and a parameterization, f, of T, that minimize the sum of the walking length of T and P, can be found in polynomial time and*

space. It is guaranteed that at any time $t \in [0,1]$, the Euclidean distance between $P(t)$ and $T(f(t))$ is at most ε.

Proof. The correctness follows directly from Corollary 2. The deformed free-space surface, $\mathcal{S}$, has $\mathcal{O}(mn)$ cells, where m (n, respectively) is the number of edges of H (T, respectively). Each cell of $\mathcal{S}$ has at most four intervals and at most $\mathcal{O}(m+n)$ Steiner points on its intervals. Therefore, the graph $\mathcal{G}$ has $\mathcal{O}\left(mn(m+n)\right)$ vertices and $\mathcal{O}\left(mn(m+n)^2\right)$ edges (including the extra edges that connect s' and t' to the graph). In addition, it takes $\mathcal{O}\left(n^2\right)$ time to compute all Type 1 Steiner points for each free-space face. Therefore, computing S_1 takes $\mathcal{O}\left(mn^2\right)$ time. In order to compute Type 2 Steiner points, we use breadth first search for each interval endpoint to propagate the projection on the instance of the graph, H_c, in the plane $z = c$. Therefore, computing S_2 takes $\mathcal{O}\left(nm^2\right)$ time. At the end, it is possible to find a shortest path in $\mathcal{G}$, from s' to t', in $\mathcal{O}\left(mn(m+n)^2\right)$ time, by using Dijkstra's algorithm. Therefore, both the total time and space complexities are $\mathcal{O}\left(mn(m+n)^2\right)$.

4 Improvement

In Section 3, we showed that the graph $\mathcal{G} = \langle V, E\rangle$ contains a path that yields an optimal solution for our problem setting. The bottleneck in the time complexity of the algorithm in Section 3 is due to the number of edges of $\mathcal{G}$. In this section, we construct a new graph $\mathcal{G}' = \langle V, E'\rangle$, such that $|E'| < |E|$ and it preserves the connectivity information of $\mathcal{G}$. More precisely, if there is a path, from $v_i \in V$ to $v_j \in V$, in $\mathcal{G}$, then there is a path, from v_i to v_j, in $\mathcal{G}'$, with the same L_1 length.

Based on the construction of $\mathcal{G}$, there are at most $\mathcal{O}(m+n)$ vertices in V (including the interval endpoints and Steiner points) on the boundary of each cell, C, of $\mathcal{S}$. We connect these $\mathcal{O}(m+n)$ vertices by a linear number of edges, in E', as follows. The weight of each edge in E' is equal to its L_1-length. Let T, B, L, and R be the intervals on the top, bottom, left and right side of C, respectively. Suppose cell C is in a $2D$ Cartesian coordinate system and the vertices on each interval $I \in \{T, B, L, R\}$ are sorted by x and y. Every two adjacent vertices, v_i and v_{i+1}, on I, are linked by two directed edges, $\langle v_i, v_{i+1}\rangle$ and $\langle v_{i+1}, v_i\rangle$ (Figure 7). Every two of the eight interval endpoints are linked by two directed edges assuming they are not identical. A vertex v_i on interval L (T, respectively), is linked by two directed edges to another vertex v'_i on R (B, respectively) if v'_i has the same y (x, respectively) coordinate as v_i; the two edges are denoted by $\langle v_i, v'_i\rangle$ and $\langle v'_i, v_i\rangle$, respectively. By this approach, each vertex of $\mathcal{G}'$ on the boundary of C is connected to a constant number of vertices of $\mathcal{G}'$ on the boundary of C. It is now straightforward to prove the following lemma.

Lemma 2 *Let v_i and v_j be two vertices, in V, on the boundary of a cell, C, of $\mathcal{S}$. There is a path, from v_i to v_j, in $\mathcal{G}'$, that has the same L_1 length as the direct line segment between them.*

Corollary 3 *There is a path in $\mathcal{G}'$ that realizes an optimal solution for our problem setting.*

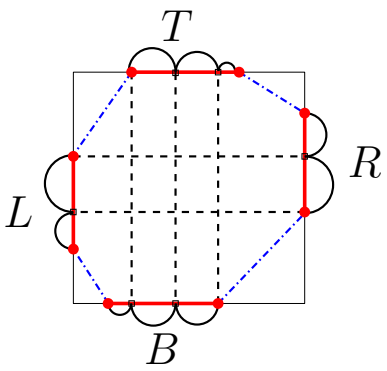

Fig. 7. A cell of the free-space surface is drawn. The red solid line segments show the four intervals on the boundary of the cell. The arcs show the edges in E' that connect every two adjacent vertices of $\mathcal{G}'$, on each interval. The dashed black line segments show the edges in E' that connect a vertex with its orthogonal projection on the opposite side of the cell. The dash dotted blue line segments show some of the edges that connect endpoints of the intervals. For simplicity, we did not draw all of them.

Theorem 2 *Let H be a planar graph with a straight-line embedding in a plane, T be a directed polygonal curve, and $\varepsilon > 0$ be a distance. A path, $P : [0, 1] \to H$, between two vertices of H, and a parameterization, f, of T, that minimize the sum of the walking length of T and P, can be found in $\mathcal{O}(nm(n + m) \log(nm))$ time and $\mathcal{O}(nm(n + m))$ space, where n (m, respectively) is the number of edges of T (H, respectively). It is guaranteed that at any time $t \in [0, 1]$, the Euclidean distance between $P(t)$ and $T(f(t))$ is at most ε.*

Proof. The correctness follows directly from Corollary 3. The number of vertices and edges of $\mathcal{G}'$ (and the total space complexity) is upper-bounded by $\mathcal{O}(nm(n+m))$. Using Dijkstra's algorithm, we find a shortest path in $\mathcal{G}'$, from s' to t'. Therefore, the time complexity of our algorithm is $\mathcal{O}(nm(n+m) \log(nm))$. Note that if there is no pair of (s_k, t_ℓ), $k, \ell = 1, \ldots, |V_H|$, in a connected component of $\mathcal{G}'$, then there is no feasible solution.

5 Weighted non-planar graphs

We assumed that the input graph H is planar. That makes the illustration of the algorithm easier since the faces of the free-space surface do not intersect except at the boundary of the faces. However, all lemmas and theorems, derived in Section 3 and 4, are proved without making an assumption regarding the planarity of H. Therefore, the algorithm proposed in this paper remains correct for any graph for which a straight-line embedding in a plane is provided (see [5], Section 2.7). In the embedding, the edges of the graph may intersect. Transition from one edge to another is allowed only at a vertex.

We also assumed that H is unweighted. Here, we sketch how the proposed algorithm can be generalized to also handle the problem instance, when H is weighted. Suppose that each edge of H has a non-negative, real weight. A weight could represent the cost of moving on the edge of the graph. The edges of the input polygonal curve T could also have weights capturing the costs of moving forwards and backwards. The objective is to find a path in H whose weighted walking length is minimized. In the weighted problem setting, inside each cell of the free-space surface, there are two weights, one corresponding to an edge of T and one corresponding to an edge of H. These weights are fixed inside the cell and do not change. Therefore, in the construction of $\mathcal{G}$ or $\mathcal{G}'$, instead of computing the L_1 length for each edge, e, we compute the orthogonal projections of e onto H and T. Then, we multiply the projection lengths with the corresponding weights, and the sum of these multiplications is the weight that we assign to e. The remaining parts of the algorithm remains the same and the time and space complexities do not change.

6 Conclusion

In this paper, we discussed a geometric algorithm for the map matching problem that minimizes the walking length. We established that this problem setting is dual to a weighted shortest path problem. Then, we proposed an algorithm with $\mathcal{O}\left(mn\left(m+n\right)\log(mn)\right)$ time and $\mathcal{O}\left(mn\left(m+n\right)\right)$ space complexities, where m (n, respectively) is the number of edges of H (T, respectively). At the end, we discussed that the proposed algorithm is easily adaptable to handle weighted non-planar graphs. It is still open if we can improve the proposed algorithm further, for planar graphs. The main challenge here is the existence of cycles in the input graph and propagation through the cycles.

7 Acknowledgment

The authors would like to thank Carola Wenk for suggesting this topic and constructive comments, and Omid Gheibi for valuable discussions.

References

1. Y. Zheng. Trajectory Data Mining: An Overview. *ACM Trans. Intelligent Systems and Technology*, 6(3), Article 1, 2015.
2. B. Chen, H. Yuan, Q. Li, W. Lam, S. Shaw, K. Yan, Map-matching algorithm for large-scale low-frequency floating car data. *Int. J. Geogr. Inf. Sc.*, 28(1):22-38, 2014.
3. F. Ruan, Z. Deng, Q. An, K. Wang, X. Li. A Method of Map Matching in Indoor Positioning. In *CSNC 2014 Proceedings: Volume III*, Lecture Notes in Electrical Engineering, 305, pp. 669-679, 2014.
4. K. Asakura, M. Takeuchi, T. Watanabe. A Pedestrian-oriented Map Matching Algorithm for Map Information Sharing Systems in Disaster Areas, *Int. J. Know. Web Intel.*, 3(4):328-342, 2012.

5. H. Alt, A. Efrat, G. Rote, and C. Wenk. Matching planar maps. In *Proceeding of the fourteenth annual ACM-SIAM symposium on discrete algorithms*, pp. 589-598, 2003.
6. H. Alt and M. Godau. Computing the Fréchet distance between two polygonal curves. *Int. J. Comput. Geometry Appl.*, 5:75-91, 1995.
7. S. Brakatsoulas, D. Pfoser, R. Salas, and C. Wenk. On map-matching vehicle tracking data. In *Proceeding of VLDB*, pp. 853-864. ACM, 2005.
8. D. Chen, A. Driemel, L. Guibas, A. Nguyen, C. Wenk. Approximate Map Matching with respect to the Frchet Distance. In *Proceeding of 13th ALENEX*, pp.75-83, 2011.
9. T. Flynn, S. Connery, M. Smutok, R. Zeballos, I. Weisman. Comparison of cardiopulmonary responses to forward and backward walking and running. *Med. Sci. Sports Exerc.*, 26(1):89-94, January 1994.
10. A. Gheibi, A. Maheshwari, J.-R. Sack, C. Scheffer. Minimum Backward Fréchet Distance. In *Proceedings of the 22nd ACM SIGSPATIAL*, pp. 381-388, 2014.
11. A. Gheibi, A. Maheshwari, J.-R. Sack. Weighted Minimum Backward Fréchet Distance. *accepted to 27th CCCG*, Kingston, 2015.
12. M. Z. A. Bhuiyan, G. Wang, A. V. Vasilakos. Local Area Prediction-Based Mobile Target Tracking in Wireless Sensor Networks, *IEEE Trans. Comput.*, 64(7):1968-1982, 2015.
13. H. Vachhani. Continuous Spatio Temporal Tracking of Mobile Targets, *Master's Thesis, Arizona State University*, 2014.
14. S. K. Ghosh and D. M. Mount. An output-sensitive algorithm for computing visibility graphs. *SIAM J. Comput.*, 20(5):888-910, 1991.
15. S. Har-Peled and B. Raichel. The Fréchet Distance Revisited and Extended. In *Proceedings of the 27th ACM SoCG*, pp. 448-457, 2011.