Rooted Uniform Monotone Minimum Spanning Trees

Konstantinos Mastakas and Antonios Symvonis

School of Applied Mathematical and Physical Sciences
National Technical University of Athens, Greece
{kmast,symvonis}@math.ntua.gr

July 13, 2016

Abstract

In this paper we study the construction of minimum cost geometric spanning graphs on a rooted point set \( P \) where we insist that the sought graphs satisfy a given property. A rooted point set is a point set having one of its points, say \( r \), designated as its root. We focus on the property of monotonicity w.r.t. a single direction (\( y \)-monotonicity), or w.r.t. a pair of orthogonal directions (\( xy \)-monotonicity). In both cases, it turns out that the wanted graphs are trees. For the case of \( y \)-monotonicity (\( xy \)-monotonicity), we want to construct the minimum cost geometric spanning tree \( T \) in which the root \( r \) of the point set is connected to every point \( p \in P \) by a path that is monotone in the \( y \) direction (resp., in both the \( x \) and \( y \) directions).

We present algorithms that, given a rooted point set \( P \), compute the rooted \( y \)-monotone and rooted \( xy \)-monotone minimum spanning tree in \( O(|P|^2) \) time when the directions of monotonicity are given, and in \( O(|P|^2 \log |P|) \) time when the optimum directions have to be determined. We also present simple algorithms that decide whether a given geometric graph on a rooted point set is rooted \( y \)-monotone or rooted \( xy \)-monotone.

1 Introduction

A concept widely studied in the fields of Computational Geometry and Graph Drawing is the concept of monotonicity. A geometric path \( W = (w_0, w_1, w_2, \ldots, w_l) \) is monotone in the direction of \( y \), also called \( y \)-monotone, if \( y(w_0) \leq y(w_1) \leq \ldots \leq y(w_l) \) or \( y(w_0) \geq y(w_1) \geq \ldots \geq y(w_l) \), where \( y(p) \) denotes the \( y \) coordinate of a point \( p \). \( W \) is monotone if there exists an axis \( y' \) s.t. \( W \) is \( y' \)-monotone. Arkin et al. [5] proposed an algorithm which connects two given points by a geometric path that is monotone in a given direction (resp. in an arbitrary direction) and does not cross a set of obstacles.

A geometric graph \( G = (P, E) \) is monotone if every pair of points \( p, q \in P \) is connected by a monotone geometric path. Note that, according to this definition, the direction of monotonicity does not need to be the same for every pairs of points \( p, q \in P \). If we insist of having a single direction of monotonicity,
we say that the graph is uniform monotone. When the direction of monotonicity is known, say y, then the graph is called y–monotone.

Monotone graphs were introduced by Angelini et al. [3]. Given a graph \( G \), the problem of drawing the vertices of \( G \) on the plane so that each pair of vertices is connected by a monotone path has been widely studied; see, for example, [2, 3, 4, 7, 10, 11, 12, 13, 17]. The reverse problem, namely, given a point set \( P \) we are asked to construct a monotone geometric graph \( G \) on the points of \( P \), has trivial solutions. Simply observe that the complete graph \( K_{|P|} \) on the points of \( P \) as well as the path graph \( W_{|P|} \) which visits all points of \( P \) in increasing order of their \( y \)-coordinates are \( y \)–monotone geometric graphs for \( P \).

The problem of constructing a minimum cost geometric graph spanning a plane point set \( P \) has also received extensive attention [21]. Shamos and Hoey [24] showed that it can be solved in \( \Theta(|P| \log |P|) \) time by utilizing the fact that the wanted tree is a subgraph of any Delaunay triangulation of the points in \( P \). The problem is usually referred to as the Euclidean minimum spanning tree problem since the cost of the tree is taken to be the sum of the Euclidean lengths of its edges. The rectilinear variant where distances between points are taken in the \( L_1 \) metric has also been studied [9, 14].

Combining the Euclidean minimum spanning tree problem with the notion of monotonicity leads to a large number of problems that, to the best of our knowledge, have not been previously investigated. The most general problem can be stated as follows: “Given a point set \( P \) find a minimum cost monotone spanning graph of \( P \), i.e., a graph such that every pair of points \( p, q \in P \) are connected by a monotone path”. Since in a monotone graph the direction of monotonicity need not be the same for all pairs of points, it is not clear whether the minimum cost monotone graph is a tree. We call this problem the Monotone Minimum Spanning Graph problem.

Let \( P \) be a rooted point set, i.e., a point set having a designated point, say \( r \), as its root. In this paper, we focus on a simple variation of the general monotone minimum spanning graph problem. We consider a rooted point set, say \( P \), and we do not insist on having monotone paths between every pair of points but rather only between the root \( r \) with all other points of \( P \). Moreover, we insist that all paths are uniform in the sense that they are all monotone with respect to the same direction, i.e., we build rooted uniform monotone graphs. Actually, as it turns out, in this problem the sought graphs are trees and, thus, we refer to it as the rooted Uniform Monotone Minimum Spanning Tree (for short, rooted UMMST) problem. In the rooted UMMST problem we have the freedom to select the direction of monotonicity. When we are restricted to have monotone paths in a specific direction, say \( y \), we have the rooted \( y \)-Monotone Minimum Spanning Tree (for short, rooted \( y \)-MMST) problem. Figure 1(a) illustrates a rooted \( y \)–monotone spanning graph of the rooted point set \( P = \{r, a, b, c, d, e, f, g\} \). The rooted \( y \)–MMST for the same point set is given in Figure 1(b).

Let \( W \) be a geometric path. If \( W \) is both \( x \)–monotone and \( y \)–monotone then it is denoted as \( xy \)–monotone. Furthermore, if there exist a Cartesian System \( x'y' \) s.t. \( W \) is \( x'y' \)–monotone then \( W \) is \( 2D \)–monotone. Based on \( xy \)–monotone paths and in analogy to the monotone, \( y \)–monotone and uniform monotone graphs, we define the \( 2D \)–monotone, \( xy \)–monotone and uniform \( 2D \)–monotone graphs.
Naturally, the notion of monotonicity in the rooted UMMST and rooted $y$-MMST problems can be replaced by any other property that the minimum spanning graph/tree may be required to satisfy. Replacing it by 2D-monotonicity gives rise to the corresponding rooted Uniform 2D-Monotone Minimum Spanning Tree (for short, rooted 2D-UMMST) and rooted $xy$-Monotone Spanning Tree (for short, rooted $xy$-MMST) problems.

While it is the first time that rooted point sets are studied in the context of monotonicity, they have been previously studied in the context of minimum spanning trees. The capacitated minimum spanning tree is a tree that has a designated vertex $r$ (its root) and each of the subtrees attached to $r$ contains no more than $c$ vertices. $c$ is called the tree capacity. Solving the capacitated minimum spanning tree problem optimally has been shown by Jothi and Raghavachari to be NP-hard \[16\]. In the same paper, they have also presented approximation algorithms for the case where the vertices correspond to points on the Euclidean plane.

Related to path monotonicity is the increasing-chord property. Let $d(p,q)$ denote the Euclidean distance between points $p$ and $q$. A path/curve $W$ is increasing-chord if for every four points $p_1, p_2, p_3, p_4$ traversed in this order along it, it holds that $d(p_2,p_3) \leq d(p_1,p_4)$. Alamdari et al. \[1\] noted that any 2D-monotone path is increasing-chord. Increasing-chord curves were studied in \[18,22\]. A geometric graph $G = (P,E)$ is increasing-chord if each two points of $P$ are connected by an increasing-chord path. Increasing-chord graphs were introduced by Alamadari et al. \[1\]. Drawing a graph as increasing-chord graph has been studied in \[1,20\]. Furthermore, the problem of introducing straight line segments between points of a given point set $P$ s.t. each pair of points of $P$ is connected by an increasing-chord path is studied in \[1,6,19\].

1.1 Our Results

We present algorithms that solve the rooted $y$–MMST and the rooted $xy$–MMST problems in $O(|P|^2)$ time when the directions of monotonicity are given, as well as algorithms that solve the rooted UMMST and the rooted 2D-UMMST problems in $O(|P|^2 \log |P|)$ time when the optimum directions have to be determined. For all four problems, it is easy to derive a $\Omega(|P| \log |P|)$ time lower bound. Finally, we also present simple algorithms that decide whether a given geometric graph on a rooted point set is rooted uniform monotone or rooted uniform 2D-monotone.
2 Definitions and Notation

In this article we deal with the Euclidean plane, i.e. every point set is a subset of \( \mathbb{R}^2 \). Let \( x,y \) be the axes of a Cartesian System. The \( x \) and \( y \) coordinates of a point \( p \) are denoted by \( x(p) \) and \( y(p) \), respectively.

Let \( P \) be a point set and \( a \) be a real number. By \( P_{y\leq a} \) we denote the set of points of \( P \) that have \( y \) coordinate less than or equal to \( a \). Subsets \( P_{y\geq a}, P_{x\leq a}, P_{x\geq a}, P_{y\leq a}, P_{y\geq a}, P_{|x|\geq a}, P_{|y|\geq a} \) and \( P_{x,y\leq a} \) are similarly defined.

\( P_{x\leq a,y\leq a} \) denotes the set \( P_{x\leq a} \cap P_{y\leq a} \). Subsets \( P_{x\geq a,y\geq a}, P_{x\geq a,y\leq a} \) and \( P_{x\geq a,y\leq a} \) are defined similarly. The Euclidean plane is divided into four quadrants, i.e. the quadrants \( R_x^2 \), \( R_y^2 \), \( R_x^2 \cap R_y^2 \) and \( R_x^2 \cup R_y^2 \).

A point set \( P \) is called positive (negative) w.r.t. the direction of \( y \) or \( y - \)positive (\( y - \)negative) if for each \( p \in P \), \( y(p) \geq 0 \) (resp. \( y(p) \leq 0 \)).

Let \( P \) be a point set and \( p \) be a point of the plane, then \( d(p,P) \) denotes the Euclidean distance from \( p \) to the point set \( P \), i.e. \( d(p,P) = \min_{q \in P} d(p,q) \).

The straight line segment with endpoints \( p \) and \( q \) is denoted as \( \overline{pq} \).

The slope of a straight line \( L \) is the angle that we need to rotate the \( x \) axis counterclockwise s.t. the \( x \) axis becomes parallel to \( L \). Each slope belongs to the range \([0,\pi]\).

A geometric path \( W = (w_0, w_1, \ldots, w_l) \) consists of the points \( w_0, w_1, \ldots, w_l \) and the straight line segments \( \overline{w_0w_1}, \overline{w_1w_2}, \ldots, \overline{w_lw_0} \). A geometric graph \( G = (P,E) \) consists of a point set \( P \) and a set \( E \) of straight line segments with endpoints in \( P \) which are denoted as its edges. If \( P \) is rooted, then \( G \) is called a rooted geometric graph. The cost of a geometric graph \( G = (P,E) \), denoted as \( \text{cost}(G) \), is the sum of the lengths of its edges.

Let \( G_1 = (P_1, E_1), G_2 = (P_2, E_2), \ldots, G_n = (P_n, E_n) \) be \( n \) geometric graphs. Then, the union \( G_1 \cup G_2 \cup \ldots \cup G_n \) is the geometric graph \( G = (P,E) \) s.t. \( P = P_1 \cup P_2 \cup \ldots \cup P_n \) and \( E = E_1 \cup E_2 \cup \ldots \cup E_n \).

In this paper we consider only rooted point sets. W.l.o.g., we assume that the root \( r \) of the point sets coincides with the origin of the Cartesian System, i.e., \( x(r) = y(r) = 0 \). We also assume that all the point sets are in general position, i.e. no three points are collinear.

3 The Rooted \( y \)-Monotone Minimum Spanning Tree (rooted \( y \)-MMST) Problem

In this section we study the rooted \( y \)-MMST problem. We provide a characterization of the rooted \( y \)-MMST of rooted \( y \)-positive (or \( y \)-negative) point sets. Based on this characterization we develop an algorithm that constructs the rooted \( y \)-MMST of a rooted point set \( P \). We also provide a lower bound on the time complexity of any algorithm that solves the rooted \( y \)-MMST problem as well as a simple recognition algorithm for rooted \( y \)-monotone graphs.

Observation 1. Let \( P \) be a rooted point set and \( G = (P,E) \) be a rooted \( y \)-monotone spanning graph of \( P \) and let \( \overline{pq} \in E \) with \( y(p) < 0 < y(q) \).

Then, every path from \( r \) to a point \( p \in P \setminus \{r\} \) that traverses \( \overline{pq} \) is not \( y \)-monotone since it moves “south” to \( p \) and then “north” to \( p_n \), or vice versa.
Corollary 1. Let $P$ be a rooted point set, $G_{\text{opt}} = (P, E)$ be the rooted $y$–monotone minimum spanning graph of $P$ and $p_d, p_u \in P$ such that $y(p_d) < 0 < y(p_u)$. Then, $\overline{p_dp_u} \notin E$.

Corollary 1 implies that the root $r$ of a rooted point set $P$ (recall that we assume that $r$ is located at point $(0, 0)$) splits the problem of finding a rooted $y$–monotone minimum spanning graph of $P$ into two independent problems.

Lemma 1. Let $P$ be a rooted point set and $G_{\text{opt}}$ be the rooted $y$–monotone minimum spanning graph of $P$. Furthermore, let $G_{y \leq 0}^{\text{opt}}$ and $G_{y \geq 0}^{\text{opt}}$ be the rooted $y$–monotone minimum spanning graphs of $P_{y \leq 0}$ and $P_{y \geq 0}$, respectively. Then, $G_{\text{opt}}$ is the union of $G_{y \leq 0}^{\text{opt}}$ and $G_{y \geq 0}^{\text{opt}}$.

Let $P$ be a rooted $y$–positive (or $y$–negative) point set. We define $S[P, y]$ to be the sequence of $P$’s points ordered by the following rule: “The points of $S[P, y]$ are ordered w.r.t. their absolute $y$ coordinates and, if two points have the same $y$ coordinate, then they are ordered w.r.t. their distance from the preceding points in $S[P, y]$.” Recall that, due to the assumption that the points of $P$ are in general position, only two points may share the same $y$-coordinate. More formally, $S[P, y] = (r = p_0, p_1, p_2, \ldots, p_n)$ s.t. $|y(p_0)| \leq |y(p_1)| \leq |y(p_2)| \leq \ldots \leq |y(p_n)|$ and $|y(p_i)| = |y(p_{i+1})|$ implies that $d(p_i, \{p_0, p_1, \ldots, p_{i-1}\}) \leq d(p_{i+1}, \{p_0, p_1, \ldots, p_{i-1}\})$ and $P = \{p_0, p_1, p_2, \ldots, p_n\}$. We now give a characterization of the rooted $y$–monotone minimum spanning graph of $P$.

Theorem 1. Let $G = (P, E)$ be a rooted geometric graph where $P$ is a rooted $y$–positive (or $y$–negative) point set with $S[P, y] = (r = p_0, p_1, p_2, \ldots, p_n)$. Then, $G$ is the rooted $y$–monotone minimum spanning graph of $P$ if and only if (i) $p_n$ is connected in $G$ only with the point $p_j$ such that $d(p_n, p_j) = d(p_n, \{p_0, p_1, \ldots, p_{n-1}\})$ and (ii) $G \setminus \{p_n\}$ is the rooted $y$–monotone minimum spanning graph of $P \setminus \{p_n\}$.

Proof. The $(\Rightarrow)$ direction can be easily proved by contradiction. We now prove the $(\Leftarrow)$ direction. $G$ is a rooted $y$–monotone graph that spans $P$ since $G \setminus \{p_n\}$ is a rooted $y$–monotone graph that spans $P \setminus \{p_n\}$ and $p_n$ is connected with another point in the graph. Let $G_{\text{opt}}$ be the rooted $y$–monotone minimum spanning graph of $P$. Then, $p_n$ is connected in $G_{\text{opt}}$ with some other point in $P$, and thus $G_{\text{opt}}$ contains an edge of cost at least $d(p_n, \{p_0, p_1, \ldots, p_{n-1}\})$. Furthermore, the graph $G_{\text{opt}} \setminus \{p_n\}$ is also rooted $y$–monotone, hence its cost is at least the cost of the rooted $y$–monotone minimum spanning graph of $P \setminus \{p_n\}$. Thus, $G_{\text{opt}}$ has at least the same cost as $G$.

Corollary 2. The rooted $y$–monotone minimum spanning graph of a rooted point set $P$ is a geometric tree.

Let $P$ be a rooted $y$–positive (or $y$–negative) point set and $S[P, y] = (r = p_0, p_1, p_2, \ldots, p_n)$. Then, for each $p_i, i = 1, 2, \ldots, n$, we call the closest point to $p_i$ that belongs in $\{p_0, p_1, \ldots, p_{i-1}\}$ the parent of $p_i$ and we denote it by $\text{par}(p_i)$. More formally, $\text{par}(p_i) = p_j$ if and only if $p_j \in \{p_0, p_1, \ldots, p_{i-1}\}$ and $d(p_i, p_j) = d(p_i, \{p_0, p_1, \ldots, p_{i-1}\})$.

Remark 1. Given a rooted $y$–positive (or $y$–negative) point set $P$, computing $S[P, y]$ and $\text{par}(p)$ for each $p \in P \setminus \{r\}$ can be performed in $O(|P|^2)$ time.
Algorithm 1 rooted $y$–MMST

Input: A rooted $y$–positive (or $y$–negative) point set $P$.

Output: The rooted $y$–Monotone Minimum Spanning Tree of $P$.

1. Create the sorted sequence $S[P,y] = (r = p_0, p_1, p_2, \ldots, p_n)$ of point set $P$.
2. $T \leftarrow$ the geometric graph with $P$ as its vertex set and $\emptyset$ as its edge set.
3. for $i \leftarrow 1$ to $n$ do
   4. $\text{par}(p_i) \leftarrow p_j \in \{p_0, p_1, \ldots, p_{i-1}\}$ s.t. $d(p_i, p_j) = d(p_i, \{p_0, p_1, \ldots, p_{i-1}\})$.
   5. Insert the edge $\text{par}(p_i)p_i$ in $T$.
   6. return $T$.

Proof. We first construct $P$ by assigning direction to the edges and removing some of them. Let $T$ and $P$ be an algorithm on $P$ to the rooted point set $P$.

Output: $y$–MMST of $P$.

Input: A rooted $y$–positive (or $y$–negative) point set $P$.

We now describe in Algorithm 1 how, given a rooted $y$–monotone graph $P$, we can compute the rooted $y$–MMST of $P$.

Theorem 2. The rooted $y$–MMST of a rooted point set $P$ can be computed in $O(|P|^2)$ time.

Proof. We first construct $P_{y \leq 0}$ and $P_{y \geq 0}$ to construct $T_{y \leq 0}$ and $T_{y \geq 0}$, respectively. By Theorem 1, $T_{y \leq 0}$ and $T_{y \geq 0}$ are the rooted $y$–MMSTs of $P_{y \leq 0}$ and $P_{y \geq 0}$, respectively. By Remark 1 we can compute $T_{y \leq 0}$ and $T_{y \geq 0}$ in $O(|P|^2)$ time. By Lemma 1 $T_{y \leq 0} \cup T_{y \geq 0}$ is the rooted $y$–MMST of $P$.

In the next Theorem, we give a lower bound for the time complexity of any algorithm which given a rooted point set $P$ produces the rooted $y$–MMST of $P$.

Theorem 3. Any algorithm which given a rooted point set $P$, produces the rooted $y$–MMST of $P$ requires $\Omega(|P| \log |P|)$ time.

Proof. We use the reduction from sorting that was given by Shamos. Let $(a_1, a_2, \ldots, a_n)$ be a sequence of nonnegative integers. We reduce this sequence to the rooted point set $P = \{(r = (0, 0), (a_1, a_1'), (a_2, a_2'), \ldots, (a_n, a_n'))\}$. Then, the rooted $y$–MMST of $P$ contains exactly the edges $(p_i, p_j)$, $i = 1, 2, \ldots, n$, where $(a_1', a_2', \ldots, a_n')$ is the sorted permutation of $(a_1, a_2, \ldots, a_n)$. The lower bound follows since sorting $n$ numbers requires $\Omega(n \log n)$ time.

We conclude this section, by realizing that rooted $y$–monotone graphs can be efficiently recognized. Our approach is similar to the approach employed in the third section of [5].

Theorem 4. Let $G = (P, E)$ be a rooted connected geometric graph with root $r$. Then, we can decide in $O(|E|)$ time if $G$ is rooted $y$–monotone.

Proof. We first transform $G$ into a directed geometric graph $\overrightarrow{G}$ in $O(|E|)$ time, by assigning direction to the edges and removing some of them. Let $\overrightarrow{pq}$ be an edge of $G$. If $p$ and $q$ belong to opposite half planes w.r.t. the $x$ axis then $\overrightarrow{pq}$ cannot be used in a $y$–monotone path from $r$ to a point of $P \setminus \{r\}$ (see Observation 1). Hence, we remove the edge $\overrightarrow{pq}$ from the graph. If $y(p) = y(q)$ then we insert both $\overrightarrow{pq}$ and $\overrightarrow{qp}$ in $\overrightarrow{G}$. Otherwise, assuming w.l.o.g., that $|y(p)| < |y(q)|$, we insert $\overrightarrow{pq}$ in $\overrightarrow{G}$. $G$ is rooted $y$–monotone if and only if $r$ is connected
with all other points of $P$ in $G$. The latter can be easily decided in $O(|E|)$ time by a depth first search traversal.

4 The Rooted Uniform Monotone Minimum Spanning Tree (rooted UMMST) Problem

In this section, we focus our attention to solving the rooted UMMST problem for a rooted point set $P$. In contrast with the rooted $y$–MMST problem where the direction of monotonicity was given, here we are asked to determine the optimum direction of monotonicity, say $y'$, and the corresponding rooted $y'$–MMST.

Crucial to the development of our algorithm for the rooted UMMST problem is the fact that it is sufficient to take into account only $O(|P|^2)$ different directions of monotonicity (to be proved in Lemma 2). This fact is based on the following observations.

Observation 2. Let $y'$ and $y''$ be axes with opposite directions. Then, the rooted $y'$–MMST of $P$ is the same with the rooted $y''$–MMST of $P$. Hence, in computing the rooted UMMST of $P$ we only need to take into account the $y'$ axes such that the angle that we need to rotate the $x$ axis counterclockwise to become codirected with the $y'$ is less than $\pi$.

Observation 3. Let $y'$ be an axis. If we rotate the $y'$ axis counterclockwise then the sequence $S[y' \geq 0]$ (or the sequence $S[y' \leq 0]$) changes only when the $y'$ axis reaches (or moves away from) the perpendicular to a line passing through two points of $P$. Then, by Theorem 1, the rooted $y'$–MMST of $P$ may only change at the same time.

Based on Observations 2 and 3, we define the set $\Theta = \{\theta \in [0, \pi) : \theta$ is the slope of the perpendicular to a line passing through two points of $P\}$. We also define $S[\Theta]$ to be the sorted sequence that contains the slopes of $\Theta$ in increasing order, i.e. $S[\Theta] = (\theta_0, \theta_1, \ldots, \theta_{m-1}), \theta_i < \theta_{i+1}, i = 0, 1, \ldots, m-2$ and $m \leq |P|^2$.

We further define the set $\Theta_{\text{critical}} = \{\theta_0, \theta_1, \ldots, \theta_{m-1}\} \cup \{\frac{\theta_{i} + \theta_{i+1}}{2}, \frac{i}{2}, \theta_{m-2} + \theta_{m-1}, \frac{\theta_{m-2} + \theta_{m-1} + \pi}{2}\}$ which we call the critical set of slopes since, as we show in Lemma 2, examining the axes with slope in $\Theta_{\text{critical}}$ is sufficient for computing the rooted UMMST of $P$. $|\Theta_{\text{critical}}| = O(|P|^2)$. We now assign “names” to the axes with slopes in $\Theta_{\text{critical}}$. Let $y_i²$ be the axis with slope $\theta_i, i = 0, 1, \ldots, m - 1$ and let $y_{2i+1}$ be the axis with slope $\frac{\theta_i + \theta_{i+1}}{2}, i = 0, 1, \ldots, m - 2$, and $y_{2m-1}$ be the axis with slope $\frac{\theta_{m-2} + \theta_{m-1} + \pi}{2}$. Note that the subscript of each axis gives its order when the axes are sorted w.r.t. their slope. Note also that axes with even subscripts correspond to slopes perpendicular to straight lines passing through two points of $P$.

Remark 2. The axes with slopes in $\Theta_{\text{critical}}$ can be computed in $O(|P|^2 \log |P|)$ time.

Lemma 2. The rooted UMMST of $P$ is one of the rooted $y'$–MMST of $P$ over all axes $y'$ with slope in $\Theta_{\text{critical}}$ and, more specifically, the one of minimum cost.
Proof. Let $y'$ and $y''$ be axes of slope $\theta'$ and $\theta''$, respectively, such that $\theta_i < \theta', \theta'' < \theta_{i+1}$ for some $0 \leq i \leq m-2$ and let $T'$, $T''$ be the rooted $y'$−MMST and the rooted $y''$−MMST of $P$, respectively. By Observation 3, $\text{cost}(T') = \text{cost}(T'')$.

As a result, we need to take into account only one of the $y'$ axes of slope $\theta'$ with $\theta_i < \theta' < \theta_{i+1}$. We take into account the axis $y_{2i+1}$. Additionally, we take into account the axis $y_{2m-1}$ for which the rooted $y_{2m-1}$−MMST of $P$ is the same with the rooted $y'$−MMST of $P$ for any $y'$ axis with slope in the range $(\theta_{m-1}, \pi) \cup [0, \theta_0)$.

We also need to take into account all the axes of slope $\theta_i$, i.e. the axes $y_{2i}$, $0 \leq i \leq m-1$, since the sorted sequences $S[P_{y_{2i} \geq 0}, y_{2i}]$ and $S[P_{y_{2i} \leq 0}, y_{2i}]$ might be different from every other $S[P_{y_j \geq 0}, y_j]$ and $S[P_{y_j \leq 0}, y_j]$ and, hence, the rooted $y_{2i}$−MMST of $P$ might differ from all the other rooted $y_j$−MMST of $P$, $j \neq 2i$. See, for example, Figure 2 where three consecutive axes $y_{2i-1}, y_{2i}$, and $y_{2i+1}$ with slopes in $\Theta_{\text{critical}}$ are considered for the point set $P = \{r, a, b, c, d\}$. Each rooted $y_j$−MMST of $P$, $j = 2i - 1, 2i, 2i + 1$, is unique.

We now describe our algorithm which produces the rooted UMMST of a rooted point set $P$. Our algorithm is a rotational sweep algorithm. It considers an axis $y'$, which initially coincides with $y_0$, and then it rotates it counterclockwise until $y'$ becomes opposite to the $x$ axis. Throughout this procedure, it updates the rooted $y'$−MMST $T$ of $P$. By Lemma 2, we only need to obtain each rooted $y_i$−MMST of $P$, where $y_i$ is an axis with slope in $\Theta_{\text{critical}}$, $0 \leq i \leq 2m - 1$. As a result, our algorithm can be stated as follows: Let $T_0^{\text{opt}}$ be the rooted $y_0$−MMST of $P$. Our algorithm initially constructs $T_0^{\text{opt}}$ by applying Algorithm 1 described in the previous section, with input $P_{y_0 \geq 0}$ and $P_{y_0 \leq 0}$ and the direction of $y_0$. Then, it iterates for $i = 1, 2, \ldots, 2m - 1$ obtaining at the end of each iteration $T_i^{\text{opt}}$ by modifying $T_{i-1}^{\text{opt}}$.

Our algorithm maintains a tree $T$ which is initially equal to $T_0^{\text{opt}}$ and throughout its operation it evolves to $T_1^{\text{opt}}, T_2^{\text{opt}}, \ldots, T_{2m-1}^{\text{opt}}$. Similarly, it maintains the sequences $S^−$ and $S^+$ which are initially equal to $S[P_{y_i \geq 0}, y_i]$ and $S[P_{y_i \leq 0}, y_i]$, respectively, and are evolved to $S[P_{y_i \geq 0}, y_i]$ and $S[P_{y_i \leq 0}, y_i]$, $i = 1, 2, \ldots, 2m - 1$. Our algorithm stores the axis which corresponds to the produced rooted UMMST of $P$, so far, in the variable “minAxis”. In its final step, it recomputes and returns the rooted “minAxis”−MMST of $P$ using Theorem 2.

The pseudocode of our algorithm is presented in Algorithm 2.
Algorithm 2 rooted UMMST

Input: A rooted point set $P$.

Output: The rooted Uniform Monotone Minimum Spanning Tree of $P$.

1: Compute the axes $y_0, y_1, \ldots, y_{2m-1}$ with slopes in $\Theta_{\text{critical}}$.
2: Compute $T_0^{opt}, S[P_{y_0 \leq 0} \leq y_0]$ and $S[P_{y_0 \geq 0} \geq y_0]$ by applying Algorithm 1 with input $P_{y_0 \leq 0}$ and $P_{y_0 \geq 0}$ and direction, the direction of the $y_0$ axis.
3: $T \leftarrow T_0^{opt}$, $S^+ \leftarrow S[P_{y_0 \leq 0} \leq y_0]$, $S^- \leftarrow S[P_{y_0 \geq 0} \geq y_0]$, $\minCost \leftarrow \text{cost}(T_0^{opt})$ and $\text{minAxis} \leftarrow y_0$.
4: for $i \leftarrow 1$ to $2m-1$ do
5: \hspace{1em} Update $T$, $S^+$, $S^-$ such that $T$ equals to $T_i^{opt}$ and $S^-$ (resp. $S^+$) equals to $S[P_{y_i \leq 0} \leq y_i]$ (resp. $S[P_{y_i \geq 0} \geq y_i]$).
6: \hspace{1em} if $\text{cost}(T) < \minCost$ then
7: \hspace{2em} $\minCost \leftarrow \text{cost}(T)$ and $\text{minAxis} \leftarrow y_i$.
8: \hspace{1em} return the minAxis-MMST of $P$, computed using Theorem 2.

Theorem 5. The rooted UMMST of a rooted point set $P$ can be computed in $O(|P|^2 \log |P|)$ time.

Proof. By Lemma 2, Algorithm 2 produces the rooted UMMST of $P$. We now show that its time complexity is $O(|P|^2 \log |P|)$.

By Remark 2, the axes $y_0, y_1, \ldots, y_{2m-1}$ with slopes in $\Theta_{\text{critical}}$ can be computed in $O(|P|^2 \log |P|)$ time. Let $k_i$ be the number of pairs of points of $P$ that have the same projection onto the $y_i$ axis, $0 \leq i \leq m-1$. Then $\sum_{i=0}^{m-1} k_i = \binom{2m}{2}$. For each $i = 0, 1, \ldots, m-1$, we compute a list $L_i$ which contains these $k_i$ pairs. All $L_i$, $0 \leq i \leq m-1$, can be computed in $O(|P|^2 \log |P|)$ total time.

We now show that we can compute all $T_i^{opt}$, $S[P_{y_i \leq 0} \leq y_i]$ and $S[P_{y_i \geq 0} \geq y_i]$, $0 \leq i \leq 2m-1$ in $O(|P|^2 \log |P|)$ time. For each point $p$ in $S^- \setminus \{r\}$ (resp. in $S^+ \setminus \{r\}$) we maintain a data structure $\text{PD}(p)$ which is a balanced binary tree that contains all the points that precede $p$ in $S^-$ (resp. $S^+$) accompanied with their distance from $p$. More formally, let $S^-$ be equal to $(r = p_0, p_1, \ldots, p_i)$. Then, for each $p_j, j = 1, 2, \ldots, s$, $\text{PD}(p_j)$ contains the pairs $(p_0, d(p_0, p_j))$, $(p_1, d(p_1, p_j))$, \ldots, $(p_{j-1}, d(p_{j-1}, p_j))$. The key of each $(p_i, d(p_i, p_j))$, $l = 0, 1, \ldots, j-1$, is the distance $d(p_i, p_j)$. Similarly, we define $\text{PD}(p)$ for each $p \in S^+ \setminus \{r\}$. Note that for each $p \in P \setminus \{r\}$, the $\text{par}(p)$ in $T$ can be obtained or updated taking into account the $\text{PD}(p)$, since the pair $(\text{par}(p), d(\text{par}(p), p))$ is the element with the minimum key in $\text{PD}(p)$.

$T_0^{opt}$, $S[P_{y_0 \leq 0} \leq y_0]$ and $S[P_{y_0 \geq 0} \geq y_0]$ are computed in $O(|P|^2)$ time (see Theorem 2). Furthermore, computing $\text{PD}(p_j)$ for some $p_j$ in $S^- \setminus \{r\}$ (resp. in $S^+ \setminus \{r\}$), when $S^-$ (resp. $S^+$) equals to $S[P_{y_0 \leq y_0} \leq y_0]$ (resp. $S[P_{y_0 \geq y_0} \geq y_0]$) takes $O(|P| \log |P|)$ time since we have to insert each $p_i$ in $S^- \setminus \{r\}$ (resp. in $S^+ \setminus \{r\}$), with $i < j$, to $\text{PD}(p_j)$ and each such insertion takes $O(|P| \log |P|)$ time. Hence, the total running time for initially computing all $\text{PD}(p), p \in P \setminus \{r\}$, is $O(|P|^2 \log |P|)$.

Let $T$ be equal to $T_i^{opt}$ and let $S^-$ (resp. $S^+$) be equal to $S[P_{y_{i-1} \leq 0} \leq y_{i-1}]$ (resp. $S[P_{y_{i-1} \geq 0} \geq y_{i-1}]$) then $T$ and $S^-$ (resp. $S^+$) can be updated such that $T$ becomes equal to $T_i^{opt}$ and $S^-$ (resp. $S^+$) becomes equal to $S[P_{y_i \leq 0} \leq y_i]$ (resp. $S[P_{y_i \geq 0} \geq y_i]$) in:
1. $O(k_{\frac{1}{3}} \log |P|)$ time if $i$ is even and $y_i$ is not perpendicular to a straight line passing through the root $r$ and another point in $P$.

2. $O(k_{\frac{1}{3}} \log |P|)$ time if $i$ is odd and $y_{i-1}$ is not perpendicular to a straight line passing through the root $r$ and another point in $P$.

3. $O(|P| \log |P|)$ time if $i$ is even and $y_i$ is perpendicular to a straight line passing through the root $r$ and another point $q \in P \setminus \{r\}$.

4. $O(|P| \log |P|)$ time if $i$ is odd and $y_{i-1}$ is perpendicular to a straight line passing through $r$ and another point $q \in P \setminus \{r\}$.

We first explain how to maintain the data structures so that all points relevant to case 1 are treated in $O(k_{\frac{1}{3}} \log |P|)$ time. Since $y_i$ is not perpendicular to a straight line connecting the root $r$ with another point in $P$ then $P_{y_{i-1}} = P_{y_{i-1}} \leq 0$ and $P_{y_{i+1}} = P_{y_{i+1}} \geq 0$. Hence, no point is inserted into (or removed from) $S^−$ or $S^+$. However, some points which had different projections onto the $y_i$ axis, now have the same projection onto the $y_i$ axis.

We only explain how to deal with the points in $S^+$; the points in $S^−$ can be treated similarly. Recall that $L_{\lceil \frac{i}{2} \rceil}$ contains the $k_{\frac{1}{3}}$ pairs of points of $P$ that have the same projection onto the $y_i$ axis. Let $S^+ = \{r = p_0, p_1, \ldots, p_s\}$ and let $(p_j, p_{j+1})$, $(p_j, p_{j+2})$, $\ldots$, $(p_j, p_{j+k+1})$ with $j_1 < j_2 < \ldots < j_k$ be the $k$ pairs of points in $S^+$ that belong to $L_{\lceil \frac{i}{2} \rceil}$, i.e. they are connected by straight line perpendicular to $y_i$. Then, each $p \in S^+ \setminus \{p_{j_1}, p_{j_1+1}, p_{j_2}, p_{j_2+1}, \ldots, p_{j_k}, p_{j_k+1}\}$ has the same relative order with the other points of $S^+$ w.r.t. both the $y_i$ axis and the $y_{i-1}$ axis. Hence, $p$ is placed at the correct position in $S^+$, the parent of $p$ in $T$ is correct (see Theorem 2 and PD(p) does not need to be updated. As a result, the only changes that may be necessary regard the points in $\{p_{j_1}, p_{j_1+1}, p_{j_2}, p_{j_2+1}, \ldots, p_{j_k}, p_{j_k+1}\}$. For these points we may need to recalculate their parent in $T$. We may also need to swap some consecutive $p_{j_l}$ and $p_{j_{l+1}}$, $l = 1, 2, \ldots, k$, in $S^+$. Finally, we may need to update some of the PD($p_{j_l}$) and PD($p_{j_{l+1}}$), $l = 1, 2, \ldots, k$.

For each $l = 1$ to $k$, we do the following:

We compute $d(p_{j_l+1}, \{p_0, p_1, \ldots, p_{j_l-1}\})$ in $O(\log |P|)$ time using PD($p_{j_l+1}$).

If $d(p_{j_l+1}, \{p_0, p_1, \ldots, p_{j_l-1}\}) \geq d(p_{j_l}, \text{par}(p_{j_l}))$, then we do not update anything, since the points $p_{j_l}$ and $p_{j_{l+1}}$ are placed at the correct position in $S^+$. If, on the other hand, $d(p_{j_l+1}, \{p_0, p_1, \ldots, p_{j_l-1}\}) < d(p_{j_l}, \text{par}(p_{j_l}))$, we remove the edges $\text{par}(p_{j_l})p_{j_l}$ and $\text{par}(p_{j_{l+1}})p_{j_{l+1}}$ from $T$. Then, we swap the order of the points $p_{j_l+1}$ and $p_{j_l}$ in $S^+$, i.e. if $S^+$ was previously equal to $(r = p_0, p_1, \ldots, p_{j_l}, p_{j_{l+1}}, \ldots, p_s)$, now $S^+$ becomes equal to $(r = p_0, p_1, \ldots, p_{j_{l+1}}, p_{j_l}, \ldots, p_s)$ with $p_{j_{l+1}}$ equal to $p_{j_l+1}$ and $p_{j_l}$ equal to $p_{j_l}$.

We then insert the pair $(p_{j_l}, d(p_{j_l}, p_{j_{l+1}}))$ into PD($p_{j_{l+1}}$) and remove the pair $(p_{j_{l+1}}, d(p_{j_l+1}, p_{j_l}))$ from PD($p_{j_l}$). Finally, in $T$ we connect the point $p_{j_l}$ (resp. $p_{j_{l+1}}$) with the point $p$ s.t. $(p, d(p, p_{j_l}))$ (resp. $(p, d(p, p_{j_{l+1}}))$) has the minimum key in PD($p_{j_l}$) (resp. PD($p_{j_{l+1}}$)) and update its parent accordingly. Using PD($p_{j_l}$) and PD($p_{j_{l+1}}$), all this process which concerns a single pair of points is completed in $O(\log |P|)$ time. Thus, $k_{\frac{1}{3}}$ pairs of points are treated in $O(k_{\frac{1}{3}} \log |P|)$ time.

Case 2 is treated similarly with Case 1.

We now treat Case 3. First, observe that this case occurs exactly $|P| - 1$ times, i.e. one time for each pair $(r, p), p \in P \setminus \{r\}$. In this case, either $P_{y_{i-1}} \leq 0$ or $P_{y_{i-1}} \geq 0$. W.l.o.g. we assume that $P_{y_{i-1}} \leq 0$.
\( y_{i-1}(q) < 0 \), i.e. \( q \) belongs to \( S^- \) w.r.t. \( y_{i-1} \) while it belongs to both \( S^+ \) and \( S^- \) w.r.t. the \( y_i \) axis.

We first explain how to deal with the insertion of \( q \) into the point set \( P_{n \geq 0} \). We insert \( q \) into the sequence \( S^+ \) right after \( r \). We do not need to update the PD\((q)\) since it already contains only the pair \((r,d(r,q))\). We also do not need to update par\((q)\). Then, for each point \( p \in S^+ \) with \( p \not\in \{r,q\} \) we insert the pair \((q,d(q,p))\) in the PD\((p)\) and if \( d(q,p) \) is the lowest key in PD\((p)\), then we remove par\((p)\) from \( T \), we assign \( q \) to the par\((p)\) and then we insert the edge \( pq \) to \( T \). We are now done with \( q \). All the previously described actions take \( O(|P| \log |P|) \) time, for the single pair \((r,q)\).

Then, for the pairs of points that have the same projection onto the \( y_i \) axis, except for \((r,q)\), we apply the procedure described in Case 4. As shown in that Case, this is done in \( O(k_{1.5} \log |P|) = O(|P| \log |P|) \) time, since \( k_{1.5} < |P| \).

Case 4 is treated similar to Case 3.

Since \( \sum_{i=0}^{m-1} k_i = O(|P|^2) \), the total running time of our algorithm is \( O(|P|^2 \log |P|) \). 

We now proceed to the problem of deciding if a given rooted connected geometric graph is rooted uniform monotone.

Let \( G = (P,E) \) be a rooted connected geometric graph and \( p \) be a point of \( P \setminus \{r\} \). Let \( A(p,y) \) be the set that contains all the adjacent points to \( p \) that are on the same side with \( p \) w.r.t. the \( x \) axis and are strictly closer to the \( x \) axis than \( p \). More formally, \( A(p,y) = \{q : q \in Adj(p) \text{, } q \text{ lies on the same half plane with } p \text{ w.r.t. the } x \text{ axis and } |y(q)| < |y(p)| \} \). Let \( B(y) \) denote the set \( \{p : p \in P \setminus \{r\} \text{ and } A(p,y) \neq \emptyset\} \). Let \( C(y) \) be the set that consists of the points \( p \in P \setminus \{r\} \) that (i) do not belong to \( B(y) \), (ii) are connected with some other point \( q \) with the same \( y \) coordinate and (iii) \( A(q,y) \neq \emptyset \). More formally, \( C(y) = \{p : p \in P \setminus \{B(y) \cup \{r\}\} \text{ such that there exists } q \in Adj(p) \text{ with } y(q) = y(p) \text{ and } A(q,y) \neq \emptyset\} \). An example of a geometric graph and the corresponding sets is given in Figure 3.

![Figure 3: Illustration of \( A(p,y) \), \( B(y) \), and \( C(y) \). \( A(a,y) = \{b\} \), \( A(b,y) = \emptyset \), \( A(c,y) = \{r\} \), \( A(d,y) = \{b,c\} \), \( A(e,y) = \emptyset \) and \( A(f,y) = \{c\} \). \( B(y) = \{a,c,d,f\} \) and \( C(y) = \{e\} \).](image)

**Lemma 3.** Let \( G = (P,E) \) be a rooted connected geometric graph such that for each \( p \in P \setminus \{r\} \), \( y(p) \neq 0 \). Then, \( G \) is rooted \( y \)-monotone if and only if \( |B(y)| + |C(y)| = |P| - 1 \).

**Proof.** We first prove the \((\Rightarrow)\) direction. We prove that each \( p \in P \setminus \{r\} \) is included to exactly one of \( B(y) \) and \( C(y) \) which implies that \( |B(y)| + |C(y)| = |P| - 1 \).
Each \( p \in P \setminus \{r\} \) is connected with \( r \) by a \( y \)-monotone path, hence there exists a point \( q \in P \setminus \{p\} \) on the same side with \( p \) w.r.t. the \( x \) axis that is adjacent to \( p \) and \(|y(q)| \leq |y(p)|\). If \(|y(q)| < |y(p)|\), then \( A(p, y) \neq \emptyset \) and hence \( p \in B(y) \). Otherwise, there exists exactly one point \( q \in P \setminus \{r, p\} \) that is adjacent to \( p \) and on the same side with \( p \) w.r.t. the \( x \) axis and \(|y(q)| = |y(p)|\). For such a point \( q \) it holds that \( y(q) = y(p) \) and hence \( p \) belongs to \( C(y) \). The \((\leftarrow)\) direction can be easily proved by induction on the number of points. \( \square \)

**Remark 3.** If there exists a point \( p \in P \setminus \{r\} \) with \( y(p) = 0 \) then \( G \) is rooted \( y \)-monotone if and only if (i) \( p \) is connected with \( r \) and (ii) \(|B(y)| + |C(y)| = |P| - 2\).

So, if we know \( B(y), C(y) \) and whether there exists a point \( p \in P \setminus \{r\} \) with \( y(p) = 0 \) connected to \( r \), we can decide if \( G \) is rooted \( y \)-monotone.

**Observation 4.** Let \( y' \) be an axis. If we rotate the \( y' \) counterclockwise, then \( B(y'), C(y') \) and the points \( p \in P \setminus \{r\} \) with \( y'(p) = 0 \), change only when the \( y' \) axis reaches (or moves away from) a line that is perpendicular to an edge of \( G \) or that is perpendicular to a straight line connecting \( r \) with another point of \( P \setminus \{r\} \).

Using similar arguments with the ones employed for solving the rooted UMMST problem, we define a set of critical slopes and appropriate axes which we have to test in order to decide if the graph is rooted-monotone. Let \( \Theta = \{ \theta \in [0, \pi) : \theta \) is a slope of a line perpendicular to either an edge of \( G \) or to a straight line passing through \( r \) and another point of \( P \} \) and \( S[\Theta] \) be the sorted sequence that contains the slopes of \( \Theta \) in increasing order, i.e. \( S[\Theta] = (\theta_0, \theta_1, \ldots, \theta_{m-1}) \), \( \theta_i < \theta_{i+1} \), \( i = 0, 1, \ldots, m-2 \) and \( m < |E| + |P| \). We define the critical set of slopes, \( \Theta_{\text{critical}} = \{ \theta_0, \theta_1, \ldots, \theta_{m-1}\} \cup \{\theta_0 + \theta_1, \theta_1 + \theta_2, \ldots, \theta_{m-2} + \theta_{m-1}, \theta_{m-1} + \pi\} \).

We now assign “names” to the axes with slope in \( \Theta_{\text{critical}} \). Let \( y_i \) be the axis with slope \( \theta_i \), \( 0 \leq i \leq m - 1 \). Moreover, let \( y_{2i+1} \) be the axis with slope \( \theta_i + \frac{\theta_{i+1}}{2} \), \( 0 \leq i \leq m - 2 \) and \( y_{2m-1} \) be the axis of slope \( \frac{\theta_{m-1} + \pi}{2} \). We note that \( C(y_{2i+1}) \) is empty for each \( i = 0, 1, \ldots, m - 1 \), since no two adjacent points of \( P \) have the same projection onto \( y_{2i+1} \).

**Remark 4.** The axes with slopes in \( \Theta_{\text{critical}} \) can be computed in \( O(|E| \log |P|) \) time.

**Lemma 4.** \( G \) is rooted uniform monotone if and only if it is rooted \( y' \)-monotone for some \( y' \) axis of slope in \( \Theta_{\text{critical}} \).

Algorithm [3] shown below, is a rotational sweep algorithm for testing whether \( G \) is a rooted uniform monotone geometric graph. It rotates a \( y' \) axis which initially coincides with \( y_0 \) until it becomes opposite to the \( x \) axis. Throughout this rotation, it checks if \( G \) is rooted \( y' \)-monotone. If \( G \) was not rooted \( y' \)-monotone to start with, i.e. when \( y' = y_0 \), then it may become rooted \( y' \)-monotone only when \( y' \) reaches (or moves away from) a line with slope in \( \Theta \) (see Observation [4]). Taking into account Lemma [4], our algorithm only needs to test if \( G \) is rooted \( y_i \)-monotone for each \( i = 0, 1, \ldots, 2m - 1 \).

**Theorem 6.** Let \( G = (P, E) \) be a rooted connected geometric graph. Then, we can decide in \( O(|E| \log |P|) \) time if it is rooted uniform monotone.
Algorithm 3 root uniform monotone

Input: A rooted connected geometric graph $G = (P, E)$.

Output: The axis of monotonicity if it exists, otherwise, null.

1: axis ← null.
2: compute the axes $y_0, y_1, \ldots, y_{2m-1}$ with slopes in $\Theta_{\text{critical}}$.
3: if $G$ is rooted $y_0$-monotone then axis ← $y_0$.
4: for $i$ ← 1 to $2m - 1$ do
5: \hspace{1cm} if $G$ is rooted $y_i$-monotone then axis ← $y_i$.
6: return axis.

Proof. By Lemma 3, it is immediate that Algorithm 3 decides if $G$ is rooted uniform monotone. We now show that its time complexity is $O(|E| \log |P|)$. Computing the axes $y_0, y_1, \ldots, y_{2m-1}$, with slope in $\Theta_{\text{critical}}$ takes $O(|E| \log |P|)$ time (see Remark 4). Let $k_i$ be the number of pairs of points of $P$ connected by an edge perpendicular to $y_{2i}, 0 \leq i \leq m - 1$. Then, $\sum_{i=0}^{m-1} k_i = |E|$. For each $i = 0, 1, \ldots, m - 1$, we construct a list $L_i$ containing the $k_i$ pairs of points of $P$ that are connected by an edge perpendicular to $y_{2i}$. All $L_i, 0 \leq i \leq m - 1$, can be computed in $O(|E| \log |P|)$ time.

Let $y_i$ be the last axis taken into account. Our algorithm maintains for each $p \in P \setminus \{r\}$ a data structure $A(p)$ which represents the set $A(p, y_i)$ (which is a subset of the $\text{Adj}(p)$). $A(p)$ contains the indices of the points of $P$ that belong to $A(p, y_i)$. $A(p)$ can be implemented by any data structure which supports insert, delete and retrieve operations in $O(\log |P|)$ time (e.g. a 2–3 tree). Our algorithm also maintains the data structure $B$ that represents the $B(y_i)$ which can be implemented as an array of boolean with size $O(|P|)$. In this way, each insert and remove operation by the data structure $B$ takes $O(1)$ time.

Let $p$ be a point in $P \setminus \{r\}$ then computing the $A(p)$ s.t. $A(p)$ equals to $A(p, y_0)$ takes $O(|\text{Adj}(p)| \log |P|)$ time since all the points $q$ adjacent to $p$ are checked and if $q$ belongs to the same half plane with $p$ w.r.t. the $x$ axis and $|y(q)| < |y(p)|$ then $q$ is inserted into $A(p)$ in $O(\log |P|)$ time. Hence, computing all $A(p), p \in P \setminus \{r\}$, takes $O(|E| \log |P|)$ total time. Given each $A(p), p \in P \setminus \{r\}$, computing $B$ s.t. $B$ equals to $B(y_i)$ takes $O(|P|)$ time, e.g. $O(1)$ time to check if $A(p) \neq \emptyset$ for each $p \in P \setminus \{r\}$.

When we move from the $y_{i-1}$ to the $y_i$ axis, the necessary updates we need to make s.t. for each $p \in P \setminus \{r\}$, $A(p)$ represents the set $A(p, y_i)$ and $B$ represents the $B(y_i)$ take:

1. $O(k_{\frac{i}{2}} \log |P|)$ time if $i$ is even and there is no point $p \in P \setminus \{r\}$ with the same projection with $r$ onto $y_i$
2. $O(k_{\frac{i}{2}} \log |P|)$ time if $i$ is odd and there is no point $p \in P \setminus \{r\}$ with the same projection with $r$ onto $y_{i-1}$
3. $O((|\text{Adj}(q)| + k_{\frac{i}{2}}) \log |P|)$ time if $i$ is even and $y_i$ is perpendicular to the straight line passing through $r$ and the point $q \in P \setminus \{r\}$
4. $O((|\text{Adj}(q)| + k_{\frac{i}{2}}) \log |P|)$ time if $i$ is odd and $y_{i-1}$ is perpendicular to the straight line passing through $r$ and the point $q \in P \setminus \{r\}$
Cases (1),(2),(3) and (4) are proved similarly with the Cases (1),(2),(3) and (4) in Theorem 5.

We also note that given $B(y_i)$ and each $A(p, y_i), p \in P \setminus \{r\}$, then computing $C(y_i), 0 \leq i \leq m - 1$ takes $O(k_i)$ time, using the list $L_i$. Furthermore, if we have both $B$ equal to $B(y_i)$ and $C$ equal to $C(y_i)$ and know if $y_i$ is vertical to some straight line passing through $r$ and another point $p$ of $P$ with $pr \in E$, then in $O(1)$ time we can test if $G$ is $y_i$-rooted-monotone (see Lemma 6).

From the previous, it follows that the running time of the algorithm is $O(|E| \log |P|)$.

We note that the approach we took for deciding if a given rooted connected geometric graph is rooted uniform monotone has some similarities with the approach employed in the third section of [5].

5 The Rooted Uniform 2D-Monotone Minimum Spanning Tree (rooted 2D-UMMST) Problem

In this section we study monotonicity w.r.t. two perpendicular axes. Our treatment is analogous to that of Section 3 and Section 4.

We first study the case where the perpendicular axes are given, i.e. they are the $x$ and $y$ axes.

In analogy with Observation 1 we obtain the following Observation.

Observation 5. Let $P$ be a rooted point set and $G = (P, E)$ be a rooted $xy$-monotone spanning graph of $P$ and let $pp' \in E$ where either $p$ and $p'$ lie on different quadrants of the plane or $|x(p)| - |x(p')|, |y(p)| - |y(p')| < 0$. Then, every path from $r$ to a point $q \in P \setminus \{r\}$ that traverses $pp'$ is not $xy$-monotone.

Corollary 3. Let $P$ be a rooted point set and $G^opt$ be the rooted $xy$-monotone minimum spanning graph of $P$. Let $p$ and $q$ be points of $P$ that do not lie on the same quadrant of the plane, then $G^opt$ does not contain the edge $pp'$.

Lemma 5. Let $P$ be a rooted point set and $G^opt$ be the rooted $xy$-monotone minimum spanning graph of $P$. Let $G^opt_{x \geq 0, y \geq 0}$, $G^opt_{x < 0, y \geq 0}$ and $G^opt_{x \geq 0, y < 0}$ be the rooted $xy$-monotone minimum spanning graph of $P_{x \geq 0, y \geq 0}$, $P_{x \geq 0, y < 0}$, $P_{x \leq 0, y \geq 0}$ and $P_{x \geq 0, y \leq 0}$, respectively. Then, $G^opt$ is $G^opt_{x \geq 0, y \geq 0} \cup G^opt_{x \geq 0, y < 0} \cup G^opt_{x \leq 0, y \geq 0} \cup G^opt_{x \leq 0, y \leq 0}$.

Let $P$ be a rooted point set confined to one quadrant of the plane. Then, we define $S[P, y, x]$ to be the sequence that consists of the points of $P$, such that the points in $S[P, y, x]$ are ordered w.r.t. their absolute $y$ coordinates and if two points have the same absolute $y$ coordinate, then they are ordered w.r.t. their absolute $x$ coordinates. More formally, $S[P, y, x] = \{r = p_0, p_1, p_2, \ldots, p_n\}$ such that $\sum_{i=0}^{n}|y(p_i)| \leq |y(p_1)| \leq |y(p_2)| \leq \ldots \leq |y(p_n)|$ and $|y(p_i)| = |y(p_{i+1})|$ implies that $|x(p_i)| < |x(p_{i+1})|$ and $P = \{p_0, p_1, p_2, \ldots, p_n\}$. Using similar arguments with the proof of Theorem 1 we obtain a characterization of the rooted $xy$-monotone minimum spanning graph of $P$.

Lemma 6. Let $P$ be a rooted point set confined to one quadrant of the plane, $S[P, y, x] = \{r = p_0, p_1, p_2, \ldots, p_n\}$ and $G$ be a geometric graph with vertex set $P$. Then, $G$ is the rooted $xy$-monotone minimum spanning graph of $P$ if and
only if (i) \( p_n \) is connected only with the point \( p_j \) such that \( |x(p_j)| \leq |x(p_n)| \) and \( d(p_n, p_j) = d(p_n, P_{\{x(p_n)\}} \setminus \{p_n\}) \) and (ii) \( G \setminus \{p_n\} \) is the rooted \( xy \)-monotone minimum spanning graph of \( P \setminus \{p_n\} \).

Lemma 5 and Lemma 6 lead to the next Corollary.

**Corollary 4.** The rooted \( xy \)-monotone minimum spanning graph of a rooted point set \( P \) is a geometric tree.

**Theorem 7.** The rooted \( xy \)-MMST of a rooted point set \( P \) can be computed in \( O(|P|^2) \) time.

**Proof.** Let \( Q \) be an arbitrary rooted point set confined to one quadrant of the plane and \( S(Q,y,x) = (r = q_0,q_1,q_2,\ldots,q_n) \). For each point \( q_i, i = 1,2,\ldots,n \), \( \text{par}(q_i) \) denotes the closest point to \( q_i \) that belongs to \( \{q_0,q_1,\ldots,q_{i-1}\} |x| \leq |x(q_i)| \).

By Lemma 5 connecting each point \( q_i \) with \( \text{par}(q_i), i = 1,2,\ldots,n \), by a straight line segment produces the rooted \( xy \)-MMST of \( Q \). This procedure clearly takes \( O(|Q|^2) \) time.

We now show how to construct the rooted \( xy \)-MMST of \( P \). We initially construct \( T_{x\leq0,y\leq0}, T_{x\geq0,y\geq0}, T_{x\leq0,y\geq0} \) and \( T_{x\geq0,y\leq0} \) and in \( O(|P|^2) \) time we obtain the trees \( T_{x\leq0,y\leq0}^\text{opt}, T_{x\geq0,y\geq0}^\text{opt}, T_{x\leq0,y\geq0}^\text{opt} \) and \( T_{x\geq0,y\leq0}^\text{opt} \) respectively. Finally, we return the union of \( T_{x\leq0,y\leq0}^\text{opt}, T_{x\geq0,y\geq0}^\text{opt}, T_{x\leq0,y\geq0}^\text{opt} \) and \( T_{x\geq0,y\leq0}^\text{opt} \) which by Lemma 5 is the rooted \( xy \)-MMST of \( P \).

Using the same reduction that we used in Theorem 3 i.e. the reduction from sorting given by Shamos [23], we obtain a lower bound for the time complexity of every algorithm which solves the rooted \( xy \)-MMST problem.

**Theorem 8.** Any algorithm which, given a rooted point set \( P \), produces the rooted \( xy \)-MMST of \( P \) requires \( \Omega(|P|\log|P|) \) time.

**Theorem 9.** Let \( G = (P,E) \) be a rooted connected geometric graph. Then, we can decide in \( O(|E|) \) time if \( G \) is rooted \( xy \)-monotone.

**Proof.** Our proof is similar with the proof of Theorem 3 We transform \( G \) to a directed graph \( \overrightarrow{G} \) in \( O(|E|) \) time as follows. Let \( \overrightarrow{pq} \) be an edge of \( G \). If \( p \) and \( q \) lie on the same quadrant of the plane and \( |x(p)| \leq |x(q)| \) and \( |y(p)| \leq |y(q)| \) (resp. \( |x(q)| \leq |x(p)| \) and \( |y(q)| \leq |y(p)| \)) we direct \( \overrightarrow{pq} \) from \( p \) to \( q \) (resp. from \( q \) to \( p \)). By observation 5 it follows that all the other edges cannot be traversed by a \( xy \)-monotone path connecting \( r \) with another point of \( P \). Hence, we remove them. Then, \( G \) is rooted \( xy \)-monotone if and only if \( r \) is connected with all other points of \( P \) in \( \overrightarrow{G} \). We decide the latter in \( O(|E|) \) time by applying a depth first search traversal.

We now study the problem of computing the rooted 2D-UMMST of a given rooted point set \( P \).

**Observation 6.** Let \( x', y' \) and \( x'', y'' \) be axes of two different Cartesian Systems s.t. \( x'' \) (resp. \( y'' \)) forms with \( x' \) (resp. \( y' \)) a counterclockwise angle equal to \( k \pi, k = 1,2,3 \). Then, the rooted \( x'y' \)-MMST of \( P \) coincides with the rooted \( x''y'' \)-MMST of \( P \).
Observation 7. Let \( x'y' \) be a Cartesian System. If we rotate the Cartesian System counterclockwise, then the rooted \( x'y' - \text{MMST} \) of \( P \) changes only when the \( y' \) axis reaches (or moves away from) a straight line that is perpendicular or parallel to a straight line passing through two points of \( P \).

Proof. This is true, since when the \( y' \) axis reaches a straight line that is perpendicular or parallel to a straight line passing through two points of \( P \), say \( p \) and \( q \), then it might become feasible (while previously this was not feasible) to connect \( p \) and \( q \), with the straight line segment \( pq \) such that \( p \) or \( q \) traverses \( pq \) in a \( x'y' \)-monotone path from it to \( r \). Similarly, when the \( y' \) axis moves away from a straight line that is perpendicular or parallel to a straight line passing through \( p \) and \( q \), then the straight line segment \( pq \), which was previously traversed by the \( xy' \)-monotone path from \( p \) or \( q \) to \( r \), now might not be feasible to be traversed in any \( xy' \)-monotone path from any point to \( r \).

Based on the previous Observations, we define the set \( \Theta = \{ \theta \in [0, \frac{\pi}{2}) : \text{a line of slope } \theta \text{ is either perpendicular or parallel to a straight line connecting two points of } P \} \). Let \( S[\Theta] \) be the sorted sequence that contains the slopes in \( \Theta \) in increasing order, i.e. \( S[\Theta] = (\theta_0, \theta_1, \ldots, \theta_{m-1}) \), \( \theta_1 < \theta_{i+1}, 0 \leq i < m - 2 \) and \( m \leq \binom{|P|}{2} \). Then, we define the critical set of slopes \( \Theta_{\text{critical}} = \{ \theta_0, \theta_1, \ldots, \theta_{m-1} \} \cup \{ \frac{\pi + \theta_1}{2}, \frac{\pi + \theta_2}{2}, \ldots, \frac{\pi + \theta_m}{2}, \frac{\pi - \theta_{m-1}}{2} \} \). We now “name” the Cartesian Systems such that their vertical axis has slope in \( \Theta_{\text{critical}} \). More formally, let \( x_0y_0, x_1y_1, \ldots, x_{2m-1}y_{2m-1} \) be the Cartesian Systems such that \( y_i \) has slope \( \theta_i, 0 \leq i \leq m - 1 \), \( y_{2i+1} \) has slope \( \frac{\pi + \theta_{i+1}}{2}, 0 \leq i \leq m - 2 \) and \( y_{2m-1} \) has slope \( \frac{\pi - \theta_{m-1}}{2} \).

Remark 5. The Cartesian Systems \( x_0y_0, x_1y_1, \ldots, x_{2m-1}y_{2m-1} \) are computed in \( O(|P|^3 \log |P|) \) time.

In analogy with Lemma 2, we obtain the following Lemma.

Lemma 7. The rooted 2D-UMMST of \( P \) is one of the solutions of the rooted \( x_iy_i - \text{MMST} \) problem with input \( P \) for some \( i = 0, 1, 2, \ldots, 2m - 1 \), and more specifically the one of minimum cost.

Theorem 10. Let \( P \) be a rooted point set. Then, we can produce the rooted 2D-UMMST of \( P \) in \( O(|P|^2 \log |P|) \) time.

Proof. We give our algorithm which produces the rooted 2D-UMMST of \( P \). This algorithm is a rotational sweep algorithm analogous with the Rooted UMMST given in Section 4. It rotates a Cartesian System \( x'y' \) which initially coincides with \( x_0y_0 \) until it coincides with the Cartesian System \( xy \), i.e. the given Cartesian System. Throughout this rotation, it updates the rooted \( x'y' \)-MMST of \( P \). By Lemma 7, we only need to compute the rooted \( x_1y_1 - \text{MMST} \) of \( P \) for the Cartesian Systems \( x_iy_i, i = 0, 1, \ldots, 2m - 1 \). Hence, the algorithm is restated as follows. The algorithm initially computes the Cartesian Systems \( x_0y_0, x_1y_1, \ldots, x_{2m-1}y_{2m-1} \) in \( O(|P|^2 \log |P|) \) time (see Remark 5). Let \( T^\text{opt}_{1} \) be the rooted \( x_1y_1 - \text{MMST} \) of \( P \). Our algorithm creates \( T^\text{opt}_{i} \) using Theorem 7. Then, it iterates for \( i = 1, 2, \ldots, 2m - 1 \), obtaining \( T^\text{opt}_{1}, T^\text{opt}_{2}, \ldots, T^\text{opt}_{2m-1} \) in this order. Throughout its execution it stores the Cartesian System \( \text{minX} \text{minY} \) in which it encountered the minimum cost solution found so far. Finally, it returns the rooted \( \text{minX} \text{minY} \)-MMST of \( P \) employing the algorithm described in Theorem 4.
From Lemma 7, the algorithm previously described produces the rooted 2D-UMMST of $P$. We only need to analyze its time complexity. We use similar data structures $PD(p), p \in P \setminus \{r\}$ with the ones employed in Theorem 5. More specifically, for each point $p \in P \setminus \{r\}$ the data structure $PD(p)$ which is a balanced binary tree contains the pairs $(q, d(p, q))$ for all the points $q$ such that $p$ can be traversed in a $x_i y_i$-monotone path from $p$ to $r$, where $i$ is the index of the current iteration of our algorithm. Then, using similar arguments with the arguments employed in Theorem 5, the time complexity of the algorithm is $O(|P|^2 \log |P|)$.

We now study the problem of recognizing if a given rooted connected geometric graph $G = (P, E)$ is rooted uniform 2D-monotone. Our approach is analogous to the approach we took for recognizing rooted uniform monotone graphs in Section 4.

For each $p \in P \setminus \{r\}$ let $A(p, x, y)$ be the set $\{q : q \in \text{Adj}(p) \text{ and } q \text{ lies on the same quadrant of the plane with } p \text{ and } |x(q)| \leq |x(p)| \text{ and } |y(q)| \leq |y(p)|\}$. Let $B(x, y)$ be the set $\{p : p \in P \setminus \{r\} \text{ and } A(p, x, y) \neq \emptyset\}$. Then, similarly with Lemma 5 we obtain the following Lemma.

**Lemma 8.** $G$ is rooted $xy$-monotone if and only if $|B(x, y)|$ equals to $|P| - 1$.

**Theorem 11.** Given a rooted connected geometric graph $G = (P, E)$, we can decide in $O(|E| \log |P|)$ time if $G$ is rooted uniform 2D-monotone.

**Proof.** The proof is similar to the proof of Theorem 5. If we rotate a Cartesian System $x' y'$ counterclockwise then the sets $A(p, x', y'), p \in P \setminus \{r\}$ and $B(x', y')$ change possibly only when the $y'$ axis reaches (or moves away from) a line perpendicular or parallel to an edge of $G$ or when the $x'$ axis reaches (or moves away from) a line perpendicular or parallel to a straight line connecting $r$ with another point of $P$. Hence, using similar arguments with the ones employed throughout this article, we need to take into account only the Cartesian Systems $x_0 y_0, x_1 y_1, \ldots, x_{2m-1} y_{2m-1}, m < |E| + |P|$ such that $y_0, y_2, \ldots, y_{2m-2}$, are all the axes that are either (i) perpendicular or parallel to some edge of $E$ or (ii) perpendicular or parallel to some straight line connecting $r$ with another point of $P$. The slope of each $y_{2i}$ is $\theta_i, 0 \leq i \leq m - 1$ and it holds that $0 \leq \theta_0 < \theta_1 < \ldots < \theta_{m-1} < \frac{\pi}{2}$. Moreover, the slope of each $y_{2i+1}$ is equal to $\frac{\theta_i + \pi}{4}, i = 0, 1, \ldots, m - 2$ and the slope of $y_{2m-1}$ is equal to $\frac{\pi}{4} + \frac{\theta_{m-1}}{2}$. We can compute these axes in $O(|E| \log |P|)$ time. Moreover, similarly with Lemma 4 $G$ is rooted uniform 2D-monotone if and only if it is rooted $x_i y_i$-monotone for some Cartesian System $x_i y_i, i = 0, 1, \ldots, 2m - 1$.

We now employ a rotational sweep algorithm that decides if $G = (P, E)$ is rooted uniform 2D-monotone. Based on the previously mentioned facts, the algorithm decides if $G$ is rooted uniform 2D-monotone by testing if $G$ is rooted $x_i y_i$-monotone for some $i = 0, 1, \ldots, 2m - 1$. It tests that in this order, i.e. it first checks $x_0 y_0$ then $x_1 y_1, \ldots$, and in the end it checks $x_{2m-1} y_{2m-1}$. The algorithm maintains for each $p \in P \setminus \{r\}$ a data structure $A(p)$ which represents the $A(p, x, y)$ (when the algorithm checks the $x_i y_i$ Cartesian System) and is implemented as a $2 - 3$ tree that stores the indices of the points that it contains. Moreover, the algorithm maintains a data structure $B$ that represents the $B(x_i, y_i)$ (when the algorithm checks the $x_i y_i$ Cartesian System) and
is implemented as an array of boolean. Using similar analysis with the one presented in Theorem 6, the initial construction of all $A(p), p \in P \setminus \{r\}$ s.t. $A(p)$ equals to $A(p, x_0, y_0)$ takes $O(|E| \log |P|)$ total time. Then, the construction of $B$ s.t. $B$ equals to $B(x_0, y_0)$ takes $O(|P|)$ time. Furthermore, the updates of all $A(p), p \in P \setminus \{r\}$, and $B$ throughout all the execution of the algorithm take $O(|E| \log |P|)$ total time. Finally, from Lemma 8, given $B$ equal to $B(x_i, y_i)$, it can be decided in $O(1)$ time if $G$ is rooted $x_i, y_i$-monotone. From all the previous, it follows that the time complexity of the algorithm is $O(|E| \log |P|)$. □

6 Conclusions and Future Work

In this paper we studied the problem of constructing the geometric minimum spanning graph of rooted point sets that is monotone w.r.t. a single direction ($y$-monotonicity) or w.r.t. a pair of orthogonal directions ($xy$-monotonicity). For both cases, we showed that the minimum spanning graph is actually a tree and we presented algorithms to construct it for the case where the directions of monotonicity are given or remain to be determined.

Several directions for further research are opened.

1. The $O(|P|^2)$ time complexity of the algorithm that constructs the rooted $y$-MMST is due to the computation of the distance between each pair of points. We anticipate that the time complexity can be reduced to $O(|P| \log |P|)$ by a dynamic version of Fortune’s plane sweep algorithm for constructing Voronoi diagrams [8].

2. We have studied rooted point sets and we have built minimum spanning trees that contain a monotone path w.r.t. a single direction from the root $r$ to any other point in the point set. What about the case where we are given a $k$-rooted point set, i.e., a set with $k$ designated points as its roots and we are asked to find a minimum spanning graph containing monotone paths from each root to every other point in the point set. In this case it is clear that the wanted graph is not a tree. Note that in the extreme case where all points in the point set are designated as roots the problem is trivial. The wanted graph is actually the path visiting all points in sorted order w.r.t. the direction of monotonicity.

3. We studied the problem of building rooted minimum cost spanning graphs that possess a specific property, and we focused on the property of monotonicity (in one or two orthogonal directions). What if we consider a different requirement/property? For example, we can ask for a minimum spanning graph containing increasing-chord paths or self-approaching paths (see [15][1]) from the root to any other point in the point set. In this case, it is most likely that the sought graph is not a tree.

References

[1] S. Alamdari, T. M. Chan, E. Grant, A. Lubiw, and V. Pathak. Self-approaching graphs. In W. Didimo and M. Patrignani, editors, Graph Drawing - 20th International Symposium, GD 2012, Redmon, WA, USA,
[2] P. Angelini. Monotone drawings of graphs with few directions. In 6th International Conference on Information, Intelligence, Systems and Applications, IISA 2015, Corfu, Greece, July 6-8, 2015, pages 1–6. IEEE, 2015.

[3] P. Angelini, E. Colasante, G. Di Battista, F. Frati, and M. Patrignani. Monotone drawings of graphs. J. Graph Algorithms Appl., 16(1):5–35, 2012.

[4] P. Angelini, W. Didimo, S. Kobourov, T. Mchedlidze, V. Roselli, A. Symvonis, and S. Wismath. Monotone drawings of graphs with fixed embedding. Algorithmica, 71(2):233–257, 2015.

[5] E. M. Arkin, R. Connelly, and J. S. B. Mitchell. On monotone paths among obstacles with applications to planning assemblies. In K. Mehlhorn, editor, Proceedings of the Fifth Annual Symposium on Computational Geometry, Saarbrücken, Germany, June 5-7, 1989; SCG ’89, pages 334–343. ACM, 1989.

[6] H. R. Dehkordi, F. Frati, and J. Gudmundsson. Increasing-chord graphs on point sets. J. Graph Algorithms Appl., 19(2):761–778, 2015.

[7] S. Felsner, A. Igamberdiev, P. Kindermann, B. Klemz, T. Mchedlidze, and M. Scheucher. Strongly monotone drawings of planar graphs. CoRR, abs/1601.01598, 2016.

[8] S. Fortune. A sweepline algorithm for voronoi diagrams. Algorithmica, 2:153–174, 1987.

[9] L. J. Guibas and J. Stolfi. On computing all north-east nearest neighbors in the $l_1$ metric. Information Processing Letters, 17(4):219 – 223, 1983.

[10] D. He and X. He. Nearly optimal monotone drawing of trees. Theoretical Computer Science, pages –, 2016.

[11] D. He and X. He. Optimal monotone drawings of trees. CoRR, abs/1604.03921, 2016.

[12] X. He and D. He. Monotone drawings of 3-connected plane graphs. In N. Bansal and I. Finocchi, editors, Algorithms - ESA 2015 - 23rd Annual European Symposium, Patras, Greece, September 14-16, 2015, Proceedings, volume 9294 of LNCS, pages 729–741. Springer Berlin Heidelberg, 2015.

[13] Md. I. Hossain and Md. S. Rahman. Monotone grid drawings of planar graphs. In J. Chen, J. E. Hopcroft, and J. Wang, editors, Frontiers in Algorithmics - 8th International Workshop, FAW 2014, Zhangjiajie, China, June 28-30, 2014. Proceedings, volume 8497 of LNCS, pages 105–116. Springer, 2014.

[14] F. K. Hwang. An $O(\sqrt{n}\log n)$ algorithm for rectilinear minimal spanning trees. J. ACM, 26(2):177–182, 1979.
[15] C. Icking, R. Klein, and E. Langetepe. Self-approaching curves. Mathematical Proceedings of the Cambridge Philosophical Society, 125:441–453, January 1999.

[16] R. Jothi and B. Raghavachari. Approximation algorithms for the capacitated minimum spanning tree problem and its variants in network design. ACM Trans. Algorithms, 1(2):265–282, October 2005.

[17] P. Kindermann, A. Schulz, J. Spoerhase, and A. Wolff. On monotone drawings of trees. In C. A. Duncan and A. Symvonis, editors, Graph Drawing - 22nd International Symposium, GD 2014, Würzburg, Germany, September 24-26, 2014, Revised Selected Papers, volume 8871 of LNCS, pages 488–500. Springer-Verlag Berlin Heidelberg, 2014.

[18] D. G. Larman and P. McMullen. Arcs with increasing chords. Mathematical Proceedings of the Cambridge Philosophical Society, 72:205–207, September 1972.

[19] K. Mastakas and A. Symvonis. On the construction of increasing-chord graphs on convex point sets. In 6th International Conference on Information, Intelligence, Systems and Applications, IISA 2015, Corfu, Greece, July 6-8, 2015, pages 1–6. IEEE, 2015.

[20] M. Nöllenburg, R. Prutkin, and I. Rutter. On self-approaching and increasing-chord drawings of 3-connected planar graphs. Journal of Computational Geometry, 7(1):47–69, 2016.

[21] F. P. Preparata and M. I. Shamos. Computational Geometry: An Introduction. Texts and monographs in computer science. Springer-Verlag, 1988.

[22] G. Rote. Curves with increasing chords. Mathematical Proceedings of the Cambridge Philosophical Society, 115:1–12, January 1994.

[23] M. I. Shamos. Computational Geometry. PhD thesis, Yale University, USA, 1978.

[24] M. I. Shamos and D. Hoey. Closest-point problems. In 16th Annual Symposium on Foundations of Computer Science, Berkeley, California, USA, October 13-15, 1975, pages 151–162. IEEE Computer Society, 1975.