Fitting by monotone orthogonal chains

J.M. Díaz-Báñez* M. A. López† H. Pérez-Roses‡ C. Seara§ I. Ventura¶

Abstract

In this paper, we study the problem of fitting a point set by a monotone orthogonal polygonal chain, i.e., a chain consisting of few consecutive orthogonal line segments, and finding the best direction for that fitting. We also extend the problem to the three-dimensional space where the chain is interpreted as a connected configuration of orthogonal planes.

1 Introduction

Fitting a curve of a certain type to a given set of points in the plane is a fundamental problem in many fields as statistics, computer graphics, or artificial intelligence. A special case is the so called *the polygonal approximation* or *the polygonal fitting problem*, where a polygonal chain with k corners or joints is fitted to a data set. The best configuration is the polygonal chain which minimize the error of approximation. In many applications, the error is defined as the maximum vertical distance of any input point from the polygonal chain (called the *Chebyshev error*). The following problem has been widely studied.

Min-Max problem: *Given k, find an approximating polygonal curve minimizing the error among those with a corner number not greater than k.*

The Min-Max problem with the vertical distance was first posed in [11]. They gave a $O(n^2 logn)$ time algorithm for this problem, and the complexity has been improved to $O(n^2)$ [20] and then to $O(n \log n)$ [9]. Notice that these problems are closely related to approximating piecewise linear curves by more simple ones, in which the input is a polygonal chain with n edges rather than a set of n points. This question arises in cartography, pattern recognition, and graphic design [16, 5, 7], and has received much attention in computational geometry [13, 18, 6, 10].

In this paper, we consider the case in which the points are fitted by an *monotone orthogonal polygonal chain with respect to an orientation*, that is, a chain of consecutive orthogonal line segments such that the extreme segments are half lines with the same slope. The case in which the slope is given, the problem becomes the *Min-Max rectilinear fitting problem* solved in $O(n^2 \log n)$ time [8]. See [19] and [15] for more recent results. In this paper, we focus on fitting orthogonal chains consisting of few segments and then we address the problem of finding the best direction for fitting a monotone orthogonal polygonal chain. We also extend the problem to the three-dimensional space where the chain is interpreted as a connected configuration of orthogonal planes.

We start by introducing some notation. Let $S = \{p_1, p_2, \ldots, p_n\}$ be a set of points in the plane, where $p_i = (x_i, y_i)$. A *k-orthogonal chain* $\mathcal{O}$, $k \geq 1$, is a chain of $2k-1$ consecutive orthogonal segments (links) such that the extreme segments are in fact half lines with the same slope. The *orientation* θ of a k-orthogonal chain is the angle formed by their extreme half lines with the x-axis. Thus, $\mathcal{O}$ consists

*Departamento de Matemática Aplicada II, Universidad de Sevilla, Spain, dbanez@us.es. Partially supported by grants BFM2003-04062 and MTM2006-03909.

†Department of Computer Science, University of Denver, 2360 South Gaylord Street Denver, CO 80208, USA.

‡Universidad de Oriente, Santiago de Cuba, Cuba.

§Departament de Matemàtica Aplicada II, Universitat Politècnica de Catalunya, Spain, carlos.seara@upc.edu. Supported by projects MEC MTM2006-01267 and DURSI 2005SGR00692.

¶Departamento de Matemáticas, Universidad de Huelva, Spain, iventura@us.es. Partially supported by grants BFM2003-04062 and MTM2006-03909.

of k segments of slope $\tan(\theta)$ and $k-1$ segments of slope $\tan(\theta+\frac{\pi}{2})$. A k-orthogonal chain is monotone with respect to a given orientation α if every line with slope $\tan(\alpha+\frac{\pi}{2})$ intersects the k-orthogonal chain either in a point or in a segment with orientation $\alpha+\frac{\pi}{2}$ (see Figure 1).

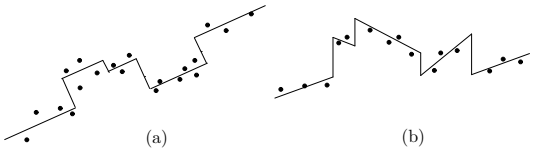

Figure 1: a) A monotone 5-orthogonal chain, b) a monotone 5-non-orthogonal chain.

A k-orthogonal chain is monotone if it is monotone with respect to its orientation. In this paper we deal with the problem of fitting a monotone k-orthogonal chain $\mathcal{O}$ to a set S of n points in the plane. A monotone k-orthogonal chain $\mathcal{O}$ with orientation θ divides the plane into k (at most two possible infinite) strips with orientation $(\theta+\frac{\pi}{2})$ (see Figure 1). Fitting $\mathcal{O}$ to S is to locate k θ-oriented segments $s_i(\theta)$, $i=1\ldots,k$ according to a given optimization criterium. Several reasonable criteria may come to mind. We consider one of them here. Let $l_i(\theta)$ be the line passing through $p_i\in S$ with orientation $\theta+\frac{\pi}{2}$. The fitting distance between p_i and $\mathcal{O}$, denoted by $d_f(p_i,\mathcal{O})$ is given by

$$d_f(p_i,\mathcal{O})=\min_{p\in l_i(\theta)\cap\mathcal{O}} d(p_i,p).$$

Notice that d_f is not the Euclidean distance. However, we can assume that this distance is always the Euclidean distance between p_i and a point on a segment with orientation θ. The *error tolerance* of $\mathcal{O}$ with respect to S, denoted by $\mu(\mathcal{O},S)$, is the maximum fitting distance between the points of S and $\mathcal{O}$, that is,

$$\mu(\mathcal{O},S)=\max_{p_i\in S} d_f(p_i,\mathcal{O}).$$

Definition 1.1. Given a set S of n points in the plane, *the k-fitting problem* consists on finding a monotone k-orthogonal chain such that its error tolerance with respect to S is minimized.

Notice that if the orientation θ of the k-orthogonal chain is fixed, for example $\theta=0$, then the k-fitting problem consists on finding an x-monotone rectilinear path composed of $(2k-1)$ links (segments) with minimum error tolerance where the fitting distance is just the vertical distance as in [8].

2 The oriented fitting problem

We consider the problem of fitting a k-orthogonal monotone chain whose extreme half lines have angle θ. Without loss of generality, we can assume that $\theta=0$. Thus, we are looking for an x-monotone rectilinear path formed by the concatenation of k horizontal segments and $k-1$ vertical segments. In a recent paper [15], Mayster and López studied two types of fitting problems for a set S of n points in the plane: (1) the min-ϵ fitting problem, where the goal is to minimize the error tolerance ϵ for a given number k of horizontal segments (our k-fitting problem, with $\theta=0$) and (2) the min-# fitting problem, where the goal is to minimize the number of horizontal segments for a given allowed error tolerance ϵ. They solve (1) in $O(\min\{n^2,nk\log n\})$ time, and, after linear time preprocessing, they solve an instance of (2) in $O(\min\{k\log n,n\})$ time.

We are interested in the first problem for small values of k, but where the orientation of the chain is a free parameter and, therefore, the segments are not necessarily horizontal and vertical. Nevertheless, in this section we discuss the oriented case, so we assume that the chain is horizontal. By the results of [15], if k is a constant, the running time is $O(n\log n)$. Next we consider the oriented k-fitting problem of a n point set S for some small values of $k\geq 1$. Let $y_{\max}=\max\{y_1,y_2,\ldots,y_n\}$, $y_{\min}=\min\{y_1,y_2,\ldots,y_n\}$, $y^i_{\max}=\max\{y_1,y_2,\ldots,y_i\}$, and $y^i_{\min}=\min\{y_1,y_2,\ldots,y_i\}$; and a similar notation for the x-coordinates of the points in S. In linear time we can compute $x_{\max}$, $x_{\min}$, $y_{\max}$, and $y_{\min}$. Without loss of generality we assume that all the points in S are located in the first quadrant

of the coordinate system and that $x_{\min} = 0$, $y_{\min} = 0$, $x_{\max} = c$, and $y_{\max} = d$. In other words, the rectangle with corners $(0, 0)$ and (c, d) is the bounding box for S.

The oriented 1-fitting problem. The oriented 1-fitting problem corresponds to finding the horizontal line $y = (y_{\max} + y_{\min})/2$. Obviously, once we know $y_{\max}$ and $y_{\min}$, then the optimal line can be computed in constant time. Thus, the case $k = 1$ can be trivially solved in linear time.

The oriented 2-fitting problem. The oriented 2-fitting problem consists on finding two half lines joined by a vertical segment. Next we show a simple $O(n \log n)$ time algorithm for this problem. Assume that S is sorted by abscissas, so that $x_1 \leq \cdots \leq x_n$. Sweeping S from left to right with a vertical line ℓ, we can update in $O(1)$ time the error tolerance $t_i^l = |y_{\max}^i - y_{\min}^i|$, $i = 1, \ldots, n-1$, of the points of S to the left of ℓ. Store these values in an array T_l. Similarly, we can define t_i^r using the reverse sequences of the y-coordinates of the points so that we can sweep S from right to left to compute the error tolerance t_i^r of the points $\{p_{i+1}, \cdots, p_n\}$ to the right of the sweep line. Store these values t_i^r in an array T_r. $\text{Max}\{T_l(i), T_r(i)\}$ provides the cost of splitting the two half lines between p_i and p_{i+1}. Thus, by traversing both arrays simultaneously, we can find the optimal solution $\min_{1 \leq i < n} \max\{T_l(i), T_r(i)\}$ in linear time.

Next we show a little more sophisticated $O(n \log n)$ time algorithm for the same problem which has the advantage that it can be translated and used later in the unoriented fitting problem. The algorithm is as follows. First, in linear time we compute $y_{\max}$, $y_{\min}$, $x_{\max}$, and $x_{\min}$. Also in linear time we can do a translation of the points to the first quadrant of the coordinate system and, without loss of generality, we can assume that $p_1 = (0, y_1)$, $p_n = (c, y_n)$, $p_i = (x_i, y_{\max})$, and $p_j = (x_j, 0)$ (see Figure 2). Let ℓ be the line containing the vertical segment of an optimal solution of the orientated 2-fitting problem which produces a bipartition of S: points on the left of ℓ, and points on the right of ℓ (see Figure 2).

Lemma 2.1. *Line ℓ separates the points with y-coordinates $y_{\min}$ and $y_{\max}$.*

Assume that the point p_i with y-coordinates $y_{\max}$ is on the left of the point p_j with y-coordinate $y_{\min}$, thus by Lemma 2.1 the line ℓ lies in between these two points. We also assume that both points have different x-coordinates because otherwise the solution is trivial. If the point p_i with y-coordinates $y_{\max}$ is on the right of the point p_j with y-coordinate $y_{\min}$, then the algorithm is similar with the unique change in the step 2 by computing the maxima points with respect to the second quadrant and the maxima points with respect to the fourth quadrant and join them forming two staircases.

The oriented 2-fitting algorithm

1. In linear time classify the points of S into: S_1 points with x-coordinate less than x_i, S_2 points with x-coordinate between x_i and x_j, and S_3 points with x-coordinate greater than x_j. In $O(n)$ time compute the point (a_1, b_1) such that $b_1 = \min\{y \mid (x, y) \in S_1\}$ and the point (a_2, b_2) such that $b_2 = \max\{y \mid (x, y) \in S_3\}$.

2. For S_2 in $O(n \log n)$ time compute the maxima points (see [17]) of $S_2 \cup \{(a_2, b_2)\}$ with respect to the first quadrant and the maxima points of $S_2 \cup \{(a_1, b_1)\}$ with respect to the third quadrant and join them forming two staircases as in Figure 2.

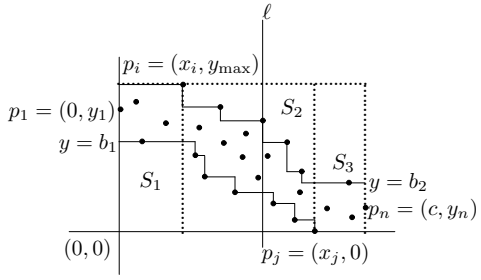

Figure 2: S_1, S_2, S_3, and the maxima points of S_2.

3. Now it is clear that the vertical line $\ell : x = a$ has to be in between the x-coordinates x_i and x_j, and in order to find the right location of the line ℓ we can do a binary search over the points in the two staircases (using their structure) in at most $O(\log n_2)$ time such that we get the best balanced to the left and to the right of the line ℓ, that is,

$$\min_{x_i \leq a < x_j} a \quad \text{subject to} \quad \max_{x_k \leq a, (x_k, y_k) \in S_2} \{y_{\max} - y_k\} \geq \max_{x_m > a, (x_m, y_m) \in S_2} y_m \tag{1}$$

or

$$\min_{x_i < a \leq x_j} a \quad \text{subject to} \quad \max_{x_k \leq a, (x_k, y_k) \in S_2} \{y_{\max} - y_k\} \leq \max_{x_m > a, (x_m, y_m) \in S_2} y_m \tag{2}$$

For at least one of the two equations (1) or (2) there exists a solution. In the case (1) the error tolerance of S is given by the points to the left of the line ℓ and in the case (2) the error tolerance is given by the points to the right of the line ℓ. With an extra $O(1)$ time we can compute this error tolerance given by the half of the difference between the bigger and smaller y-coordinates of the points to the left (or to the right) of the line ℓ.

Theorem 2.2. *The oriented* 2*-fitting problem can be solved in* $O(n \log n)$ *time.*

The oriented 3-fitting problem. The above two staircases structure computed in $O(n \log n)$ time (see Figure 2) can be used for the 3-fitting problem as follows. For this problem by Lemma 2.1 at least one of the two lines has to be in between $y_{\max}$ and $y_{\min}$. Thus either (1) the two vertical lines are in between the $y_{\max}$ and $y_{\min}$ (see Figure 3); or (2) only one vertical line is in between the $y_{\max}$ and $y_{\min}$ and the other is to the left of to the right of $y_{\max}$ or $y_{\min}$.

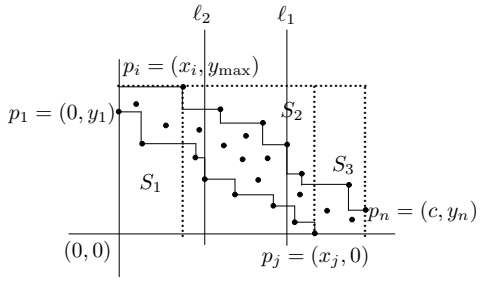

Figure 3: The 3-fitting problem.

In the case (1), the vertical line ℓ_1 has at most $n-2$ possible locations in between the $y_{\max}$ and $y_{\min}$, nevertheless notice that the number of possible locations does not depend on the number of points in S but on the number of points in the staircases which can be smaller than $n-2$. Once we have chosen a position for vertical line ℓ_1, then the second vertical line ℓ_2 can be found in $O(\log n)$ time by a binary search either in between $y_{\max}$ and ℓ_1 or in between ℓ_1 and $y_{\min}$ by trying to minimize the partial values of the tolerance errors just computed and updating the best solution.

In the case (2), once we have fixed the vertical line ℓ_1 (at most $n-2$ possible locations in between the $y_{\max}$ and $y_{\min}$, and as above the possible locations depend on the number of points in the staircases), then either to the right of ℓ_1 or to the left of ℓ_1 there is only a horizontal segment which can be computed in $O(1)$ time using the structure in Figure 2 together with its error tolerance. Assume that the horizontal segment is to the right. Then, we have to find the second vertical line ℓ_2 to the left of ℓ_1. It can be found in $O(\log n)$ time by a binary search using the structure in $O(\log n)$ time trying to minimize the partial value of the tolerance error we have just computed and updating the best solution. Thus, in any case the time complexity is $O(n \log n)$.

Theorem 2.3. *The oriented* 3*-fitting problem can be solved in* $O(n \log n)$ *time.*

Now we would like to extend this procedure to a number k of vertical lines. By Lemma 2.1 there will be al least one vertical line in between $y_{\max}$ and $y_{\min}$. The 4-fitting problem can be solved using the same technique but with more case analysis. The algorithm in [15] solves these problems with an $O(\min\{n^2, nk \log n\})$-time algorithm which is $O(n \log n)$ time for k being a constant. Nevertheless our algorithm is sensitive to the number of points in the staircases, thus if this number is small our algorithm is faster.

3 The unoriented fitting problem

Now we consider the problem of fitting the point set S by an unoriented k-orthogonal monotone chain where the orientation θ of the orthogonal polygonal chain is free.

The unoriented 1-fitting problem. The unoriented 1-fitting problem is equivalence to the problem of computing the width of the set of points S, if we know the width we solve the unoriented 1-fitting problem and vice versa. The width of a set of points S in the plane can be computed in $O(n)$ using the rotating caliper technique if we know the convex hull of S (see [12]), otherwise the problem of computing the width of a set of points in the plane has an $\Omega(n \log n)$ time lower bound (see [14]). Therefore the unoriented 1-fitting problem can be solved optimally in $\Theta(n \log n)$ time

Theorem 3.1. *The unoriented 1-fitting problem can be solved optimally in $\Theta(n \log n)$ time.*

The unoriented 2-fitting problem. An optimal solution for the unoriented 2-fitting problem is determined by a line ℓ splitting S into two parts and by two pairs of points (one per each part) giving their partial errors. We denote by S_l (left) and S_r (right) the subsets in which the separating line ℓ divides S and by e_l^u, e_l^b (resp. e_r^u, e_r^b) the points giving the error tolerance in S_l (resp. S_r). Thus, given a monotone orthogonal 2-polygonal chain $\mathcal{O}$ with orientation θ, the error tolerance is

$$\mu(\mathcal{O}, S) = \max_{p_i \in S} d(p_i, \mathcal{O}) = \max\{d_\theta(e_l^u, e_l^b), d_\theta(e_r^u, e_r^b)\}$$

where $d_\theta(p, q)$ denotes the distance between lines through p and q of orientation θ.

We call left and right antipodal-pairs to the points e_l^u, e_l^b, and e_r^u, e_r^u, respectively. By shifting the separating line ℓ of an optimal orthogonal 3-polygonal chain, we can prove easily the following lemma.

Lemma 3.2. *There always exists a fitting by a monotone and orthogonal 3-polygonal chain for S such that its separating line ℓ pass through a point of S, furthermore the line ℓ separates the points with y-coordinates $y_{\min,\theta}$ and $y_{\max,\theta}$ for each fixed orientation θ.*

We will use Lemma 3.2 to look for an optimal solution. The goal of our approach is to adapt the $O(n \log n)$-time staircases algorithm shown in Section 2 for the oriented case to the changes of the orientation of the optimal 3-polygonal chain. The main idea is to maintain the structure formed by the two staircases (initially formed by horizontal and vertical segments for orientation $\theta = 0$) as we change the orientation θ of the polygonal chain and deleting and updating points in the staircases according to the orientation changes. First notice that the two staircases (formed by horizontal and vertical segments) correspond to the points of S which are $\pi/2$-maxima with respect to the orientation given by the bisector of the first quadrant (one staircase) and the bisector third quadrant (the other staircase), so the difference of the orientation is 180[3]. The goal is to maintain the a list and the order of the points of S which are unoriented $\pi/2$-maxima for some orientation in $\theta \in \mathbb{S}^2$ and analogously for the orientation $\theta + 180$, both forming a staircases pair. In the steps of the algorithm we assume that for the current orientation θ the point with y-coordinate $y_{\max,\theta}$ is on the left of the point with y-coordinate $y_{\min,\theta}$, otherwise we only update the changes in the staircases according to the current orientation without computing optimal solution. Then we will repeat the algorithm assuming that for the current orientation the point with y-coordinate $y_{\max,\theta}$ is on the right of the point with y-coordinate $y_{\min,\theta}$, and the algorithm will output the best obtained optimal solution.

Sketch of the unoriented 2-fitting algorithm

1. In $O(n \log n)$ time compute the list of the points of S which are unoriented $\pi/2$-maxima and the (at most 3) orientation intervals in $\mathbb{S}^2$ where each point is unoriented $\pi/2$-maxima. This can be done with the algorithm from Avis et al. [3]. Consider the set of orientation intervals as an arrangement of intervals ordered by their endpoints in such a way that we know which unoriented $\pi/2$-maxima points are *actives* in current sweeping orientation. Notice that the set of endpoints is linear. We compute the arrangement of intervals such that the starting point corresponds to the orientation $\alpha = \pi/4$, i.e., the algorithm starts from the staircases with horizontal and vertical segments computing the 3-polygonal chain with orientation $\theta = 0$.

2. In $O(n \log n)$ time compute the two (horizontal and vertical) staircases for S and the vertical line ℓ corresponding to the optimal solution for $\theta = 0$ which also gives the partition S_l (left) and S_r (right) and their corresponding antipodal-pairs (e_l^u, e_l^b), (e_r^u, e_r^b), which define the error tolerance of S_l and S_r. While the staircases do not change in any point, the error tolerance is defined by the same antipodal-pairs (e_l^u, e_l^b), (e_r^u, e_r^b). The error tolerance can increase or decrease according to the functions defined by the distance between two rotating calipers passing through the antipodal-pairs. Thus in constant time we can compute the optimal error tolerance while no change is produced.

3. Start the sweeping from left to right and each time that we get an endpoint: either (1) a new unoriented $\pi/2$-maxima point enter in the staircases, or (2) an active unoriented $\pi/2$-maxima point is deleted from the staircases, update all the changes in $O(\log n)$ time, including also the possible changes of the points with minimum and maximum y-coordinates for the current orientation θ. Since we consider the points in general position, at most two aligned points are updated at the same time producing a constant number of changes. When a change in the staircases is produced we have to compute the new line separating ℓ corresponding to the new optimal solution which can be computed in $O(\log n)$ time. Store and update the information of the obtained optimal solution.

The $\Omega(n \log n)$ time lower bound for the unoriented 1-fitting problem implies the $\Omega(n \log n)$ time lower for the unoriented 2-fitting problem. Nevertheless we show a concrete reduction for this problem. We reduce the problem to the MAX-GAP problem for points in the first quadrant of the unit circle centered at the origin of the coordinates system, which has an $\Omega(n \log n)$ time lower bound in the algebraic decision tree model [14]. Let $P = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ be an instance of the MAX-GAP problem. Consider a symmetric copy of points in the third quadrant and a new copy of the overall circle in other arbitrary position. It is easy to see that the optimal unoriented 2-fitting polygonal chain defines the maximum gap for P and vice versa, the maximum gap of the points in P defines the best solution for the unoriented 2-fitting problem.

Theorem 3.3. *The unorientated 2-fitting problem can be optimally solved in $\Theta(n \log n)$ time.*

4 The fitting problem in $\mathbb{R}^3$

In this section we study the fitting problem in the three-dimensional space, where a polygonal chain is interpreted as a connected configuration of orthogonal planes.

4.1 The oriented fitting problem in $\mathbb{R}^3$

The oriented 1-fitting problem. This problem corresponds to find the width of a set of points in a given direction, say the z-axis direction, or to find a plane (the splitting plane) with direction given by the z-axis which split the set S into two subsets of the same minimal width. To solve this problem we proceed as follows according to how much information we have in order to fix the orientation:

(1) Fixed the orientation of the splitting plane: For example, the orientation is given by the unit vector $(0, 0, 1)$. We solve this problem in $O(n)$ time by computing the points with maximum and minimum z-coordinates. The splitting plane is the horizontal plane passing through the mid points of the vertical distance between those maximum and the minimum points.

(2) Fixed the orientation of a line contained in the splitting plane: For example, a line in the splitting plane with orientation given by the unit vector $(0, 1, 0)$). To solve the problem we project the points onto the plane $y = 0$ and compute the convex hull of the projected points in $O(n \log n)$ time. Then we compute the width of this convex hull with a rotating caliper in (n) time. Then, in constant time, we compute both the mid line ℓ of the two parallel lines defining this width and its vector direction $\overrightarrow{v_\ell}$. The desired splitting plane is the plane which contain the line ℓ and has normal vector the cross product $\overrightarrow{v_\ell} \times (0, 1, 0)$. The total running time is $O(n \log n)$.

Proposition 4.1. *The oriented 1-fitting problem in 3D fixing the orientation of the splitting plane can be solved in $O(n)$ time and $O(n)$ space. If we only fix the orientation of a line contained in the splitting plane then the problem can be solved in $O(n \log n)$ time and $O(n)$ space.*

The oriented 2-fitting problem. This problem is defined by a monotone orthogonal 3-polygonal chain composed by three orthogonal consecutive planes. We call the *separating plane* to the plane giving the bipartition of the point set S and call the *supporting planes* to the parallel planes giving the tolerance error in each part of the bipartition of S. Fixing the orientation for the 2-fitting problem three cases can be considered: (1) fixing the orientation of both the separating plane and the parallel supported planes; (2) fixing the orientation of the separating plane; and (3) fixing the orientation of the parallel supporting planes.

(1) Fixed the orientation of both the separating plane and the parallel supported planes. Assume that the orientation of the separating plane is given by the vector $(0, 1, 0)$ and the orientation of the parallel supporting planes is given by the vector $(0, 0, 1)$. We project the points in S onto a plane with vector orientation $(1, 0, 0)$ and solve the problem as the 2D oriented 2-fitting problem in optimal $\Theta(n \log n)$ time using Theorem 2.2.

Proposition 4.2. *The oriented 2-fitting problem in 3D fixing the orientation of both the separating plane and the parallel supported planes can be solved in $\Theta(n \log n)$ time and $O(n)$ space.*

(2) Fixed the orientation of the separating plane. Assume that the orientation of the separating plane is given by the vector $(0, 1, 0)$. Sort the points in S by y-coordinate in $O(n \log n)$ time. Let $S_i = \{p_1, \ldots, p_i\}$ and $S_{n-i} = \{p_{i+1}, \ldots, p_n\}$ be the bipartition of S as the separating plane pass through p_i. In order to compute the two pairs of parallel supporting planes of S_i and S_{n-i} which gives the optimal solution, we project the points of S_i and the points of S_{n-i} into two planes parallel to the separating plane with orientation $(0, 1, 0)$. Thus, we work with the convex hulls of the projected points. Let $CH(S'_i)$ and $CH(S'_{n-i})$ be the respective convex hulls of the projected points. In order to determine the widths of the corresponding parallel supporting planes, we use two simultaneously (clockwise) rotating calipers over $CH(S'_i)$ and $CH(S'_{n-i})$. Each step is defined by the minimum rotating angle of the two calipers on antipodal pairs. Suppose that at some step, the rotating caliper over $CH(S'_i)$ has antipodal points q_1 and q_2 and the rotating caliper over $CH(S'_{n-i})$ has antipodal points q_3 and q_4 and let α be the angle which define the rotation; let w_i (w_{n-i}) be the width function of $CH(S'_i)$ ($CH(S'_{n-i})$) and $d_i = d(q_1, q_2)$ and $d_{n-i} = d(q_3, q_4)$, in the interval of rotation of the two calipers in this step. The width function w_i (w_{n-i}) depends on d_i (d_{n-i}) and $\cos(\alpha)$, thus the minimum of the maximum of the two values w_i and w_{n-i} is located either in the extremes of the rotation interval or in the intersection point of w_i and w_{n-i}; both cases can be computed in constant time. Thus, we only have to compute the widths when this fact occurs (a linear number of times) and maintain the best solution, which is the one such that the maximum of the two widths is minimum. We can update $CH(S'_i)$ and $CH(S'_{n-i})$ in $O(\log n)$ time when a point P_i changes from S_{n-i} to S_i [4], but it is not clear how to compute the new widths in $O(\log n)$ time. Thus, in each stop we spend linear time to obtain the optimal solution. The total running time is $O(n \log n) + O(n)O(n) = O(n^2)$.

(3) Fixed the orientation of the parallel supporting planes. We assume that the separating plane passes through a point p_i of S given a bipartition S_i, S_{n-i} of S, and the supporting planes of S_i and S_{n-i} are horizontal, i.e., have as normal vector $(0, 0, 1)$. Thus, the separating plane can have any normal vector in the unit circle of directions $\mathbb{S}^2$, i.e., the circle $x^2 + y^2 = 1$, so we can say that it is a vertical plane. Fixed the point p_i contained in the separating plane, we start with an orientation of the separating plane (for example the y-axis orientation) and compute the points located in both sides of this plane in $O(n)$ time, computing also the points $\max z$ and $\min z$ on each side of the plane. Then, we can preform a radial sweep with the rotating vertical plane anchored at p_i and each time the rotating plane bumps a new point we update in constant time $\max z$ and $\min z$ and store the optimal one. In order to do this step we need to know the radial order of the points $S \setminus \{p_i\}$ with respect to p_i as they are projected on a horizontal plane $z = 0$. We can compute this order in $O(n)$ time per point by reading the order in the dual formed by the arrangement of the lines corresponding to the projected points. Then we have an overall $O(n^2)$-time algorithm as we spend linear time per point p_i.

Theorem 4.3. *The oriented 3-fitting problem in 3D fixing the orientation of the separating plane or fixing the orientation of the four parallel supporting planes can be solved in $O(n^2)$ time and $O(n)$ space.*

4.2 The unoriented fitting problem in $\mathbb{R}^3$

The unoriented 1-fitting problem. The unoriented 1-fitting problem corresponds to find the orientation of the supporting planes defining the width of S, and therefore the plane just in the middle of these two planes. The width of S, $w(S)$, is defined as the minimum distance between parallel planes of support of S. Houle and Toussaint [12] show that $w(S)$ can be computed in $O(n \log n + I)$ time and $O(n)$ space, where I is the number of antipodal pairs of edges of the convex hull of S, and n is the number of vertices; in the worst case, $I = O(n^2)$. For a convex polyhedra the time complexity becomes $O(n + I)$. Agarwal and Sharir [2] give a randomized algorithm for computing the width of a point set in 3D which expected running time is $O(n^{3/2+\epsilon})$, for any $\epsilon > 0$. Therefore, the unoriented 1-fitting problem can be solved in $O(n \log n + I)$ which in the worse case is $O(n^2)$ time.

These problems were posed and partially solved during the *Third Spanish Workshop on Geometric Optimization*, July 4-8, 2006, El Rocío, Huelva, Spain. The authors would like to thank the other workshop participants for helpful comments.

References

[1] P. K. Agarwal, O. Schwarzkopf, and M. Sharir. The overlay of lower envelopes and its applications. *Discrete Computational Geometry*, 15, 1996, pp. 1–13.

[2] P. K. Agarwal and M. Sharir. Efficient randomized algorithms for some geometric optimization problems. *Discrete and Computational Geometry*, Vol. 16, No. 4, 2004, 317–337

[3] D. Avis, B. Beresford-Smith, L. Devroye, H. Elgindy, E. Guvremont, F. Hurtado, and B. Zhu. Unoriented Θ-maxima in the plane: complexity and algorithms. *SIAM Journal of Computing*, Vol. 28, No. 1, 1999, pp. 278–296.

[4] D. Avis, H. Elgindy, and R. Seidel. Simple on-line algorithms for convex polygons. *Computational Geometry*, G. T. Toussaint, ed., North-Holland, Amsterdam, 1985, pp. 23–42.

[5] P. J. Burt. Fast filter transforms for image processing. *Computer Graphics and Image Processing*, 16, (1979), 20–51.

[6] W. S. Chan and F. Chin. Approximation of polygonal curves with minimum number of line segments or minimun error. *International Journal of Computational Geometry & Applications*, 6, (1996), 59–77.

[7] F. Chin, A. Choi, and Y. Luo. Optimal generating kernel for image pyramids by piecewise fitting. *IEEE Trans. Pattern Anal. Machine Intell.*, 14, (1992), 1190–1198.

[8] J. M. Díaz-Báñez and J. A. Mesa. Fitting rectilinear polygonal curves to a set of points in the plane. *European Journal of Oper. Research*, 130, 1, 214–222, 2001.

[9] M. T. Goodrich. Efficient piecewise-linear function approximation using the uniform metric, *Discrete and Computational Geometry*, 14, (1995), 445–462.

[10] L. J. Guibas, J. E. Hershberger, J. S. B. Mitchell, and J. S. Snoeyink. Approximating polygons and subdivisions with minimum-link paths. *International Journal of Computational Geometry & Applications*, 3, (1993), 383–415.

[11] S. L. Hakimi and E. F. Schmeichel. Fitting polygonal functions to a set of points in the plane. *Graphical Models and Image Processing*, 53, (1991), 132–136.

[12] M. E. Houle and G. T. Toussaint. Computing the width of a set. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, no. 5, pp. 761–765, 1988.

[13] H. Imai and M. Iri. Polygonal approximations of a curve-formulations and algorithms. *Computational Morphology.* G. T. Toussaint ed., North Holland, 1988.

[14] D. T. Lee and Y. F. Wu. Geometric complexity of some location problems. *Algorithmica*, 1 (1986) 193–211.

[15] M. A. López and Y. Mayster. Approximating a set of points by a step function. *Journal of Visual Communication and Image Representation*, (accepted), 2006.

[16] T. Pavlidis. Algorithms for shape analysis of contours and waveforms. *IEEE Transation Pattern Analysis Machine Intelligence*, Pami-2, (1980), 301–312.

[17] F. P. Preparata and M. I. Shamos. *Computacional Geometry, an Introduction.* Springer-Verlag, 1988.

[18] G. T. Toussaint. On the complexity of approximating polygonal curves in the plane. *Proc. IASTED, International Symposium on Robotics and Automation*, Lugano, Switzerland, 1985.

[19] D. P. Wang. A new algorithm for fitting a rectilinear x-monotone curve to a set of points in the plane. *Pattern Recognition Letters*, vol. 23, no. 1, 329–334(6), 2002.

[20] D. P. Wang, N. F. Huang, H. S. Chao, and R. C. T. Lee. Plane sweep algorithms for polygonal approximation problems with applications. *Lecture Notes in Computer Science*, 762, (1993), 515–522.