Finding Specified Sections of Arrangements: 2D Results

P. Bose[1] F. Hurtado[2] H. Meijer[3] S. Ramaswami[4] D. Rappaport[3]
V. Sacristán[2] T. Shermer[5] G. Toussaint[6]

October 6, 2000

Abstract

Given a configuration $\mathcal{C}$ of geometric objects in R^2 (called the *input configuration*), a *target configuration* $\mathcal{T}$ of geometric objects in R^1, and a class $\mathcal{S}$ of allowable sectioning lines we consider in this paper many variations on the following problem: "Is there a line $S \in \mathcal{S}$ such that the section $S \cap \mathcal{C}$ is equivalent by rigid motion to the target $\mathcal{T}$?"

KEYWORDS: Section, projection, tomography, probe, stereology, morphology, recognition.

1 Introduction

Mathematical Tomography deals with a set of "techniques of reconstructing internal structures in a body from data collected by detectors (sensitive to some sort of energy) outside the body" [23]. For example, one technique in computerized tomography consists on measuring the attenuation of X-rays between multiple pairs of points outside the body, each pair giving positions for a source and a detector, from that the *Radon transform* is estimated, and the density distribution approximated. Mathematical methods in tomography constitute a rich area of research whose basic results go back to the beginning of the century (Radon proved his inversion formula in 1917), but that has become especially active in the last two decades, as new technologies and the spreading of computers have made possible many ways of gathering data about bodies as well a powerful capability of handling the information [10, 22, 23, 24, 25, 30].

When density functions are replaced by geometric objects (for example we have a convex polytope instead of a body with non-constant internal density) then geometric information – shape, measure, ... – is the detected data, and we arrive to the area of *Geometric Tomography* [14, 15], a topic recently emerged as a well defined domain of research, which is described by Gardner in [15] as "the area of mathematics dealing with the retrieval of information about a geometric object from data about its sections, or projections or both". As an example of how geometry changes the situation, let us recall the fact that a planar density distribution is determined by its X-rays taken in *every* direction, while Gardner and McMullen [19] proved that a selected set of *four* directions determine an homogeneous convex body. Certainly Geometric Tomography and Computerized Tomography overlap in several problems, and there are also many related domains, such as Stereology, Mathematical Morphology, Image Analysis, Pattern Recognition and Geometric Probing.

Computational Geometric Tomography is a natural name for an area grouping results in which the emphasis is on the algorithmic aspect of the problems above. It is illuminating to distinguish first between *direct* and *inverse* problems. In direct problems the input is a geometric object and the scope is to efficiently compute a section or a projection with prescribed properties. In inverse problems, which are at the core of Mathematical Tomography, sections and/or projections are the input and the aim is to determine, verify, reconstruct or approximate the object, or some of its properties. Direct problems, besides its intrinsical

[1]Carleton University, Ottawa.
[2]Universitat Politècnica de Catalunya, Barcelona. Partially supported by Proyecto DGES-SEUID PB96-0005-C02-02.
[3]Queen's University, Kingston.
[4]Rutgers University, Camden.
[5]Simon Fraser University, Burnaby.
[6]McGill University, Montréal.

interest, provide often the basis for solving the inverse ones. Both kind of problems have attracted a lot of attention in Computational Geometry, a few examples follow.

In [28] McKenna and Seidel gave algorithms for minimizing or maximizing the shadow of a polyhedron by projection, a topic also studied by Burger, Gritzmann and Klee in [8]. In [2] Avis et al. compute the maximum-area horizontal cross-section of a convex polytope. Bose et al. in [6] and Gómez, Hurtado and Toussaint in [21] consider the problem of computing "nice" orthographic projections of objects in 3-space, according to different criteria of "niceness". The computation of shadows has been also studied by Chazelle, Edelsbrunner and Guibas in [9], Ponce et al. in [31, 32] and Amenta and Ziegler in [1]; these papers contain also combinatorial results and pointers to many related works.

Boissonnat in [5], Barequet and Sharir in [3] and Gitlin, O'Rourke and Subramannian in [20] studied the problem of reconstructing polyhedra from parallel slices by interpolation, a problem in which fairly extensive work has been done. The compatibility of projections of point sets (whether or not they can come from the same object) is considered in [21]. *Geometric Probing*, as described by Skiena in [33], "considers problems of determining a geometric structure or some aspect of that structure from the results of a mathematical or physical measuring device, a *probe*". This area of research, which is certainly related to Geometric Tomography, has attracted a lot of attention [11, 13, 26, 27, 29].

In spite of that amount of research several basic facts remain unexplored, a fact which is not strange given the huge variety of problems and tools involved.

In this paper we study some fundamental 2D recognition problems involving objects and arrangements that are typical in Computational Geometry (3D analogous results are described in a companion paper [7]). Specifically, given a configuration $\mathcal{C}$ of geometric objects in R^2 (called the *input configuration*), a *target configuration* $\mathcal{T}$ of geometric objects in R^1, and a class $\mathcal{S}$ of allowable sectioning lines we consider many variations on the following problem: "Is there a line $S \in \mathcal{S}$ such that the section $S \cap \mathcal{C}$ is equivalent by rigid motion (translation plus rotation) to the target $\mathcal{T}$?" In the affirmative, the algorithms will report all the solutions, if there are a finite number. When there is an infinite set of parallel sectioning lines that give the same combinatorial solution, only one of them will be reported. This framework is denoted throughout the paper as the *sectioning problem*.

The classes of input configurations that we will consider are sets of line segments (where the target is a set of points)), sets of lines (where the target is a set of points), sets of discs (where the target is a set of intervals), and sets of polygons (where the target is a set of intervals). We consider all lines, parallel lines and lines through a point, as allowable sectioning classes. We use n to denote the number of objects in $\mathcal{C}$, and k for the number of objects in $\mathcal{T}$.

The following table summarizes our results.

Input dimension	Input configuration (target)	allowable sectioning hyperplanes: horizontal	origin	any
2D	line segments (points)	$O(n^2)$	$O(n^2)$	$O(n^3)$
	lines (points)	$\Theta(n \log n)$	$O(n^2)$	$O(n^3)$
	polygons (intervals)	$O(n^2)$	$O(n^2)$	$O(n^3)$
	discs (intervals)	$O(n^2)$	$O(n^2)$	$O(n^3)$

2 Preliminary results

In this section we will prove two lemmas that are used in subsequent sections of this paper. Assume we have a line l containing three points a, b and c, and three lines l_a, l_b and l_c. These three lines are fixed, but the line l can be rotated and translated. We are interested in determining the number of ways the line l can be placed with its points a, b and c on the three lines l_a, l_b and l_c respectively. For lines in general position the number of placements is at most 2, but in some degenerate cases the number can be infinite. For example when the three lines are parallel, the number of placements is either zero or infinite. If two of the lines are parallel, but the third has a different direction, then the number of placements is at most 2. All other cases follow immediately from our first lemma below. The second lemma of this section applies to a similar situation.

Lemma 2.1 *Let l_a and l_b be two non-parallel lines. Let l be a line containing three points a, b and c. If we consider all the placements of l such that a and b lie on l_a and l_b respectively, then the collection of corresponding placements of the point c forms an ellipse.*

PROOF

Let the point c be determined by the condition $c = a + s \cdot \overrightarrow{ab}$, where s is a fixed real number different from 0 and from 1. Let k be the distance between a and b. Without loss of generality assume that l_a is the x-axis and that l_b is the line $x = my$, for some constant m. Pick any point $(\alpha, 0)$ in l_a and any point $(m\beta, \beta)$ in l_b and consider these points as candidate positions for a and b; the corresponding position for c would be the point $(x_c, y_c) = (\alpha, 0) + s(m\beta - \alpha, \beta) = ((1-s)\alpha + ms\beta, s\beta)$.

From this we get

$$\alpha = \frac{x_c - my_c}{1-s} \quad , \quad \beta = \frac{y_c}{s} .$$

But the candidates for a and b give a true placement if and only if they are at distance k apart, i.e. if and only if $(\alpha - m\beta)^2 + \beta^2 = k^2$; by substitution we get the locus

$$\left(\frac{sx_c - my_c}{s^2 - s}\right)^2 + \left(\frac{y_c}{s}\right)^2 = k^2$$

for the placements of c, which is an ellipse as claimed. □

Lemma 2.2 *Let l_a and l_b be two lines. Let c_a and c_b two circles bounding disjoint discs, and let p be a point, $p \notin l_a \cap l_b$. Consider a line l containing two points a and b separated by a distance k. The line l can be rotated and translated, but all the other objects are fixed. Then: i) there are at most four placements of l such that l contains p, l_a contains a and l_b contains b; ii) there are at most a constant number δ of placements of l such that l contains p, c_a contains a, c_b contains b, and all the points of $l \cap c_a$ and $l \cap c_b$ lie between a and b.*

PROOF

(i) If l_a and l_b are parallel, the result obviously holds, so without loss of generality assume that p is the point $(0, 0)$, l_a is the line $y = 1$ and l_b is the line $x = cy + d$. A candidate placement for l as the line $x = my$ through $(0, 0)$ produces the point $(m, 1)$ in l_a and the point $(\frac{md}{m-c}, \frac{d}{m-c})$ in l_b, which corresponds to a placement for a and b if and only if the distance between them is k, hence we get the equation

$$\left(m - \frac{md}{m-c}\right)^2 + \left(1 - \frac{d}{m-c}\right)^2 = k^2 ,$$

which can be expressed as

$$(m^2 + 1)(m - c - d)^2 - k^2(m - c)^2 = 0 ,$$

a degree four polynomial from which we obtain at most four values for m.

(ii) We prove here an upper bound $\delta \leq 32$ by relaxing the constraints. Without loss of generality assume that p is the point $(0, 0)$, c_a is the circle $(x - a_1)^2 + (y - b_1)^2 = r_1^2$, c_b is the circle $(x - a_2)^2 + (y - b_2)^2 = r_2^2$, and that at least one of the circle lies entirely in the halfplane $x > 0$. Let us consider a candidate placement for l as the line $y = mx$, and assume that the points a and b are (t, mt) and (z, mz) respectively. As they must be at distance k we have $(t - z)^2 + (mt - mz)^2 = k^2$, and this equation gives the two possibilities

$$z = \frac{-k\sqrt{1 + m^2} + t + m^2 t}{1 + m^2} \; ; \; z = \frac{k\sqrt{1 + m^2} + t + m^2 t}{1 + m^2} .$$

We take the first solution (the argument is identical for the second one), use the value of z for point b, and impose now that c_a and c_b must contain a and b, respectively, which gives

$$(t - a_1)^2 + (mt - b_1)^2 - r_1^2 = 0 ;$$

$$\left(\frac{-k\sqrt{1+m^2}+t+m^2t}{1+m^2}-a_2\right)^2+\left(\frac{m\big(-k\sqrt{1+m^2}+t+m^2t\big)}{1+m^2}-b_2\right)^2-r_2^2=0\,.$$

These two equations can be considered as polynomials in t, which will have a common root, therefore their resultant ([18], page 135) must be 0. This resultant is an algebraic equation in m, which can be transformed into a polynomial equation in m with degree 16 and leading coefficient r_2^8; therefore there are at most 16 possible solutions for m. □

3 Two dimensional problems

The classes of input configurations that we will consider are sets of line segments (where the target is a set of points)), sets of lines (where the target is a set of points), sets of polygons (where the target is a set of intervals), and sets of discs (where the target is a set of intervals). Each of these classes of input configurations is treated in a separate subsection below.

3.1 Input configuration: set of non-crossing line segments

In the first three theorems we look at the case of non-crossing line segments i.e. line segments that may intersect each other only in their endpoints. Therefore the results also apply to sets of non-intersecting polygons.

Theorem 3.1 *The following sectioning problem can be solved in $O(n^2)$ time:*

> **Input configuration:** *A set of n non-crossing line segments in R^2.*
>
> **Target configuration:** *A set of k points in R^1.*
>
> **Allowable sections:** *Horizontal lines.*

PROOF

1. Sort the points in $\mathcal{T}$. Let $p_0, p_1, \cdots, p_{k-1}$ denote the sorted points. Compute the distances between consecutive points.
2. Let b_i denote the bottom y-coordinate of line segment i, and t_i denote its top y-coordinate. Compute b_i and t_i for all line segments. Sort the list of all values b_i and t_i and define a horizontal slice as the set of all horizontal lines between consecutive values in this sorted list.
3. Sort the list of line segments in each horizontal slice. By processing the horizontal slices from top to bottom this step can be done in $O(n^2)$ time.
4. In each horizontal slice, find two neighbouring line segments that are non-parallel, line segments l_i and l_{i+1} say. Let δ_0 be the distance between the corresponding points p_i and p_{i+1} in $\mathcal{T}$ and let δ_1 be the distance between the corresponding points p_{k-1-i} and p_{k-2-i}. Determine the two positions of the sectioning line with the correct distance δ_0 or δ_1 between l_i and l_{i+1}. Verify in linear time whether or not the candidate sectioning lines are equivalent to $\mathcal{T}$. If all lines in the horizontal slice are parallel, then either all horizontal lines in this slice are correct sectioning lines, or none are, which can be verified in linear time.
5. For each horizontal line on the boundary of a horizontal slice, verify in linear time whether or not it is equivalent to $\mathcal{T}$.

Since there are a linear number of horizontal slices, this algorithm requires $O(n^2)$ time. □

Theorem 3.2 *The following sectioning problem can be solved in $O(n^2)$ time:*

Input configuration: *A set of n non-crossing line segments in R^2.*

Target configuration: *A set of k points in R^1.*

Allowable sections: *Lines through the origin.*

PROOF

1. Sort the points in $\mathcal{T}$. Let $p_0, p_1, \cdots, p_{k-1}$ denote the sorted points. Let Δ be the distance between p_0 and p_{k-1}.
2. For each line segment, compute the lines l_i and t_i that pass through the origin and the lower and top endpoints of the segment respectively.
3. Radially sort the t_i's and l_i's and define an interval as the set of lines through the origin between consecutive lines in this sorted list.
4. Sort the list of line segments in each interval. By processing the intervals consecutively this step can be done in $O(n^2)$ time.
5. In each interval compute the locations with a correct value for Δ. From Lemma 2.2 we know that there are at most a constant number of such locations in each interval. Test each candidate in linear time.
6. For each line l_i and t_i, verify in linear time whether or not it is equivalent to $\mathcal{T}$.

Since there are a linear number of lines l_i and t_i, this algorithm requires $O(n^2)$ time. □

Theorem 3.3 *The following sectioning problem can be solved in $O(n^3)$ time:*

Input configuration: *A set of n non-crossing line segments in R^2.*

Target configuration: *A set of k points in R^1.*

Allowable sections: *Any line.*

PROOF

Sort the points in $\mathcal{T}$. Let $p_0, p_1, \cdots, p_{k-1}$ denote the sorted points. We first determine whether there is a sectioning line equivalent to $\mathcal{T}$ that does not pass through an endpoint of a line segment. We will solve this problem by mapping points (a, b) to a line in dual space with equation $y = ax - b$. Each line segment in primal space is represented in the dual space by a collection of lines through a single point, resulting in an area usually called a bow tie [4]. The combinatorial complexity of the resulting arrangement of all bow ties in dual space is $O(n^2)$. Assume the target contains k points. We traverse the arrangement in dual space and maintain a sorted list of corresponding line segments in primal space. For each area that is the intersection of k bow ties, there is a corresponding collection of k line segments in the primal space that can be stabbed by a line. For each such collection of k line segments, we either discover that all line segments are parallel or find three line segments l_0, l_1 and l_i that are not all three parallel to each other. From Lemma 2.1 we derive that in the latter case there are at most two lines that can stab the k line segments such that the distances between the intersection points of the three lines l_0, l_1 and l_i and the sectioning lines are equal to the distances between p_0, p_1 and p_i. Also there are at most two sectioning lines that give distances equal to the distances between p_{k-1}, p_{k-2} and p_{k-1-i}. If all lines segments are parallel, we verify whether or not there is a sectioning line that matches $\mathcal{T}$. Since each verification step can be done in linear time, the overall complexity of this part of the algorithm is $O(n^3)$.

Sectioning lines equivalent to $\mathcal{T}$ that do pass through an endpoint of a line segment can be found by applying the $O(n^2)$ algorithm of the previous lemma, using the endpoints of the line segments as the origins of the sectioning lines. Since there are a linear number of endpoints, the overall complexity of this step remains $O(n^3)$. □

3.2 Input configuration: set of lines

The following theorem shows that there is an $O(n \log n)$ algorithm if the lines are non-parallel.

Theorem 3.4 *The following sectioning problem can be solved in $O(n \log n)$ time:*

> **Input configuration:** *A set of n non-parallel lines in R^2.*
>
> **Target configuration:** *A set of k points in R^1.*
>
> **Allowable sections:** *Horizontal lines.*

PROOF

Sort the target points. Let $p_0, p_1, \cdots, p_{k-1}$ denote the sorted points. Compute the diameter Δ, the distance between p_0 and p_{k-1}. Compute the left and right envelopes of the line arrangement (by halfplane intersection). Both steps use $O(n \log n)$ time.

Find the candidate positions for the sectioning line with the correct value for Δ. It is not hard to see that there are at most two such positions since the diameter of the sectioning lines is a unimodal function. Hence the two positions can be found by binary search. We check each one in $O(n \log n)$ time to see if we get our target set of points. □

If we allow parallel input lines, we can still solve the above problem in $O(n \log n)$ time if $k = n$. We may find an infinite number of candidate positions for the sectioning line with the correct value for Δ. If that occurs, there are two lines, l_0 and l_{n-1} say, that intersect all candidate sectioning lines in points corresponding to p_0 and p_{n-1} (or p_{n-1} and p_0). These two lines l_0 and l_{n-1} are parallel. If all input lines are parallel, we may have zero or an infinite number of correct positions for the sectioning line, which we can verify in $O(n \log n)$ time. Otherwise we remove from the set of input lines all lines that are parallel to l_0 (but not l_0 itself). We can verify that the corresponding points on $\mathcal{T}$ exist in the list $p_0, p_1, \cdots, p_{n-1}$ (or in the list $p_{n-1}, p_{n-2}, \cdots, p_0$). If they do not all exist, there is no correct position for the sectioning line. If they do exist, we remove these points from $\mathcal{T}$ and repeat this algorithm, with the added restriction that the intersection of l_0 with the sectioning line should correspond to p_0 (or p_{n-1}) on $\mathcal{T}$. In this second iteration, there can at most be two positions where the points on the sectioning line have the correct diameter. So the overall complexity of the algorithm is $O(n \log n)$.

If $k < n$, we do not know whether an $O(n \log n)$ algorithm exists. We can prove a lower bound, which matches the complexity of the algorithm when $k = n$ or the lines are non-parallel:

Theorem 3.5 *The following sectioning problem requires $\Omega(n \log n)$ time:*

> **Input configuration:** *A set of n lines in R^2.*
>
> **Target configuration:** *A set of k points in R^1.*
>
> **Allowable sections:** *Horizontal lines.*

PROOF

We prove the theorem by reduction from **Uniform Gap**: given a set X of n real numbers, $X = \{x_0, \cdots, x_{n-1}\}$, and a positive value $\epsilon \in R$, it takes $\Omega(n \log n)$ time to decide whether or not the gap between each pair of consecutive numbers in X is ϵ. The main idea of the reduction is to construct a set of input lines and a set of target points from the set X, such that the uniform gap problem can be answered by solving the sectioning problem.

As input configuration consider the n lines through the origin and the points $(i, 1)$, $i = 1, \ldots, n$. As target configuration, simply consider the set $X \subset R$. Without loss of generality, we can assume that $x_0 \leq x_i \leq x_{n-1}$ for all $i = 1, \cdots, n-2$. If you find that the target X matches the arrangement, check whether or not $x_{n-1} - x_0 = (n-1)\epsilon$. If so, the answer to the uniform gap problem is yes, otherwise, the answer is no. □

Theorem 3.6 *The following sectioning problem can be solved in $O(n^2)$ time:*

Input configuration: *A set of n lines in* R^2.

Target configuration: *A set of k points in* R^1.

Allowable sections: *Lines through the origin.*

PROOF

First, construct the arrangement formed by the input lines, in $O(n^2)$ time. Sort the target points p_0, $p_1, \cdots, p_{k-1}$ and compute the diameter Δ, the distance between p_0 and p_{k-1}.

Consider the diameter of any section whose section line contains the origin. The points realizing this diameter will be from (radially opposite) edges on the outer envelope of the input arrangement. By Lemma 2.2 any pair of radially opposite lines on the outer envelope can give rise to a constant number of candidate sections.

There are at most $O(n)$ such opposite pairs, and they can be found in $O(n)$ time from the arrangement by a traversal of the outer envelope. From the opposite pairs, we construct the $O(n)$ candidate sections, and then test each section to see if it is similar to the target. This test can be performed by walking the zone of the section line in the input arrangement (in $O(n)$ time), seeing if the distance between consecutive points is the same as in the sorted target set.

Since we do an $O(n)$ test for $O(n)$ candidate sections, and have $O(n^2)$ preprocessing, we have spent $O(n^2)$ time. □

Theorem 3.7 *The following sectioning problem can be solved in* $O(n^3)$ *time:*

Input configuration: *A set of n non-parallel lines in* R^2.

Target configuration: *A set of k points in* R^1.

Allowable sections: *Any line.*

PROOF

First, construct the arrangement formed by the input lines, in $O(n^2)$ time. Sort the points in $\mathcal{T}$ and let $p_0, p_1, \cdots, p_{k-1}$ denote the sorted points. Let δ_i be the distance between p_i and p_{i+1}. We first assume that $n = k$.

For each unbounded cell C in the arrangement we repeat the following algorithm. Let the edges in the arrangement forming the unbounded cell C be the 0-level edges. Define an i-level edge as follows: draw a line segment from a point on the interior of the edge to C in such a way that it does not intersect a vertex in the arrangement; the number of edges intersected by this line segment is i. From [12] we know that the combinatorial complexities of the sets of 0-, 1- and 2-level edges are linear. Consider sectioning lines that originate in C and pass through a 2-level edge e. Because there are 2 lines between e and C, there are at most two different 1-level edges intersected by these sectioning lines. From Lemma 2.1 we derive that each one of these two 1-level edges give rise to at most two positions for the sectioning line such that it gives the correct values for δ_0 and δ_1. For each candidate we traverse the arrangement in linear time and verify whether or not it is equivalent to $\mathcal{T}$. This part of the algorithm requires $O(n^2)$ time for each cell C, resulting in on overall complexity of $O(n^3)$.

If $k < n$ then the sectioning line is parallel to one of the input lines or intersects at least one vertex in the arrangement. The first case can be solved by applying the $O(n \log n)$ algorithm of Theorem 3.4 for n possible directions. For the second case, let p_i be the first point on $\mathcal{T}$ that is the intersection of the sectioning line and a vertex of the arrangement. If $i > 2$ then the above algorithm will find the sectioning line. If $i \leq 2$ then the $O(n^2)$ algorithm of Theorem 3.6 can be applied to all the vertices in the 2-, 1- and 0-levels. □

As is the case with Theorem 3.4, the above theorem holds if there are parallel lines in the input set, provided that $n = k$. The only case for which the algorithm has to be modified is when we find a line intersecting three parallel 0-level, 1-level and 2-level edges that give the correct values for δ_0 and δ_1. In that case, there are two possible orientations for the sectioning line, but an infinite number of positions. First we can find all lines parallel to the first three edges and verify that the resulting intersection points with the sectioning lines exist in $\mathcal{T}$. If they do, we can remove these points from $\mathcal{T}$ and now use the diameter Δ of the remaining points in $\mathcal{T}$ to find at most 2 locations for the sectioning line for each of the two possible orientations. So this modified step can still be executed in linear time.

3.3 Input configuration: set of disjoint discs

Theorem 3.8 *The following sectioning problem can be solved in $O(n^2)$ time:*

> **Input configuration:** *A set of n disjoint discs in R^2.*
>
> **Target configuration:** *A set of k intervals in R^1.*
>
> **Allowable sections:** *Horizontal lines.*

PROOF

1. Compute Δ, the diameter of the set of intervals in $\mathcal{T}$.
2. Let b_i denote the bottom y-coordinate of disc i, and t_i denote its top y-coordinate. Compute b_i and t_i for all discs. Sort the list of all values b_i and t_i and define a horizontal slice as the set of all horizontal lines between consecutive values in this sorted list.
3. Sort the list of discs in each horizontal slice. By processing the horizontal slices from top to bottom this step can be done in $O(n^2)$ time.
4. In each horizontal slice there are at most two candidate sectioning lines with the correct diameter Δ. Verify in linear time whether or not the candidate sectioning lines are equivalent to $\mathcal{T}$.
5. For each horizontal line on the boundary of a horizontal slice, verify in linear time whether or not it is equivalent to $\mathcal{T}$.

Since there are a linear number of horizontal slices, this algorithm requires $O(n^2)$ time. □

Theorem 3.9 *The following sectioning problem can be solved in $O(n^2)$ time:*

> **Input configuration:** *A set of n disjoint discs in R^2.*
>
> **Target configuration:** *A set of k intervals in R^1.*
>
> **Allowable sections:** *Lines through the origin.*

PROOF

1. Let Δ be the diameter of the intervals in $\mathcal{T}$.
2. For each disc, compute lower and top tangent line slopes l_i and t_i.
3. Radially sort the t_i's and l_i's and define an interval as the set of lines through the origin between consecutive lines in this sorted list.
4. Sort the list of discs in each interval. By processing the intervals consecutively this step can be done in $O(n^2)$ time.
5. In each interval compute the locations with a correct value for Δ. From Lemma 2.2 we know that there are at most a constant number of such locations in each interval. Test each candidate in linear time.
6. For each line l_i and t_i, verify in linear time whether or not it is equivalent to $\mathcal{T}$.

Since there are a linear number of lines l_i and t_i, this algorithm requires $O(n^2)$ time. □

Theorem 3.10 *The following sectioning problem can be solved in $O(n^3)$ time:*

> **Input configuration:** *A set of n disjoint discs in R^2.*

Target configuration: *A set of k intervals in R^1.*

Allowable sections: *Any line.*

PROOF

Sort the intervals on $\mathcal{T}$ and let δ_0 and δ_1 be the lengths of the first two intervals. We first assume that there is no interval on $\mathcal{T}$ of length 0. We will solve this problem by mapping points (a, b) to a line in dual space with equation $y = ax - b$. It can be shown that a disc in primal space maps to the set of lines enclosed in the region between two hyperbolic curves [4]. The combinatorial complexity of the resulting arrangement of all hyperbolic areas in dual space is $O(n^2)$. Assume the target contains k intervals. We traverse the arrangement in dual space and maintain a sorted list of corresponding discs in primal space. For each area that is the intersection of k hyperbolic areas, there is a corresponding collection of k discs in the primal space that can be stabbed by a line. For each such collection of k discs we examine the eight sectioning lines whose intersection with the first two or the last two discs in the collection have the correct lengths δ_0 and δ_1. Since the verification steps can be done in linear time, the overall complexity of this part of the algorithm is $O(n^3)$.

If $\mathcal{T}$ has intervals of length 0, we also have to consider the edges of the arrangement in dual space. Since there are $O(n^2)$ edges, this does not increase the asymptotic running time of the algorithm. □

3.4 Input configuration: set of disjoint polygons

Since a polygon is a set of line segments that intersect each other only in their end points, the following three theorems follow immediately from the previous theorems dealing with non-crossing line segments.

Theorem 3.11 *The following sectioning problem can be solved in $O(n^2)$ time:*

Input configuration: *A set of n disjoint polygons in R^2.*

Target configuration: *A set of k intervals in R^1.*

Allowable sections: *Horizontal lines.*

Theorem 3.12 *The following sectioning problem can be solved in $O(n^2)$ time:*

Input configuration: *A set of n disjoint polygons in R^2.*

Target configuration: *A set of k intervals in R^1.*

Allowable sections: *Lines through the origin.*

Theorem 3.13 *The following sectioning problem can be solved in $O(n^3)$ time:*

Input configuration: *A set of n disjoint polygons in R^2.*

Target configuration: *A set of k intervals in R^1.*

Allowable sections: *Any line.*

Acknowledgment

This research started at Bellairs Research Institute of McGill University during the International Workshop on Computational Geometric Tomography.

References

[1] N. Amenta and G. Ziegler, Shadows and slices of polytopes, *Proc. of 12th ACM Symp. on Comp. Geom*, 1996, pp. 10-19.

[2] D. Avis, P. Bose, T. Shermer, J. Snoeyink, G. Toussaint and B. Zhu, On the Sectional Area of Convex Polytopes, *Proc. of 12th ACM Symp. on Comp. Geom*, 1996.

[3] G. Barequet and M. Sharir, Piecewise Linear Interpolation between Polygonal Slices, *Proc. of 10th ACM Symp. on Comp. Geom*, 1994, pp. 93-102.

[4] M. de Berg, M. van Kreveld, M. Overmars, O. Schwarzkopf, Computational Geometry: Algorithms and Applications, Springer-Verlag (1997).

[5] J. D. Boissonnat, Shape reconstruction form planar cross sections, *Computer Vision, Graphics and Image Processing*, 44 (1988), pp. 1-29.

[6] P. Bose, F. Gómez, P. Ramos and G. Toussaint, Drawing nice projections of objects in space, *Proc. of Graph Drawing'95, Lect. Notes in Comp. Sci. 1027*, pp. 52-63, Springer-Verlag, 1996.

[7] P. Bose, F. Hurtado, H. Meijer, S. Ramaswami, D. Rappaport, V. Sacristán, T. Shermer and G. Toussaint, Finding Specified Sections of Arrangements: 3D Results, manuscript in preparation.

[8] T. Burger, P. Gritzmann and V. Klee, Polytope projection and projection polytopes, TR. No 95-14, Dept. Math., Trier University.

[9] B. Chazelle, H. Edelsbrunner and L. Guibas, The complexity of cutting complexes, *Discr. and Comp. Geom.* 4, pp. 139-182, 1989.

[10] S. Deans, *The Radon Transform and Some of Its Applications*, John Wiley &Sons, New York, 1983.

[11] D. P. Dobkin, H. Edelsbrunner and C. K. Yap, Probing convex polytopes, *Proc. 18th ACM Symp. Theory of Computing*, 1986, pp. 424-432.

[12] H. Edelsbrunner, Algorithms in Combinatorial Geometry, *EATCS Monographs on Theoretical Computer Science*, Vol. 10, Springer-Verlag, 1987.

[13] H. Edelsbrunner and S. Skiena, Probing convex polygons with X-rays, *SIAM J. Comp.*, **17** (1988), pp. 870-882.

[14] R. J. Gardner, Geometric Tomography, *Notices of the AMS*, Vol. 42, 4, April 1995, pp. 422-429.

[15] R. J. Gardner, *Geometric Tomography*, Encyclopedia of Mathematics and its Applications, vol. 58, Cambridge University Press, New York, 1995.

[16] R. J. Gardner and P.Gritzmann, Successive determination and verification of polytopes by their X-rays, *J. London Math. Society*, (2) **50** (1994), pp. 375-391.

[17] R. J. Gardner and P.Gritzmann, Determination of finite sets by X-rays, *Abstracts of the 12th European Workshop on Comp. Geom.*, Münster, 1996, pp. 71-72.

[18] Serge L. Lang, *Algebra*, Addison-Wesley, 1967.

[19] R. J. Gardner and P. McMullen, On Hammer's X-ray problem, *J. London Math. Soc.* (2) **21** (1980), pp. 171-175.

[20] C. Gitlin, J. O'Rourke and V. Subramannian, On reconstructing polyhedra from parallel slices, *Internat. J. Comput. Geom. Appl.* (**6**), pp. 103-122, 1996.

[21] F. Gómez, F. Hurtado and G. Toussaint, Proyecciones de calidad y reconstrucción de conjuntos (in spanish), *Proc. of Congreso Espanol de Informática Gráfica*, pp. 18-27, 1996.

[22] G. Herman *Image Reconstruction from Projections: The Fundamentals of Computerized Tomography* Academic Press, New York, 1980.

[23] G. T. Herman, A. K. Louis and F. Natterer (Eds.), *Mathematical Methods in Tomography*, Lecture Notes in Mathematics, vol. 1497, Springer-Verlag, Heidelberg, 1990.

[24] G. T. Herman and F. Natterer (Eds.), *Mathematical Aspects of Computerized Tomography*, Springer-Verlag, Berlin, 1981.

[25] G. Herman, H. Tuy, K. Langenberg and P. Sabatier, *Basic Methods of Tomography and Inverse Problems*, Adam Hilger, Bristol, England, 1987.

[26] S. Y. R. Li, Reconstruction of polygons from projections, *Inf. Proc. Letters*, vol. 28, pp. 235-240, 1988.

[27] M. Lindenbaum and A. Bruckstein, Reconstruction of polygonal sets by constrained and unconstrained double probing, *Annals of Math. and Artif. Int.*, to be published.

[28] M. McKenna and R. Seidel, Finding the optimal shadows of a convex polytope, *Proc. ACM Symp. on Comp. Geom.*, 1985, pp. 24-28.

[29] H. Meijer and S. Skiena, Reconstructing Polygons from X-Rays, *Proc. 5th Canadian Conf. on Comp. Geom*, pp. 381-386, 1993.

[30] F. Natterer, *The Mathematics of Computerized Tomography*, John Wiley &Sons, Chichester, England, 1986.

[31] J. Ponce and B. Faverjon, On computing three-finger force-closure grasps of polygonal objects, *IEEE Trans. on Robotics and Automation*, 11:6, 1995.

[32] J. Ponce, S. Sullivan, A. Sudsang, J.D. Boissonnat and J.P. Merlet, On computing four-finger equilibrium and force-closure grasps of polyhedral objects, to appear in *Int. Journal of Rob. Research*, 1996.

[33] S. Skiena, Interactive reconstruction via geometric probing, *Proc. IEEE* **80**, 1992, pp. 1364-1383.