Optimal Nearly-Similar Polygon Stabbers of Convex Polygons
Extended Abstract

Paul Colley colley@qucis.queensu.ca
Henk Meijer henk@qucis.queensu.ca
David Rappaport daver@qucis.queensu.ca
Department of Computer and Information Science
Queen's University
Kingston, Ontario, Canada K7L 3N6

April 14, 1994

Abstract

A convex polygon that is *nearly-similar* to a model polygon P has sides parallel and in the same order to the corresponding sides of P. The lengths of the sides are unrestricted and may be zero.

Given a set of target convex polygons in the plane with a total of n vertices, and a fixed model convex stabbing polygon P, the minimum-perimeter polygon nearly-similar to P that stabs the targets can be found in time $O(n)$ (but time exponential in the number of sides in the stabbing polygon).

If P is an isothetic rectangle, then the minimum-area polygon stabbing the targets can be found in $O(n \log n)$ time.

1 Introduction

The problem of intersecting a finite collection of closed subsets of the plane with a common line is the subject of many results in discrete and computational geometry. Such a line is known as a *line transversal*, or *stabbing line*. A good account of combinatorial aspects of the problem is found in [11]. Algorithms for computing line stabbers include [13] for computing a line stabber for a set of parallel lines, and [9] for computing a line stabber of a set of arbitrarily oriented line segments. In [1, 7, 10] algorithms are given for stabbing collections of simple objects with a line. Algorithms for stabbing lines, line segments and polyhedra with a line in three dimensions are given in [3]. In [2], a general approach based on linear programming is given for stabbing d-dimensional polyhedra with a $d-1$ hyperplane.

In this paper we examine optimization problems as a generalization of stabbing line problems. That is, given a class of stabbers, find one which is optimal under some measure.

In [4] an $O(n \log n)$ algorithm is presented to determine a shortest line segment that stabs a set of n convex polygons. When using a two dimensional object for stabbing one can choose to optimize either area or perimeter. In the first case [5] presents an $O(n \log n)$ algorithm to determine a minimum area convex polygon that stabs a set of n parallel line segments. Optimizing the perimeter of a convex polygon stabber is discussed in [12, 14] where $O(n \log n)$ algorithms are given to determine a minimum perimeter convex polygon stabber of line segments restricted to lie in one or k fixed directions respectively. Stabbing line segments with a smallest convex polygon, optimizing either area or perimeter, is not known to be polynomial nor is it known to be NP-complete.

In this paper we are primarily concerned with smallest convex polygonal stabbers. We admit an input set, T, of closed convex polygonal subsets of the plane. In section 3 we give a linear time algorithm to determine a smallest perimeter nearly-similar stabber of T, by formulating the problem as a linear program. In section 4 we use a divide and conquer approach to determine a smallest area isothetic rectangle stabber of S in $O(n \log n)$ time. We have omitted some of the proofs in this extended abstract. We refer the reader to [6] for the full version of this paper.

2 Preliminaries

A *polygon* P consists of a set of n points in the plane $p_1, p_2, \ldots p_n$ with (p_i, p_{i+1}) joined by a line segment (an *edge* of the polygon) for all i (subscript arithmetic modulo n) such that the edges form a simple closed curve in the plane. The plane minus those edges is divided into two regions, a bounded region (the *interior* of the polygon) and the unbounded *exterior* of the polygon. We conventionally use "polygon" to refer to the boundary and interior of the polygon together.

We define a *convex* polygon as the intersection of half-planes.

The direction vector along an edge e from vertex p_i to its next neighbour p_{i+1} is denoted $\vec{e}$ or $\vec{p_i}$. We denote the normal to $\vec{e}$ directed towards the interior of the polygon as $\vec{n_e}$. The *inside* of e is the $\vec{n_e}$ side of an infinite line through e.

We denote the x and y coordinates of a point p_i as $p_i(x)$ and $p_i(y)$ respectively. The distance between two points p and q is denoted $\mathrm{d}(p, q)$.

Alternatively, a polygon can be described by the location of the first vertex and specifying a counterclockwise walk around the polygon by giving the tuple (direction vector, distance) for each side. Thus P above can be represented by its first vertex p_1 and each side i by the tuple $(\vec{p_i}, \mathrm{d}(p_i, p_{i+1}))$.

An isothetic rectangle is a rectangle whose sides are parallel to the x and y axes.

We will normally assume any region X of the plane to include its boundary; when we need to distinguish the interior of X from its boundary we will denote the strict interior of X as X^o.

We say that a stabbing polygon P *stabs* a target polygon S if $P \cap S \neq \emptyset$. The polygon P stabs a collection of targets if P stabs each individual target.

3 Minimum-perimeter stabbing with a nearly-similar convex polygon

Let Q be a fixed convex polygon. Let $P = p_1, p_2, \ldots p_k$ be a k-sided convex polygon nearly-similar to Q. If we describe P in (direction, distance) format, we have the position of the starting vertex p_1, a set of direction vectors $\vec{p_1}, \vec{p_2}, \ldots \vec{p_k}$ and corresponding distances $l_1, l_2, \ldots l_k$.

We will develop neccessary and sufficient conditions for P to stab a single conves polygon. These conditions will be in the form of linear inequalities, which we will later use in a linear program for finding the minimum perimeter stabber which stabs all of the polygons.

Lemma 1 *If two convex polygons A and B with positive area do not intersect, then there is an edge of A or B whose supporting line separates A and B.*

Lemma 2 *Two convex polygons A and B intersect if and only if for every edge e of A there is a point of B on the inside of e and for each edge e of B there is a point of A on the inside of e.*

Lemma 2 just says that there must be some point on the inside of each edge e, but in fact, we can identify a particular point that must be on the inside of e. If e is an edge of A, let p be the vertex of B that is most extreme in the direction $\vec{n_e}$. Clearly, p is inside e if any point of B is.

Let $S = s_1, s_2, \ldots s_n$ be an n-sided convex polygon. For each edge i of the stabbing polygon P we have the condition

$$(s - p_i) \cdot \vec{n_{p_i}} \geq 0, \tag{1}$$

where s is the vertex of S that is most extreme in direction $\vec{n_{p_i}}$.

Similarly, we have for each edge i of the target S,

$$(p - s_i) \cdot \vec{n_{s_i}} \geq 0, \tag{2}$$

where p is the most extreme vertex of P in direction $\vec{n_{s_i}}$. Note that the choice of p is not affected by the lengths l_i.

Theorem 1 *If S is fixed in the plane and P is represented as the location of the vertex p_1 and a set of (direction vector, distance) tuples $(\vec{p_i}, l_i)$, then each constraint from (1) and (2) is a linear inequality.*

Theorem 2 *A minimum-perimeter polygon, nearly similar to a fixed convex polygon, that stabs a collection of convex polygons $S_1, S_2, \ldots S_m$ having a total of n vertices can be found in $O(n)$ time.*

Proof: For each S_i, create the constraints as outlined above. If S_i has n_i vertices, this gives $k + n_i$ constraints for S_i, for a total of $km + n$ inequalities. However, each constraint from (1) need not be included for each target polygon; only the most extreme vertex of the entire set of target polygons needs to be used. So we have a total of $k + n$ inequalities.

The perimeter of P is given by

$$\sum_{i=1}^{k} l_k. \tag{3}$$

Each side i of P must have non-negative length, giving another k constraints,

$$l_i \geq 0, \qquad (4)$$

and finally two constraints to force P to be a polygon, i.e., that the sides form a closed loop:

$$\sum_{i=1}^{k} l_i(\vec{p_i}) = 0 \qquad (5)$$

(this gives two equations when put in component form).

The result is a $k+1$-variable linear program with $2k+n+2$ inequalities, which can be solved in time linear in the number of inequalites for fixed k. However, the time required grows exponentially with k [8].

Constructing the constraints takes time $O(kn)$ (linear for bounded k).

The constraints have all been shown to be necessary and together sufficient, and the objective function is the perimeter of the polygon, so the solution is a minimum-perimeter polygon with the given orientations.

4 Stabbing with a minimum-area isothetic rectangle

We show how stabbing a set of convex polygons can be reduced to a problem of stabbing a single centre rectangle CR and a set of four monotone chains in the corner regions around CR.

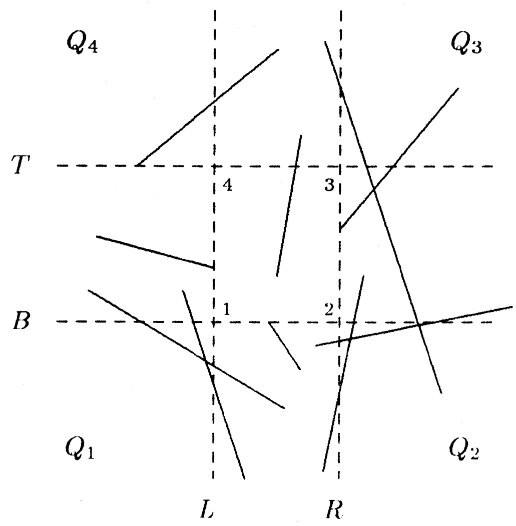
In this section, all rectangles are assumed to be isothetic and include their interior. We construct an algorithm to determine a smallest area rectangle stabber P that stabs a set of m convex polygons. The algorithm will run in $O(n \log n)$ time, where n is the total number of vertices in the target collection.

Given a set S of convex polygons, define T, B, L and R as follows:

$$B = \min_{s \in S}\left(\max_{(x,y)\in s} y\right)$$
$$T = \max_{s \in S}\left(\min_{(x,y)\in s} y\right)$$
$$L = \min_{s \in S}\left(\max_{(x,y)\in s} x\right)$$
$$R = \max_{s \in S}\left(\min_{(x,y)\in s} x\right)$$

It is clear that if $B \geq T$ or $L \geq R$, then there is a rectangle of zero area that stabs S, so without loss of generality we assume that $B < T$ and $L < R$.

Let CR denote the centre rectangle defined by the four lines $y = B, y = T, x = L$ and $x = R$. From the definitions of B, T, L and R it follows that any stabbing rectangle has to contain CR. These four lines divide the plane into nine regions. We will denote the four corners by 1, 2, 3 and 4 and the corner regions by Q_1, Q_2, Q_3 and Q_4, as shown in the following figure.



We can now prove the following lemma:

Lemma 3 *Let $s \in S$. If $s \cap CR = \emptyset$ then there is a unique corner region Q such that*

- $s \cap Q^o \neq \emptyset$
- *s intersects both sides of Q*

The corner region uniquely associated with s will be denoted by Q_s.

Define a new set S' of line segments from S as follows: Polygon s contains two monotone chains connecting the two boundaries of Q_s. Ignore the chain whose edges contain CR in their interior. For each edge in the other chain, add the line segment formed by intersecting the infinite line through the edge with Q_s to S'.

The stabbing problem for S is essentially the same as for S':

Lemma 4 *An isothetic rectangle stabs S if and only if it contains CR and stabs S'.*

In the remainder of this paper we will continue with the problem of stabbing CR and S'.

An infinite line through each $s \in S'$ divides the plane into two halfplanes. The halfplane containing CR will be called the outside of s, the halfplane not containing CR is the inside of s. This is consistent with the use of "interior" for edges of S' derived from convex polygons.

As was shown in the above lemma, each corner region Q has to contain one of the corners of a stabbing rectangle and this corner has to be inside or on each s in Q. Therefore corner i of a stabbing rectangle is in the intersection of Q_i and the inside closed halfplanes of all s in Q_i. This area is convex and is bounded by a polygonal chain. We denote the chain in corner region Q_i by C_i.

The chains can be computed in $O(n \log n)$ time. For example, in corner region Q_1, the intersection points of $s \in S'$ and the line $y = B$ can be sorted on their x-coordinates. This order determines the order in which the line segments appear in the chain C_1. The line segments in Q_1 that do not lie on C_1 can now be found by a left-to-right scanning algorithm.

If a corner of a stabbing rectangle lies on a chain C_i we will call it a *bouncing* corner; otherwise, it is called a *stabbing* corner. The following lemma shows that a smallest stabbing rectangle cannot have too many stabbing corners:

Lemma 5 *A smallest stabbing rectangle cannot have two adjacent stabbing corners*

Lemma 6 *If a smallest stabbing rectangle has two stabbing corners and two bouncing corners, then the bouncing corners are vertices of their respective polygonal chains C_i.*

Corollary 3 *A smallest stabbing rectangle has two opposite bouncing corners on vertices of two chains C_i or has at least 3 bouncing corners.*

A smallest stabbing rectangle can now be found as follows: find a smallest stabbing rectangle with opposite corners on two chains, and find the smallest stabbing rectangle with three bouncing corners. As will be shown, both can be found in $O(n \log n)$ time.

Below is an algorithm to find a smallest stabbing rectangle with corners on vertices of C_1 and C_3.

Input: Four monotone chains of vertices, C_1, C_2, C_3 and C_4, with C_1 and C_2 lying strictly below C_3 and C_4 and with C_1 and C_4 lying strictly to the left of C_2 and C_3.

initialization: (Done only once for all vertices)

For each vertex of $a \in C_1$, extend a line vertically up until it intersects C_4, and store this intersection point with a. Then extend a line horizontally from this intersection point through C_3. Let $X(a)$ be the subset of C_3 vertices that lie on or above this horizontal line. Since C_3 is monotone, this is a contiguous subset of C_3.

Similarly, extend a line horizontally from a to C_2, recording the intersection point, then up through C_3, and let $Y(a)$ be the contiguous subset of C_3 that lies on or to the right of this line.

For each vertex a we have stored the intersection points on C_2 and C_4, as well as $X(a)$ and $Y(a)$. Since the subsets are contiguous it suffices to store two indices.

The symmetrical information is calculated and stored for each vertex $b \in C_3$.

The two chains C_2 and C_4 are not used again.

SmallestRectangle(A, B): This recursive procedure takes two contiguous subchains of C_1 and C_3 as input, which we denote A and B respectively. It will return a smallest-area rectangle with bouncing corners on vertices of A and B, or nil if no such rectangle exists.

step 1: If $|A| \geq |B|$:

Let $a \in A$ be the vertex that most nearly divides A into two equal pieces. Let A_l be the subset of A to the left of a and A_r be the subset of A to the right of a.

If $X(a) \cap Y(a) = \emptyset$, then

- r_1 = SmallestRectangle(A_l, $Y(a)$)
- r_2 = SmallestRectangle(A_r, $X(a)$)
- Return the smaller of r_1 and r_2, if any.

else

- Find a smallest stabbing rectangle with corner a and a corner $b \in B$, subject to the constraint that the rectangle intersects C_2 and C_4. By definition of $X(a)$ and $Y(a)$, we know that such a b must exist and that $b \in X(a) \cap Y(a)$. Call this rectangle r_0.
- Let B_l be the subset of B to the left of b and let B_r be the subset of B to the right of b.
- r_1 = SmallestRectangle(A_l, $\{b\} \cup B_r$)
- r_2 = SmallestRectangle(A_r, $B_l \cup \{b\}$)
- Return the smallest of r_0, r_1 and r_2.

step 2: Otherwise, we have $|B| > |A|$. This is handled symmetrically to step 1, splitting on the middle element of B instead of A.

The correctness of the algorithm follows from the following lemma:

Lemma 7 *Consider some points (a,b) and (c,d) and $X = \{(x_1,y_1),(x_2,y_2),\ldots\}$ with*

$$x_i > a > c, \text{for all } i$$
$$y_i > d > b, \text{for all } i,$$

and a monotone chain lying above (c,d), and a monotone chain lying ot the right of (a,b).

Let P be a smallest stabbing rectangle with corner (a,b) and a corner in X. Let the corner in X be (x_p,y_p). Let Q be a smallest stabbing rectangle with corner (c,d) and a corner in X. Let the corner in X be (x_q,y_q). Then $x_p \leq x_q$ and $y_p \geq y_q$.

Theorem 4 *The algorithm finds a smallest rectangle with bouncing corners on vertices of C_1 and C_3.*

The algorithm does not consider all possible pairs of corners. The pairs it eliminates from consideration in an invocation arise from two possible events, depending on whether $X(a) \cap Y(a) = \emptyset$.

First, if $X(a) \cap Y(a) \neq \emptyset$, then the vertices eliminated in both recursive calls are precisely those that lemma 7 shows cannot be smaller than the rectangle formed by a and b.

In the second case, $X(a) \cap Y(a) = \emptyset$, suppose without loss of generality that $|A| \geq |B|$. For a vertex $a' \in A_l$, we have $Y(a') \subseteq Y(a)$, and the call to SmallestRectangle(A_l, $Y(a)$) has not eliminated any potentially valid rectangles that use a'. Similarly for $a' \in A_r$ we have $X(a') \subseteq X(a)$ and SmallestRectangle(A_r, $X(a)$) has not ignored any potential rectangles.

Since we never eliminate a smallest rectangle from consideration, it will be eventually found if it exists. ■

The input to the algorithm is n points, split into four sets, C_1, C_2, C_3 and C_3. The sets are monotone and ordered. The initialization step can be done in $O(n)$ time by walking along the four chains in order.

Let $T(t)$ be the running time of the recursive algorithm on t vertices.

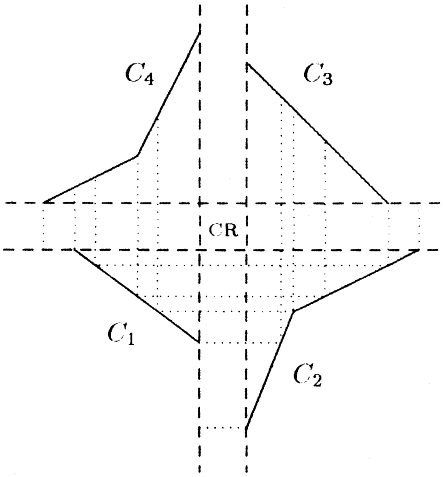
The recurrence is $T(t) = T(k_1) + T(k_2) + c_2 t$ for some $c_2 > 0$, where $t/4 \leq k_1, k_2 \leq 3t/4$ and $k_1 + k_2 \leq t$. It can be shown by induction that for $t \geq 2$, $T(t) \leq c_3 t \log t$, where $c_3 = \max(\max_{0 \leq i \leq 4} T(i), c_2/\log(4/3))$.

A smallest stabbing rectangle with two bouncing and two stabbing corners can be found by applying this algorithm also to chains C_2 and C_4.

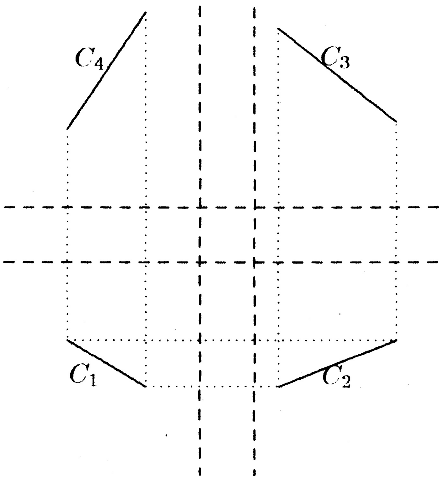
A smallest stabbing rectangle with three bouncing corners can be found as follows. We first find a smallest rectangle with bouncing corners on chains C_1 and C_2 (and also on either C_3 or C_4 or both). We then apply the same algorithm with chains C_3 and C_4. One of these has to be the smallest rectangle with three bouncing corners.

Below is an algorithm to find a smallest stabbing rectangle with corners on C_1 and C_2:

1. For all vertices on C_4, draw vertical lines downward, until they hit C_1; from these intersection points, draw horizontal lines to C_2; from these intersection points, draw vertical lines to C_3. Draw similar lines from C_3 via C_2 and C_1 to C_4. Also draw lines from the vertices of C_1 to C_4 and from the vertices of C_1 via C_2 to C_3. Similarly for C_2 as shown in the next figure.



The problem is now divided into a set of problems as in the following figure. Each of these simple problems has exactly one line segment in each of the four areas $Q_1, \ldots, Q_4$.



2. Find a smallest stabbing rectangle that bounces on C_1 and C_2 for each subproblem.

Since the number of vertices on the chains is less than n, the number of subproblems is $O(n)$.

Since each subproblem can be solved in constant time, this algorithm runs in $O(n)$ time.

5 Acknowledgements

Thanks to Arnold Rosenbloom for bringing this problem to our attention and assisting with early work on this topic.

References

[1] M. Atallah and C. Bajaj, "Efficient algorithms for common transversals." *Information Processing Letters, 25*, 1987, 87-91.

[2] D. Avis and M. Doskas, "Algorithms for high dimensional stabbing problems." Technical Report SOCS 87.2, McGill University, 1987.

[3] D. Avis and R. Wenger, "Polyhedral line transversals in space." Discrete and Computational Geometry, 1989.

[4] B. K. Bhattacharya, J. Czyzowicz, P. Egyed, I Stojmenovic, G. Toussaint, and J. Urrutia, "Computing shortest transversals of sets." *Proc. of the seventh ACM Symp. on Comput. Geometry,* North Conway, 1991, 71-80.

[5] B. Bhattacharya, C. Kumar, and A. Mukhopadhyay, "Computing an area-optimal convex polygonal stabber of a set of parallel line segments." *Proc. 5th Canad. Conf. Comput. Geom.*, Waterloo, Canada (1993).

[6] P. Colley, H. Meijer, D. Rappaport, "Optimal Nearly-Similar Polygon Stabbers of Convex Polygons." *Technical Report CISC-94-357* Queen's University (1994).

[7] H. Edelsbrunner, "Finding transversals for sets of simple geometric figures." *Theoretical Computer Science, 35*, 1985, 31-45.

[8] Herbert Edelsbrunner, "Algorithms in Combinatorial Geometry," EATCS Monographs on Theoretical Computer Science, Volume 10, 1987.

[9] H. Edelsbrunner, H. A. Maurer, F. P. Preparata, A. L. Rosenberg, E. Welzl, D. Wood, "Stabbing line segments." *BIT 22*, 1982, 274-281.

[10] P. Egyed and R. Wenger, "Stabbing pairwise disjoint translates in linear time." *Proc. of the Fifth Symp. on Computational Geometry*, 1989, 364-369.

[11] J. E. Goodman and R. Pollack and R. Wenger, "Geometric Transversal Theory." *DIMACS Technical Report 91-76*, (1991).

[12] Henk Meijer and David Rappaport, "Minimum polygon covers of parallel line segments." *Proc. of the 2nd Annual Canadian Conference in Computational Geometry*, University of Ottawa, 1990.

[13] J. O'Rourke, "An on-line algorithm for fitting straight lines between data ranges." *Comm. ACM 24*, 1981, 574-478.

[14] D. Rappaport, "Minimum Polygon Transversals of Line segments", *Technical Report CISC-92-333* Queen's University (1992).