Optimal Algorithms for Stabbing Polygons by Monotone Chains and Paths[1]

Vladimir Estivill-Castro
Laboratorio Nacional de Informática Avanzada
Rébsamen 80, Xalapa Veracruz 91000, México
vlad@xalapa.lania.mx

Sven Schuierer
Institut für Informatik, Universität Freiburg
Am Flughafen 17, D-79110 Freiburg, Germany
schuierer@informatik.uni-freiburg.de

Abstract

Two algorithms to compute monotone stabbers for convex polygons are presented. More precisely, given a set of m convex polygons with n vertices in total we want to stab the polygons with an x-monotone polygonal chain such that each polygon is entered at its leftmost point and exited at its rightmost point. Since such a stabber does not exist in general, we study two related problems. The first problem requests a monotone stabber that stabs as many convex polygons as possible. The second problem is to compute the minimal number of x-monotone stabbers that are necessary to stab all given convex polygons. We present optimal $O(m \log m + n)$ algorithms for both problems.

KEYWORDS: *Posets, Algorithms, Computational Geometry.*

1 Introduction

Given a set of m objects, an object that intersects each of them is said to be a *stabber* or a *traversal* of the set. There have been many variations of the problem depending on the types of objects, as well as on the form of the answer [5]. In particular, the problem of stabbing convex polygons by monotone chains has important applications in robotics [1, 4]. Since a single stabber may not exist, we may be required to find a stabber that stabs the maximum number of objects or a minimum set of stabbers. Recently, an algorithm that requires $O(n^{2.5})$ time for computing a minimum cardinality set of disjoint x-axis monotone chains stabbing a set of m convex polygons, with n vertices in total, has been developed [3]. An algorithm that requires $O(n^2 \log n)$ time for finding an x-axis monotone chain that stabs the maximum number of convex polygons is given as well [3]. The

[1]This work was partially carried out under grant CONACYT 1707-A9209.

monotone chains must be disjoint since they correspond to trajectories of robotic agents [1].

We consider these two problems and provide optimal algorithms for their solution. Section 2 provides the definitions required for further sections. In Section 4 we present an algorithm that requires $\Theta(n + m \log m)$ time and linear space for finding a monotone chain that stabs a maximum number of convex polygons from a given set. In Section 5 we provide an algorithm that requires $\Theta(n+m \log m)$ time and linear space for finding a minimum stabber set. Our approach is based on a reduction of the problem to the problem of stabbing horizontal segments by x-axis monotone chains that is presented in Section 3. The algorithms in [3] can be used to compute stabbing paths. In Section 6 we also study the two variants of the problem for stabbing paths and we provide $O(n \log m)$ time and linear space algorithms for both of them. Again, this improves the previous bound by at least a linear factor. Finally, in Section 7 we give some final remarks.

2 Definitions and a lower bound

Let $q_1, q_2, \ldots, q_k$ be a set of points in the plane defining a polygonal path or a *chain*. A chain is called *monotone with respect to line l* if any line orthogonal to l intersects the chain in a connected set. A monotone chain with respect to the x-axis is said to be *x-axis monotone* or simply *x-monotone*.

Consider a two dimensional scene consisting of m disjoint convex polygons $P_1, P_2, \ldots, P_m$. Let n be the total number of edges in all polygons from the scene, i.e., n is the size of the scene. A convex polygon P_i is stabbed by an x-monotone chain C, if C enters the polygon P_i at the leftmost point p_L and departs from it at the rightmost point p_R. We want to solve the *Stabber Cover Problem (SCP)*, that is, we want to compute a minimum set of disjoint x-monotone chains that stab all polygons and such that no polygon is stabbed by more than one chain. It is not always possible to solve SCP with only one stabber; however, we are also interested in the *Maximum Stabber Problem (MSP)*, that is, we want to find an x-monotone chain that stabs a maximal number of polygons.

A simple reduction from sorting provides a lower bound of $\Omega(m \log m)$ for solving SCP or MSP. The transformation that requires linear time can be described as follows. Let $X = \langle x_1, x_2, \ldots, x_m \rangle$ be an instance of sorting. In linear time, we construct m small enough squares around the points (x_i, i), for $i = 1, \ldots, m$. Then, a solution to SCP will result in one x-monotone chain such that, tracing the chain will require linear time and will return the x_i in sorted order. An adversary argument gives a lower bound of $\Omega(n)$, and combining these bounds we obtain a lower bound of $\Omega(n + m \log m)$ operations.

3 Stabbing horizontal segments

In this section we show that MSP and SCP for convex polygons have equivalent respective problems for horizontal segments. A preliminary version of

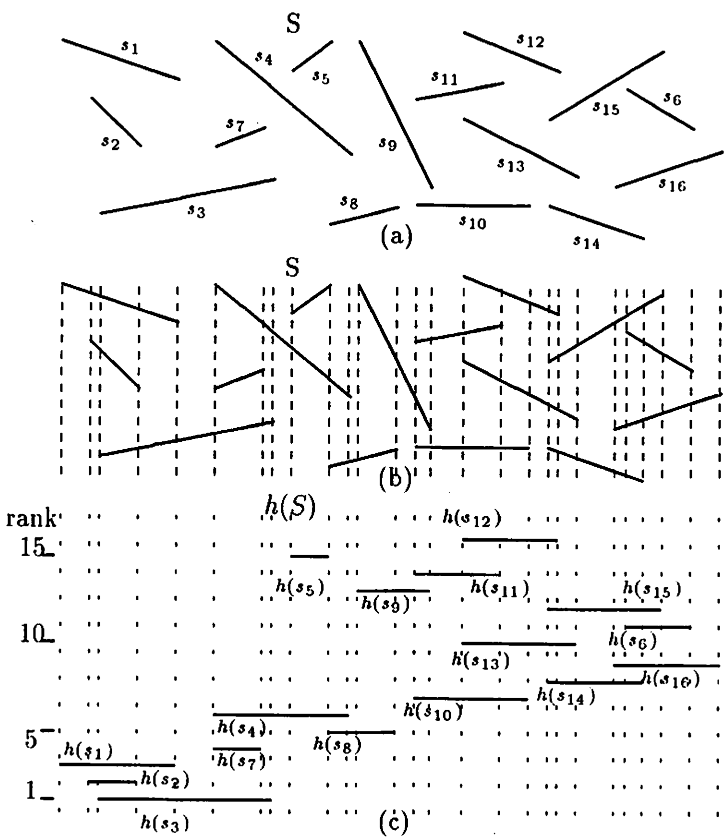

Figure 1: Computing the ranks of the line segments in S and the horizontal line segments in $h(S)$.

this paper describing the results of this and the following two sections was presented at ICCI-95 [2].

The reduction we present proceeds in two steps. First we show that the two stabber problems under consideration can be reduced to respective problems on line segments. Then, we show that the problems under study for a set of line segments are equivalent for a set of horizontal line segments.

First, we observe that it suffices to consider the set S of line segments that connect the left and right endpoints of the convex polygons. More precisely, consider a scene of size n with m polygons $P_1, \ldots, P_m$. In $O(n)$ time we can find the left-most vertex $p_{j_L} = (x_j, y_j)$ and the right-most vertex $p_{j_R} = (x'_j, y'_j)$ for each polygon P_j, $j = 1, \ldots, m$. Let s_j be the line segment that connects p_{j_L} to p_{j_R}. An x-monotone chain C stabs a maximum number of polygons if and only if C stabs a maximum number of segments in S (where a segment $p_L p_R$ is stabbed by C when it is part of C). A set of disjoint monotone stabbers that traverses each P_j from the left-most to the right-most point stabs each line segment s_j and vice versa.

We now reduce MSP and SCP for S to equivalent problems for horizontal segments. We say line segment $s \in S$ is *below* a line segment s' if there is a vertical line l_v that intersects both s and s' and the intersection point of l_v and s is below the intersection point of l_v and s'. The *below* relation induces a partial order $\preceq_{below}$ on the line segments. The partial order $\preceq_{below}$ of S can be easily computed by a vertical plane sweep that needs $O(m \log m)$ time and $O(m)$ space [7, Remark 1, page 69].

We extend the partial order $\preceq_{below}$ to a total order that is consistent with $\preceq_{below}$ and compute (with topological sorting in $O(m)$ time) the position of each segment s_i in this total order, that is, we compute the rank r_i of each segment s_i in this total order. For $j = 1, \ldots, m$, we replace each line segment s_j with an horizontal line segment $h(s_j)$ such that:

- The x-coordinate of the left-most vertex of s_j is x_j, and this is also the x-coordinate of the left-most vertex of $h(s_j)$.
- The x-coordinate of the right-most vertex of s_j is x'_j, and this is also the x-coordinate of the right-most vertex of $h(s_j)$.
- For all vertical lines l_v, s_j intersects l_v below $s_{j'}$ if and only if $h(s_j)$ intersects l_v below $h(s_{j'})$. This can be achieved by setting the y-coordinate of $h(s_j)$ to r_j.

For illustration refer to Figures 1 (a)-(c). We now establish the equivalence of MSP and SCP for S and the transformed set $h(S)$.

Lemma 3.1 *For all $s_i, s_j \in S$, $s_i \preceq_{below} s_j$ if and only if $h(s_i) \preceq_{below} h(s_j)$.*

Proof: If $s_i \preceq_{below} s_j$, since the y-coordinates of $h(s_i)$ and $h(s_j)$ are chosen as the ranks of s_i and s_j and $r_i < r_j$, we have $h(s_i)$ is below $h(s_j)$. To prove necessity, observe that, if $h(s_i) \preceq_{below} h(s_j)$, then there is a vertical line that intersects $h(s_i)$ and $h(s_j)$ but the intersection point with $h(s_i)$ is below the intersection point eith $h(s_j)$. Since the y-coordinates of the $h(s_i)$ is the rank r_i, we have $r_i < r_j$. But then, $s_i \preceq_{below} s_j$. □

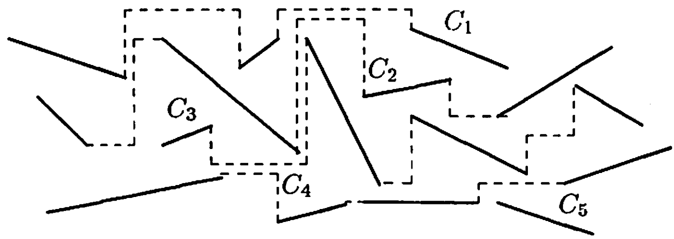

Figure 2: A set of x-monotone chains that covers a set S of line segments.

Lemma 3.2 *There is a set $C_1, \ldots, C_k$ of disjoint x-monotone chains that covers $s_1, \ldots, s_m$ if and only if there is a set of disjoint x-monotone chains $C'_1, \ldots, C'_k$ that covers $h(s_1), \ldots, h(s_m)$. $C'_1, \ldots, C'_k$ can be chosen in such a way that C_i covers s_i if and only if C'_i covers $h(s_i)$.*

Proof: Let $C_1, \ldots, C_k$ be a set of disjoint x-monotone chains that covers $s_1, \ldots, s_m$; see Figure 2. It is easy to see that we can extend each of the

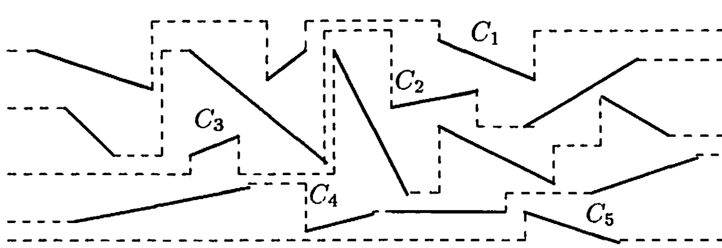

Figure 3: Extending each monotone chain to infinity.

chains C_i to infinity in such a way that each chain remains x-monotone and none of the chains intersect each other; see Figure 3. If l_v is a vertical line, then l_v intersects all the chains independently of its location and the order of the occurrence of the intersection points induces a total order on the chains. This order is the same for all vertical lines since the chains do not intersect one another. We assume that the chains are numbered in order from top to bottom. If a segment s_i belongs to chains C_1, then s_i is a maximal element (there is no other segment above it). Since the rank r_i of each horizontal line segment $h(s_i)$ is consistent with the below relation and the x-interval is the same as that of s_i, $h(s_i)$ is also a maximal element of the poset $(h(S), \preceq_{below})$. Hence, if C_1 consists of the line segments $s_{i_1}, \ldots, s_{i_p}$, then we can choose C'_1 to contain the horizontal line segments $h(s_{i_1}), \ldots, h(s_{i_p})$; see Figure 4 (a) and (b).

Remove from S the line segments $s_{i_1}, \ldots, s_{i_p}$ stabbed by C_1 and remove from $h(S)$ the horizontal line segments $h(s_{i_1}), \ldots, h(s_{i_p})$ stabbed by C'_1, then, by Lemma 3.1, we can recursively apply the construction to $C_2, \ldots, C_k$ and $S - \{s_{i_1}, \ldots, s_{i_p}\}$ to establish a correspondence between $C'_2, \ldots, C'_k$ and $h(S) - \{h(s_{i_1}), \ldots, h(s_{i_p})\}$. □

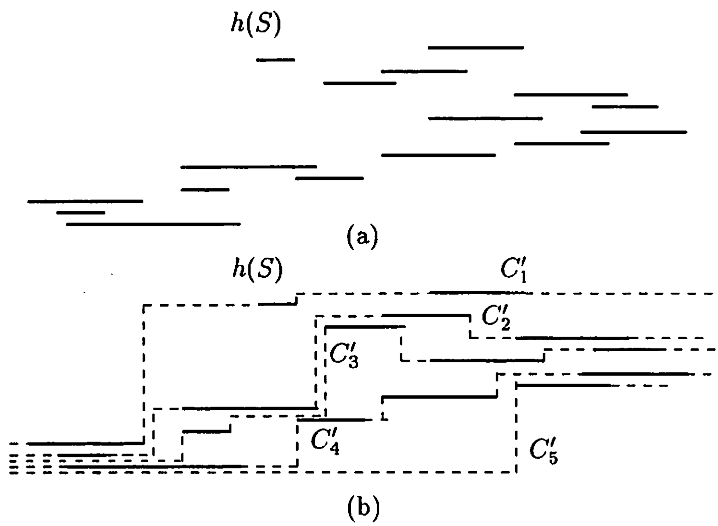

Figure 4: The set $h(S)$ and the corresponding chains $C'_1, \ldots, C'_5$ for $h(S)$.

4 The stabber of most polygons

In this section we provide an algorithm that requires $O(n + m \log m)$ time and $O(n)$ space for finding the x-monotone chain that stabs the maximum number of polygons. During the first step of our algorithm we apply the transformation of Section 3 and, in $O(m \log m)$ time, we transform the Maximum Stabber Problem (MSP) to the equivalent problem for horizontal segments. By Lemma 3.2 we can easily see that, by finding a maximal x-monotone stabber for the horizontal line segments, we obtain a maximal monotone stabber for the original polygons and vice versa.

In order to compute the maximal stabber for the horizontal line seg-

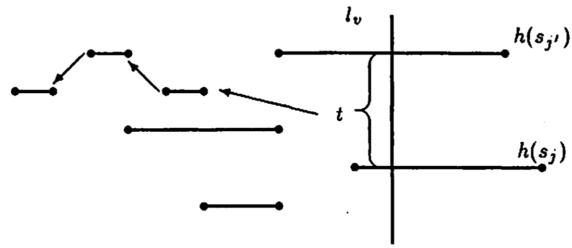

Figure 5: A plane-sweep with a vertical line l_v records the intersection points with horizontal segments and the intervals in l_v between intersecting horizontal segments. For an interval t, we register a x-monotone chain that stabs the most horizontal segments and ends at t.

ments we perform a plane-sweep from left to right with a vertical line l_v, where the event points are determined by the leftmost and rightmost vertex of the horizontal segments. The line l_v is subdivided into vertical segments by the intersection points with the horizontal line segments. The dictionary data structure representing l_v stores two types of objects. The first type of objects are vertical segments t over l_v defined by two consecutive horizontal segments $h(s_j)$ and $h(s_{j'})$ intersecting l_v. The second type of objects are the horizontal segments $h(s_j)$ intersecting l_v. Objects in l_v are ordered by y-coordinate. For each object b in l_v a pointer to the last horizontal segment $h(s_i)$ of the longest x-monotone chain C that can reach b is stored in l_v; see Figure 5. Also, for each object b, the length of the chain C is stored. The pointers and the temporary lengths are updated as l_v advances.

Each time l_v meets a horizontal segment $h(s_{j''})$ between segments $h(s_j)$ and $h(s_{j'})$, the segment t between $h(s_j)$ and $h(s_{j'})$ is divided into two new segments:

- the segment t' between $h(s_j)$ and $h(s_{j''})$, and
- the segment t'' between $h(s_{j''})$ and $h(s_{j'})$; refer to Figure 6 (a).

The segments t' and t'' can be reached by an x-monotone chain of the same length as the one for t. However, the segment $h(s_{j''})$ defines now a x-monotone chain one element larger that the longest chain C that reached t. The pointer of $h(s_{j''})$ is set to the last segment of C, but the length of the chain associated with the object $h(s_{j''})$ is one more than the length associated with t. The dictionary data structure is updated, object t is deleted and objects t', t'' and $h(s_{j''})$ are inserted.

Each time l_v leaves a horizontal segment $h(s_{j''})$ between two horizontal segments $h(s_j)$ and $h(s_{j'})$, the segments t' and t'' become a single segment t determined by the intersection of l_v with $h(s_j)$ and $h(s_{j'})$; see Figures 6 (b). In this case, the horizontal segment $h(s_{j''})$ is deleted from l_v as well as t' and t''. The new segment t is inserted in l_v. The pointer at t to the last

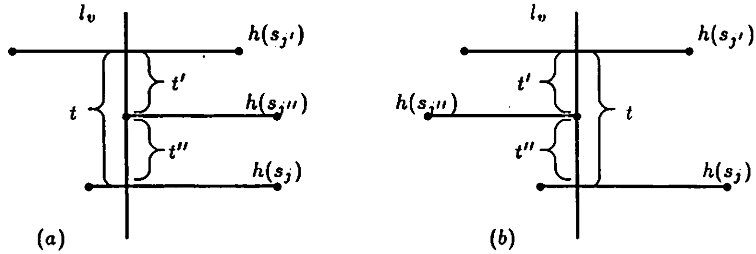

Figure 6: A plane-sweep with a vertical line l_v. In case (a) a new horizontal segment is found. In case (b) a horizontal segments leaves.

segment of the longest x-monotone chain ending at t is the pointer of the longest of the chains that reach t', t'' and $h(s_{j''})$. The length of the chain ending at t is the maximum of the lengths of the chains ending at t', t'' and $h(s_{j''})$.

The line l_v is implemented as a dictionary data structure that takes $O(\log m)$ time for an update and stores the horizontal line segments that intersect l_v in the order of their y-coordinate. We obtained the following result.

Theorem 4.1 *MSP can be solved optimally in $\Theta(n + m \log m)$ time by reducing it to computing a Maximum Stabber for horizontal segments.*

5 The stabber cover problem

In this section we show that SCP is equivalent to the *Minimal Slot Assignment Problem* [6]. The *Minimal Slot-Assignment-Problem* (MSAP) for a set of horizontal line segments in the plane can be stated as follows:

> Given a set of m non-overlapping horizontal line segments in the plane find the smallest integer s which satisfies the condition that the segments can be assigned slot numbers in the range $[1, s]$ such that the total order of line segments at each vertical sweep line position equals the total order given by their slot number assignments.

MSAP can be solved in optimal $O(m \log m)$ and $O(m)$ space [6]. An intuitive description of the algorithm that solves MSAP is as follows. Assume snow is falling vertically down on the line segments from above. Some line segments (for example, $h(s_1)$, $h(s_5)$ and $h(s_{12})$ in Figure 1 (c) and Figure 4 (a)) are completely covered with snow, while the other segments are either partially covered or completely clean (snow on horizontal segments does not slide horizontally). The completely covered segments are assigned slot number 1 and are removed. The process is now repeated for slot number 2, and so on until all segments have been assigned slot numbers. The algorithm consists of a bottom-to-top sweep of the horizontal line segments. The horizontal sweep line stores the assigned slot numbers of the most recent line segments it has met. The sweep line is assigned slot number 0 initially, and when a line segment is met, the slot number assigned to it is one plus the maximum slot number of the line segment it covers. Again, a dictionary data structure represents the horizontal seep line as it moves up in the scene and the line segments are stored by the order of their x-coordinate. An incoming line segment is treated as a range query with deletion. The range query determines all line segments it covers completely together with the, at most, two line segments it partially covers. All completely covered intervals are deleted, the maximum of their slot numbers being noted. The, at most, two segments partially covered are shrunk, that is either a left endpoint moves further right or a right endpoint further left. The maximum of all effected line segments is known and this value plus one is the slot number of the new line segment which is now inserted.

We reduce SCP for convex polygons to MSAP in two steps. First we use the reduction in Section 3 to reduce SCP on convex polygons to SCP on a set of horizontal line segments. Note that, the bottom-to-top plane sweep illustrates that two horizontal line segments can be connected by an x-monotone chain if they are assigned the same slot number. Thus, assigning a slot number to each of the horizontal line segments is equivalent to connecting them with monotone chains. Hence, the Minimal Slot-Assignment Problem solves SCP and computing a stabbing cover by x-monotone chains of a set of $O(m)$ horizontal segments can be done in $O(m \log m)$ time and linear space [6]. Therefore, we have obtained the following result.

Theorem 5.1 *SCP can be solved optimally in $\Theta(n + m \log m)$ time by reducing it to the Minimal Slot-Assignment Problem.*

6 Stabbing paths

The algorithms in the previous sections do not ensure that a x-monotone chain C only intersects the polygons that it stabs. For example, in the scene depicted in Figure 7, our algorithm for the maximum stabber computes correctly and efficiently the sequence $\langle P_1, P_2, P_3, P_4 \rangle$ as the sequence of polygons stabbed by a maximum stabber. However, the line segment from the rightmost point of P_i to the leftmost point of P_{i+1} (for $i = 1, 2, 3$) intersects P_5.

A preliminary version of this paper describing the results of previous sections was presented at ICCI-95 [2]. These results provide enough un-

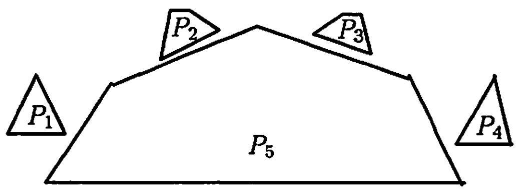

Figure 7: A scene with 5 polygons where x-monotone stabbers may not be paths.

derstanding to obtain algorithms to compute stabbing paths in the original scene. In this section we modify the algorithms previously given to compute paths that a mobile robot can follow. Paths are x-monotone chains that do not collide with other polygons in the scene. We think of the convex polygons as rooms with doors at their leftmost and rightmost vertices. The polygons constitute obstacles for the paths at other points that are not the doors. Figure 7 also shows that the output of stabbing paths has size $\Omega(m+n)$.

6.1 The stabbing monotone path of most polygons

We first present an algorithm for the MSP problem where the polygonal chain must only intersect polygons that it actually stabs. The algorithm performs a plane-sweep from left to right in the original scene, and the event points are all the n points in the scene. As before, the vertical sweep line l_v stores two kinds of objects (see Figure 8). The first type of objects are the vertical segments exterior to all polygons. The second type of objects are the interior vertical line segments that constitute the intersection of l_v and one convex polygon in the scene.

An object b in l_v has associated with it the number of polygons stabbed by the maximum x-monotone chain that ends at b and does not intersect other polygons. The objects also store an associated pointer to the previous joint point of the chain. Those segments in l_v that are in the interior of a polygon (for example, segments P_i' and P_j' in Figure 8) are treated in a similar way as horizontal line segments where treated in Section 4. That is, the length of a chain that reaches them is one element larger after departing them. For the type of objects that are the intersection of a polygon P_i and l_v, the pointer associated with them is just the leftmost point of the polygon P_i (since all P_i are convex).

However, for an object s that is a segment in l_v corresponding to an interval between polygons, the pointer associated with s points to the head

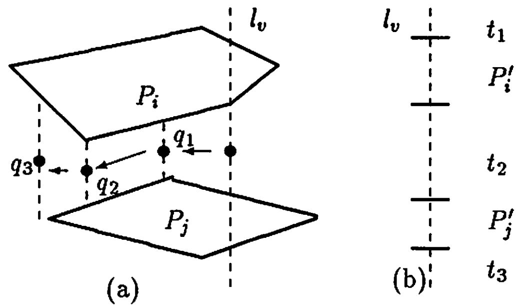

Figure 8: The intersection of a vertical line l_v and the scene results in two types of objects: line segments that are intervals of l_v inside a polygon, and line segments that are intervals of l_v exterior to all polygons.

of a linked list that defines the x-monotone path passing through the corridors defined by the exterior of the polygons. For example, in Figure 8, object t_2 has a pointer to point q_1, that is the head of the list q_1, q_2 and q_3 that defines a x-monotone polygonal path in the corridor between P_i and P_j.

As the line l_v advances we have the following cases for an event.

1.– The line l_v arrives at an event point that is not a leftmost or rightmost point of a polygon; see Figure 9 (a).

2.– We arrive at an event point that is a rightmost point of a polygon; see Figure 9 (b).

3.– We arrive at an event point that is a leftmost point of a polygon in the scene; see Figure 9 (c).

In Case 1, we simply compute a new point q'_1 halfway between the two polygons and on l_v. The point q'_1 is inserted at the head of the list previously headed by q_1 and the new object in l_v has q'_1 as the new associated point.

Case 2 is similar to the case of Figure 6 (b); three objects are removed from l_v and a new object t is inserted to replace them. The maximum number of stabbed objects is determined by choosing the largest among the values from the chain through q_u, the chain through q_d or the diagonal of the polygon between its leftmost and rightmost points. If the stabber chain of most polygons reaching the new position of l_v is obtained through the polygon, the rightmost point p_R is the point associated with the new segment in l_v. The point p_R is inserted, as the new head, in the list with head p_L.

If the stabber chain of most polygons is obtained through q_u, a point q'_u with same x-coordinate as p_R but slightly larger y-coordinate and such

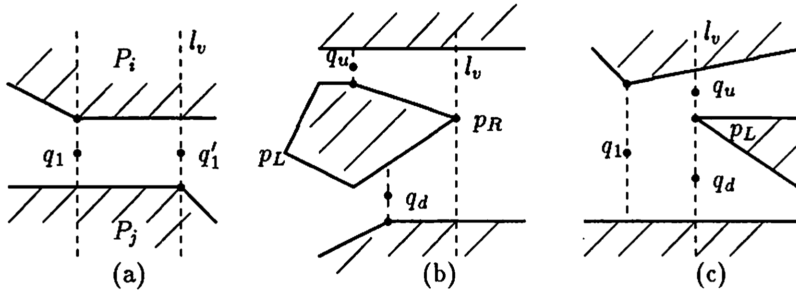

Figure 9: The three cases that occur during the line sweep algorithm for computing stabbing paths.

that the line segment $q_u q_u'$ does not intersect any polygon is added as the point associated with the new object t. The point q_u' is inserted, as the head, of the list with head q_u. The situation is symmetric if the maximum stabber chain is obtained trough q_d, but the new point is just below p_R.

Case 3 is similar to the case of Figure 6 (a). Three new objects replace a previous object t. The middle object is a polygonal object with associated point, the event point p_L. The other two objects have associated points halfway between the edges of the corridors of the polygons. All points become heads of lists pointing to the point q_1 associated with the old object t; however, the maximum number of stabbed polygons is one more for the middle object, while for the other two objects, it remains the same.

It is not hard now to see that the vertical line l_v preserves the invariant that, for each object b in l_v, we can retrieve the x-monotone chain that stabs the maximum number of polygons without invading other polygons and ending at b.

This provides correctness for the algorithm. Since the scene consists of m polygons, at most $O(m)$ objects are ever stored in l_v, and since there are n event points the algorithm requires $O(n \log m)$ time. Since there are at most three elements in the linked lists representing monotone chains for each event point, the algorithm requires $O(n)$ space. We have proved the following result.

Theorem 6.1 *A Maximum Stabbing Path that does not collide with other polygons can be computed in $O(n \log m)$ time and $O(n)$ space.*

6.2 The minimum stabbing cover with paths

A minimum stabbing cover by x-monotone chains as computed is Section 5 results in disjoint chains among each other, but each chain may intersect other polygons in the scene besides the ones it stabs. Again, Figure 7 illustrates that two x-monotone chains cover the 5 polygons. The top-most chain covers P_1, P_2, P_3 and P_4 while the second one covers P_5. A straight line from the rightmost point p_{1_R} of P_1 to the leftmost point p_{2_L} of P_2 intersects P_5. Fortunately, the transformation from a set of stabbing chains to a set of stabbing paths can be easily obtained by a plane-sweep from left to right that makes sure each chain avoids the polygons it does not stab. The details of this transformation are similar to the previous subsection, and are omitted. By Theorem 5.1, the SCP solution can be obtained in $O(n + m \log m)$ time but the transformation takes $O(n \log m)$ time. Thus we have the following result.

Theorem 6.2 *A Minimum Set of x-monotone Stabbing Paths of a scene with m convex polygons defined by n points can be computed in $O(n \log m)$ time and $O(n)$ space.*

7 Final remarks

We have presented optimal algorithms for computing a stabbing cover by x-monotone chains of convex polygons and for computing the x-monotone chain that stabs the most polygons. Our results improve, by at least a linear factor, previously known results.

A monotone chain C with respect to a line l has the property that lines orthogonal to l intersect C in a connected set. We say that a a chain C is xy-monotone if any line parallel to the x-axis or the y-axis intersects C in a connected set. It is not hard to see that our algorithms for stabbing horizontal segments extend to computing covers of xy-monotone chains that intersect all segments horizontally. Natural open question regarding stabbing covers that we leave for further research are:

1. Can SCP or MSP be solved efficiently for convex polygons using xy monotone chains?

2. Given a set of orientations ω can minimum stabbing covers of ω-monotone chains be efficiently found for convex polygons?

3. If we use our algorithm, first in the x-direction, and afterwards in the y-direction for computing two covers by xy-monotone chains of the same set of polygons, and take the smallest, how far is this from the optimum?

Similar open question for maximum covering stabbers are also left for further research.

References

[1] E. M. Arkin, R. Connelly, and J. S. B. Mitchell. On monotone paths among obstacles, with applications to planning assemblies. In *Proc. 5th Annual ACM Sympos. Comput. Geom.*, pages 334–343, 1989.

[2] V. Estivill-Castro and S. Schuierer. Optimal Algorithms for Stabbing Polygons by Monotone Chains. *Proceedings of the 7th International Conference of Computing and Information*, July 1995.

[3] L.P. Gewali. Stabbing polygons by monotone chains. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 14:79–85, 1993.

[4] M. T. Goodrich and J. Snoeyink. Stabbing parallel segments with a convex polygon. *Comput. Vision Graph. Image Process.*, 49:152–170, 1990.

[5] W. D. Jones and Y. Ke. Maximal segment-stabbing of segments in the plane. In *Proc. 2nd Canad. Conf. Comput. Geom.*, pages 85–88, 1990.

[6] Th. Ottmann and D. Wood. Space-Economical Plane-Sweep Algorithms. *Computer Vision, Graphics, and Image Processing*, 34:35–51, 1986.

[7] F. P. Preparata and M. I. Shamos. *Computational Geometry: an Introduction.* Springer-Verlag, New York, NY, 1985.

[8] D. Wood. *Data Structures, Algorithms and Performance.* Addison-Wesley Publishing Co., Reading, MA, 1993.