Finding the Largest Axis-Aligned Rectangle in a Polygon in $O(n \log n)$ time

Ralph P. Boland*† Jorge Urrutia‡

Abstract

We consider the problem of finding the largest area axis-aligned rectangle contained in an n vertex polygon. We present an algorithm that solves this problem in $O(n \log n)$ time. This is an improvement by a factor of $O(\log n)$ over the best known algorithm. Our method of achieving this improvement is noteworthy. The previous algorithm constructs a cover of the polygon by a collection of subpolygons, of the type we call SAM polygons, such that any rectangle in the polygon is contained in one of the SAM subpolygons. This cover requires $O(n \log n)$ time and space to construct. Our algorithm also constructs a cover of the polygon by SAM subpolygons such that any rectangle in the polygon is contained in one of the subpolygons but does so using only linear time and space.

1 Introduction

An axis-aligned rectangle is a rectangle whose sides are parallel to the x-axis or y-axis. In this paper we present an algorithm for finding the largest axis-aligned rectangle contained in a simple polygon. Henceforth, all rectangles used in this paper are assumed to be axis-aligned. As in [4], we refer to this problem as the *LR problem*. In [4], the authors showed how to solve the LR problem on an n vertex simple polygon P in $O(n \log^2 n)$ time. Their method generates a set of simpler subpolygons of P such that any rectangle in P is contained in one of them. It then finds the largest rectangle in each of the simpler subpolygons and selects the largest one. In this paper we solve the LR problem using the same approach except that the total size of the simpler subpolygons that we generate is smaller by a factor of $O(\log n)$; the result is an algorithm that solves the LR problem in $O(n \log n)$ time.

For any set of polygons $\wp$ we denote $\|\wp\| = \sum_{Q_i \in \wp} |Q_i|$. Let Π be a cover of P by subpolygons. Let R be any rectangle contained in P such that the number of elements of Π needed to cover R is maximized. We denote this number by $\mathsf{C}_\square(\Pi)$. Using the terminology in [1], we call $\mathsf{C}_\square(\Pi)$ the *rectilinear visibility cover number* of Π. If $\mathsf{C}_\square(\Pi) = o(|\Pi|)$ then we call Π a *rectilinear visibility cover* of P. All covers in this paper, that are not partitions, will have a rectilinear visibility cover number of 1. Both this paper and [4] use the following observation.

Observation 1.1. *Assume, for some function g, that the LR problem can be solved, for any element Q of Π, in $O(|Q| * g(|Q|))$ time. If $\mathsf{C}_\square(\Pi) = 1$ and Π can be constructed in $O(\|\Pi\| * g(n))$ time then the LR problem can be solved on P in $O(\|\Pi\| * g(n))$ time.*

Proof. First, build Π in $O(\|\Pi\| * g(n))$ time. Then find a largest rectangle in each element of Π in $\sum_{Q_i \in \Pi} O(|Q_i| * g(|Q_i|)) = O(\|\Pi\| * g(n))$ time and select the largest one. □

We call a polygon that is either a y-monotone polygon that admits a vertical chord joining a highest point on its boundary with a lowest point on its boundary or an x-monotone polygon that admits a horizontal chord joining a leftmost point on its boundary with a rightmost point on its boundary a *separated axis monotone polygon* or *SAM polygon*.[1] The following lemma is proved in [4].

Lemma 1.2. *The LR problem for any m vertex SAM polygon can be solved in $O(m \log m)$ time.*

In [4], the authors first showed how to construct a cover, say Π, of P, such that $\|\Pi\| = O(n \log n)$, the elements of Π are SAM polygons, and $\mathsf{C}_\square(\Pi) = 1$. Thus, by Observation 1.1 and Lemma 1.2, the LR problem on P can be solved in $O(n \log^2 n)$ time. Their algorithm also works for polygons with holes and in the same time complexity (where n is then the total number of vertices in P and its holes). They also provided a survey of previous results on this problem and gave some applications for the case that there are holes in P.

Our contribution to this problem is a method of constructing a cover, say Π', of P such that $\mathsf{C}_\square(\Pi') = 1$ and the elements of Π' are SAM polygons and, if P is not rectilinear, quadrilaterals; note that quadrilaterals cause no negative consequences since the LR problem is easily solved on a quadrilateral. Our improvements,

*University of New Brunswick, Canada
email: rboland@unb.ca

†Funded in part by NSERC and the University of New Brunswick

‡Instituto de Mathematicas Universidad Nacional Autonoma de Mexico
email: urrutia@math.unam.mx

[1] In [4] these polygon classes are called vertically separated horizontally convex polygons and horizontally separated vertically convex polygons respectively.

over the algorithm in [4], are that $\|\Pi'\| = O(n)$ and our algorithm constructs Π' in linear time. Thus, by Observation 1.1 and Lemma 1.2, the LR problem on P can be solved, using our algorithm, in $O(n \log n)$ time; an improvement by a factor of $O(\log n)$.

In order to construct Π' it is necessary to construct a horizontal trapezoidization of P and a vertical trapezoidization of P. Either trapezoidization of P can be constructed in linear time by a complex algorithm [2] or in $O(n \log n)$ time by a number of algorithms; see for example [3], [5], and [6]. If an $O(n \log n)$ time trapezoidization algorithm is used then the overall time complexity of our algorithm remains at $O(n \log n)$.

All polygons used in this paper are assumed to be simple and without adjacent collinear edges. We need some basic terminology. Let $P = \{v_0, v_1, ..., v_{n-1}\}$ be any polygon. We assume that P has a counterclockwise orientation and that the edges of P are directed accordingly. To simplify matters, P is assumed to have an edge, denoted base(P) and called the *base edge* of P, which is axis-aligned and furthermore, base(P) $= \overline{v_0 v_1}$.

Let γ be any chord of P. We denote by P_γ the subpolygon of P determined by γ that does not contain v_0; if v_0 is an endpoint of γ then P_γ denotes the subpolygon that contains v_{n-1} on its boundary. Let ϕ be a chord or edge of P. We denote by $\text{vis}_\perp(\phi, P)$ the subpolygon, say Q, of P such that, for any point p interior to Q, there is a point $q \in \phi$ where the line segment $\overline{pq}$ is perpendicular to ϕ and interior to P. When P is implied we abbreviate this to $\text{vis}_\perp(\phi)$. The following idea is used in [4] and by our algorithm.

Observation 1.3. *Let γ be any axis-aligned chord of a polygon. Then $vis_\perp(\gamma)$ is a SAM polygon.*

The remainder of this paper is organized as follows.

In Section 2 we define rectilinear tree partitions. We then show, for any rectilinear tree partition, say Π, of a polygon P, that any rectangle interior to P that intersects more than one element of Π must be contained in one of a set of SAM subpolygons of P determined by the chords used to construct Π. This set of SAM subpolygons can be constructed in linear time and space.

In Section 3 we solve the LR problem for rectilinear polygons using a kind of rectilinear tree partition called a histogram partition; see [8]. The results of this section can be modified to deal with general polygons. This will be dealt with in the full paper.

2 Rectilinear Tree Partitions

Let P be a polygon and Q be any subpolygon of P such that the edges of Q are either chords of P or parts of edges of P; however, either all or none of the interior of base(P) is on the boundary of Q. Then we call Q a *cut subpolygon* of P. We further denote by door(Q) the chord of P on the boundary of Q such that Q is contained in $P_{\text{door}(Q)}$; if there is no such chord then door(Q) = base(P). We denote by windows(Q) the remaining edges of Q that are chords of P and refer to them as the *windows* of Q. We assume that the endpoints of door(Q) are the first two vertices of Q visited during a counterclockwise traversal of Q and that base(Q) = door(Q).

Let U be any partition of P into cut subpolygons. We call U a *tree partition* of P. This term is appropriate since the graph in which each vertex corresponds to an element of U and the edges of the graph correspond to coincident elements of U is a tree. We denote by $\widehat{\Gamma}(U)$ the chords of P used to construct U. If the elements of $\widehat{\Gamma}(U)$ are all axis-aligned, and no axis-aligned chord of P intersects both the door and a window of the same element of U then we call U a *rectilinear tree partition*. Rectilinear tree partitions have a number of properties useful in solving the LR problem. Let Π be any rectilinear tree partition of P.

Lemma 2.1. *Any axis-aligned chord of P intersects at most 2 elements of $\widehat{\Gamma}(\Pi)$ and 3 elements of Π.*

Proof. Let γ be any axis-aligned chord of P such that γ intersects at least 2 elements of Π and let T be any element of Π such that γ intersects a window of T. Note that, for any element Q of Π, γ cannot intersect both door(Q) and any element of windows(Q), nor can γ intersect more than 2 windows of Q. Therefore, since the windows of T are doors of other elements of Π, it must be that γ cannot intersect any window of any element of Π except those of T. Thus, γ can intersect at most 2 elements of $\widehat{\Gamma}(\Pi)$ and so γ can intersect at most 3 elements of Π. □

Let $S = \{\text{vis}_\perp(\gamma) : \gamma \in \widehat{\Gamma}(\Pi)\}$ with duplicates allowed and $S' \subset S$ be the set of elements of S that are interior to other elements of S; in the case that S contains 2 copies of the same subpolygon only one is put into S'. We denote SAMPC(Π) $= S - S'$; see Fig. 1.

Lemma 2.2. $\mathtt{C}_\square(SAMPC(\Pi) \cup \Pi) = 1$.

Proof. Let R be any rectangle contained in P but not contained in any element of Π. Then R intersects some chord $\gamma \in \widehat{\Gamma}(\Pi)$. But then $R \subseteq \text{vis}_\perp(\gamma)$ since 2 of the edges of R are perpendicular to γ. But $\text{vis}_\perp(\gamma) \in$ SAMPC(Π) so R is covered by a single subpolygon of SAMPC(Π) $\cup \Pi$. □

Lemma 2.3. $\|SAMPC(\Pi)\| \leq 8|P|$.

Proof. For any chord γ of $\widehat{\Gamma}(\Pi)$ and vertex v of $Q = \text{vis}_\perp(\gamma)$ let $l(\gamma, v)$ be the line through v and perpendicular to γ and $V(\gamma, v)$ be the vertices of Q belonging to line $l(\gamma, v)$. If there are elements of $V(P) \cap V(\gamma, v)$ on

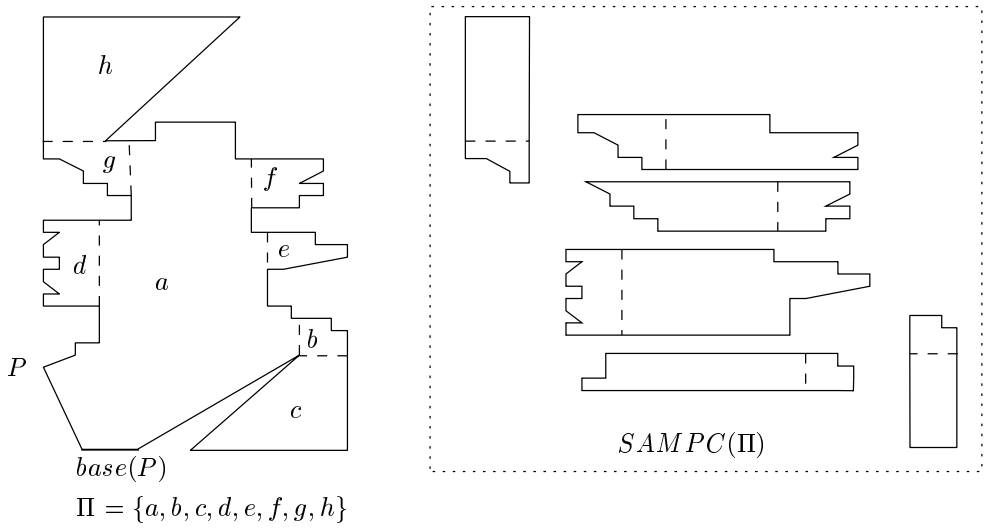

Figure 1: *A rectilinear tree partition* Π *of a polygon* P *and the partial cover of* P *by the SAM subpolygons* $SAMPC(\Pi)$

both sides of γ then let $c(\gamma, v)$ be the number of elements of $V(\gamma, v)$ on the same side of γ as v; otherwise let $c(\gamma, v) = |V(\gamma, v)|$. Then $c(\gamma, v_\gamma) \leq 2$.

Let u be any vertex of P. Then, by Lemma 2.1, there exists at most 2 vertical elements of $\widehat{\Gamma}(\Pi)$, say γ_1 and γ_2, such that u is a vertex of $\text{vis}_\perp(\gamma_1)$ and a vertex of $\text{vis}_\perp(\gamma_2)$. Now $c(\gamma_1, u) + c(\gamma_2, u) \leq 4$. By applying these counting arguments to every vertex of P, we count $4|P|$ vertices in total. Applying the same arguments to the horizontal elements of $\widehat{\Gamma}(\Pi)$, counts another $4|P|$ vertices. The lemma now follows since we have counted each vertex of every element of Π at least once. □

The result of Lemma 2.3 is not tight. For a tighter bound see [1].

Lemma 2.4. *Given* Π, $SAMPC(\Pi)$ *can be constructed in* $O(|P|)$ *time.*

Proof. Let T_h and T_v be the horizontal trapezoidization and vertical trapezoidization of P respectively. Now, for each vertical chord γ_i of $\widehat{\Gamma}(\Pi)$, we construct $Q_i = \text{vis}_\perp(\gamma_i)$ using the trapezoids of T_h contained in Q_i or intersecting Q_i. This is easily done in $O(|Q_i|)$ time. We must also delete duplicates and subpolygons contained in larger subpolygons but this is easily done in $O(|P|)$ time.[2] Next, we repeat this construction for the horizontal chords of $\widehat{\Gamma}(\Pi)$ using T_v. Thus, constructing SAMPC(P) requires $O(\|\text{SAMPC}(P)\| + |T_h| + |T_v|) = O(|P|)$ time by Lemma 2.3. □

Clearly, from Observation 1.1, and Lemmas 1.2, 2.2, and 2.4, we can solve the LR problem on P in $O(n \log n)$ time by constructing a rectilinear tree partition of P in $O(n \log n)$ time in which the LR problem can be solved on the elements of the partition in $O(n \log n)$ time.

[2] By being more careful, we can avoid generating duplicates and subpolygons contained in other subpolygons in the first place.

3 The LR Problem for Rectilinear Polygons

Let P be a polygon. We say that P is a *histogram polygon* if $P = \text{vis}_\perp(\text{base}(P))$. We point out that in [7] a histogram polygon is called a normal histogram polygon. We also point out that, unlike standard histograms, these polygons need not be rectilinear. We denote $\text{hist}(P) = \text{vis}_\perp(\text{base}(P))$. Let γ be any axis-aligned chord of P and $Q = P_\gamma$. Note that $\text{hist}(Q) = \text{vis}_\perp(\text{door}(Q), Q)$. We point out that any histogram polygon is a degenerate SAM polygon. We solve the LR problem for rectilinear polygons using histogram partitions; see [9].

Definition 3.1. [3] If $P = \text{hist}(P)$ then

$$\pi_{\text{hist}}(P) = \{P\};$$

otherwise

$$\pi_{\text{hist}}(P) = \{\text{hist}(P)\} \cup \Big(\bigcup_{\gamma_i \in \text{windows}(\text{hist}(P))} \pi_{\text{hist}}(P_{\gamma_i}) \Big).$$

We refer to $\pi_{\text{hist}}(P)$ as the *histogram partition* of P; see Fig. 2. The histogram partition of a rectilinear polygon can be constructed in linear time [8]; see also [9].

Theorem 3.2. *The LR problem can be solved on an* n *vertex rectilinear polygon in* $O(n \log n)$ *time.*

[3] This definition is based on a similar definition in [9].

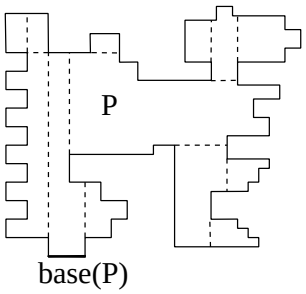

Figure 2: *A rectilinear polygon P and $\pi_{\text{hist}}(P)$*

Proof. Let Q be an n vertex rectilinear polygon. Let $\Pi = \text{SAMPC}(Q) \cup \text{hist}(Q)$. Now $\pi_{\text{hist}}(Q)$ is clearly a rectilinear tree partition and each element of $\pi_{\text{hist}}(Q)$ is a SAM polygon so $\text{SAMPC}(Q) \cup \pi_{\text{hist}}(Q)$ is a cover of Q by SAM polygons and furthermore, $\mathbb{C}_{\square}(\text{SAMPC}(Q) \cup \pi_{\text{hist}}(Q)) = 1$ by Lemma 2.2. However, this considers more subpolygons than necessary. We observe that every element of $\pi_{\text{hist}}(Q)$, except $\text{hist}(Q)$, is contained in an element of $\text{SAMPC}(Q)$. Thus, Π is also a cover of Q by SAM polygons and $\mathbb{C}_{\square}(\Pi) = 1$. Furthermore, Π can be constructed in $O(|Q|)$ time by Lemma 2.4. Thus, by Observation 1.1 and Lemma 1.2, we can solve the LR problem on Q in $O(n \log n)$ time. □

This is an improvement by a factor of $O(\log n)$ over the algorithm in [4]. We made this improvement by replacing one rectangle visibility cover by another more efficient one in that the new cover can be constructed in $O(\log n)$ less time and uses $O(\log n)$ less space. However, since it is not necessary to store all of the cover at once in order to solve the LR problem and the algorithm in [4] avoids doing so, there is no corresponding improvement in space efficiency.

In the full paper we generalize the results of this section to general polygons and solve the LR problem for this case in the same bounds. The generalization involves partitioning the polygon into pseudo-histogram polygons (see [7]), instead of histogram polygons. The difficulty with this generalization is in dealing with rectangles that are entirely contained by a single pseudo-histogram of the partition. We show how to construct a cover of a pseudo-histogram by SAM polygons and quadrilaterals in linear space and time such that the rectilinear cover number is 1. Using this result and the techniques described here we can construct a rectangle visibility cover of a general polygon by SAM polygons and quadrilaterals in linear space and time such that the cover number is 1. This allows us to solve the LR problem for general polygons in the same complexity as for rectilinear polygons.

References

[1] R. P. Boland. *Polygon Visibility Decompositions with Applications.* Ph.D. thesis, submitted January, 2000, University of Ottawa., Ottawa, Ontario, Canada, 2000.

[2] Bernard Chazelle. Triangulating a simple polygon in linear time. *Discrete Comput. Geom.*, 6(5):485–524, 1991.

[3] Bernard Chazelle and Janet Incerpi. Triangulation and shape-complexity. *ACM Trans. Graph.*, 3(2):135–152, 1984.

[4] Karen Daniels, Victor Milenkovic, and Dan Roth. Finding the largest area axis-parallel rectangle in a polygon. *Comput. Geom.*, 7(1-2):125–148, 1997. Fifth Canadian Conference on Computational Geometry (Waterloo, ON, 1993).

[5] A. Fournier and D. Y. Montuno. Triangulating simple polygons and equivalent problems. *ACM Trans. Graph.*, 3(2):153–174, 1984.

[6] Martin Held. Efficient and reliable triangulation of polygons. In *Proc. Comput. Graphics Internat. 1998*, pages 633–643, June 1998.

[7] Rolf Klein and Andrzej Lingas. A linear-time randomized algorithm for the bounded Voronoi diagram of a simple polygon. In *Proc. 9th Annu. ACM Sympos. Comput. Geom.*, pages 124–132, 1993.

[8] C. Levcopoulos. *Heuristics for Minimum Decompositions of Polygons.* Ph.D. thesis, Linköping Studies in Science and Technology, Linköping, Sweden, 1987. No. 155.

[9] S. Schuierer. An optimal data structure for shortest rectilinear path queries in a simple rectilinear polygon. *Internat. J. Comput. Geom. Appl.*, 6:205–226, 1996.