Approximating Convex Polygons by Histogons[*]

Jaehoon Chung[†] Sang Won Bae[‡] Chan-Su Shin[§] Sang Duk Yoon[¶] Hee-Kap Ahn[‖]

Abstract

We study the problem of finding the largest inscribed histogon and the smallest circumscribed histogon for a convex polygon. A histogon is an axis-aligned rectilinear polygon such that every horizontal edge has an integer length. Depending on whether the horizontal width of a histogon is predetermined or not, we consider four different versions of the problem and present exact algorithms.

1 Introduction

Motivated by optimization problems in shape analysis, classification, and simplification [1, 2], we consider two optimization problems of approximating a convex polygon P, one by a largest inscribed histogon in P, and the other by a smallest circumscribing histogon.

A *histogon* is an axis-aligned rectilinear polygon such that every horizontal edge has an integer length. We call a histogon of width 1 a *unit histogon* and histogon of *width k* a *k-histogon.* Thus, a unit histogon is simply an axis-aligned rectangle of horizontal width 1, and its height is the length of the vertical sides which is a positive real number. A k-histogon H for a positive integer k can be described by k interior-disjoint unit histogons whose union is H. See Figure 1 for an illustration.

In the inscribed histogon problem, we compute a histogon with maximum area that can be inscribed in P.

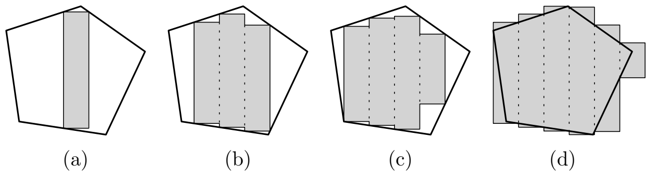

Figure 1: Histogons. (a) The largest inscribed unit histogon of P. (b) The largest inscribed histogon with width 3 of P. (c) The largest inscribed histogon of P. (d) The smallest circumscribed histogon of P

We call such a histogon a *largest* inscribed histogon of P. Depending on whether the horizontal width of a histogon is predetermined (1 or a positive integer k) or not, we consider three versions of the problem.

In the circumscribed histogon problem, we compute a histogon with minimum area that can be circumscribed to P. We call such a histogon a *smallest* circumscribed histogon of P.

We call a copy of a histogon rotated by $\theta \in [0, \pi)$ in counterclockwise direction a *histogon of orientation* θ. Our results can be applied to inscribed and circumscribed problems for histogons of orientation θ with the same time and space.

Approximation of shapes by histogons found its applications in several topics in calculus, most notably in Riemann sums and optimization. For a function graph (or a curve), the area under the graph can be approximated by a histogon: an inscribed histogon is an under-approximation of the area, called a *lower sum,* and a circumscribed histogon is an over-approximation of the area, called an *upper sum.* Many optimization problems are concerned with the largest inscribed figure and the smallest circumscribed figure of a shape. They are also closely related to real-world cost-optimization problems such as painting a piece using a spray gun, etching VLSI masks by electron beams with a fixed minimum width, and inspection.

Related Work. Extensive research has been done in past decades in computational geometry for inscribing and circumscribing polygons, and most of which handle relatively elementary shapes such as triangles, rectangles or parallelograms in a convex or a simple polygon. Alt et al. [3] gave an $O(\log n)$-time algorithm for finding a maximum-area axis-aligned rectangle that can be inscribed in a convex n-gon. Daniels et al. [7] gave an

[*]Work by H.-K. Ahn and J. Chung was supported by the Institute of Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (No. 2017-0-00905, Software Star Lab (Optimal Data Structure and Algorithmic Applications in Dynamic Geometric Environment)) and (No. 2019-0-01906, Artificial Intelligence Graduate School Program(POSTECH)). Work by S. D. Yoon was supported by "Cooperative Research Program for Agriculture Science & Technology Development (Project No. PJ015269032022)" Rural Development Administration, Republic of Korea.

[†]Department of Computer Science and Engineering, Pohang University of Science and Technology, Pohang, Korea. sk7755@postech.ac.kr

[‡]Division of Computer Science and Engineering, Kyonggi University, Suwon, Korea. swbae@kgu.ac.kr

[§]Division of Computer Engineering, Hankuk University of Foreign Studies, Seoul, Korea. cssin@hufs.ac.kr

[¶]Department of Service and Design Engineering, SungShin Women's University, Seoul, Korea. sangduk.yoon@sungshin.ac.kr

[‖]Graduate School of Artificial Intelligence, Department of Computer Science and Engineering, Pohang University of Science and Technology, Pohang, Korea. heekap@postech.ac.kr

$O(n \log^2 n)$-time algorithm for finding a maximum-area axis-aligned rectangle in a simple polygon with n vertices, possibly with holes. The running time was improved to $O(n \log n)$ by Boland et al. [4].

DePano et al. [8] gave an $O(n^2)$-time algorithm for finding a maximum-area equilateral triangle and square that can be inscribed in a convex n-gon and an $O(n^3)$-time algorithm for finding a maximum-area equilateral triangle that can be inscribed in a simple polygon with n vertices. Cabello et al. [5] first suggested $O(n^3)$-time exact algorithm for finding maximum-area or maximum-perimeter rectangles in a convex n-gon. Jin et al. [12] designed an $O(n^2)$-time algorithm for computing all the parallelograms with maximum area in a convex n-gon. Choi et al. [6] gave an $O(n^3 \log n)$-time algorithm for finding maximum-area rectangles in a simple polygon, possibly with holes. Lee et al. [14] studied maximum-area triangles with various restrictions in a convex or a simple polygon, possibly with holes.

Using the observation due to Freeman and Sharpia [10], Toussaint [16] gave an $O(n)$-time algorithm for finding a minimum-area rectangle enclosing a convex n-gon. The algorithm also works for finding a minimum-perimeter rectangle enclosing a convex polygon. O'Rourke et al. [15] gave an $O(n)$-time algorithm for finding a minimum-area triangle enclosing a convex n-gon.

Our Results. Our main results are efficient algorithms for computing optimal histogons (largest inscribed and smallest circumscribed histogons) for a convex polygon P with n vertices and all our algorithms use $O(n)$ space. We assume that the vertices of P are stored in an array in counterclockwise order along the boundary of P.

For the problem of inscribing a largest histogon in a convex n-gon, we present an $O(\log n)$-time algorithm for a largest unit histogon, an $O(\min\{n, k \log^2 \frac{n}{k}\})$-time algorithm for a largest histogon of width k for a fixed $k > 1$, and an $O(\min\{n, w \log^2 \frac{n}{w}\})$-time algorithm for a largest histogon. The symbol w denotes the width of a largest inscribed histogon in P, so the last algorithm is output-sensitive.

For the problem of circumscribing a smallest histogon of a fixed orientation for a convex n-gon, we present an $O(\min\{n, W \log \frac{n}{W}\})$-time algorithm. The symbol W denotes the (horizontal) width of P, so our algorithm is output-sensitive.

Sketch of Our Results. For the problem of inscribing a largest unit histogon, we define a function f that maps $t \in \mathbb{R}$ into the height of the largest inscribed unit histogon of P with the left side at $x = t$. We show that f is a concave, piecewise linear function, so we can perform a binary search to find a maximum of f, which corresponds to a largest unit histogon inscribed in P.

To find a largest k-histogon inscribed in P with $k > 1$, we present a characterization for the existence of k-histogon inscribed in P. For a k-histogon with the leftmost vertical side at $x = t$, we define a function $F(t)$ by the height of the k-histogon and show that F is a concave, piecewise linear function. We find a closed interval containing the $x = t^*$ which maximizes F and apply binary search to find t^* in the restricted domain. We present two algorithms for finding t^*, one using $O(n)$ time which is optimal for $k = \Omega(n)$ (Section 3.2.1) and the other using $O(k \log^2 \frac{n}{k})$ time for $k = O(n)$ (Section 3.2.2). When there is no restriction on the width of the histogon, we show that a largest inscribed histogon can be computed by invoking the algorithm for fixed width a constant number of times. (Section 3.3).

For the problem of circumscribing a smallest histogon, we show that the smallest circumscribed histogon H has width $\lceil W \rceil$. Moreover, either the leftmost vertical side of H contacts the leftmost vertex or the rightmost vertical side of H contacts the rightmost vertex of P. Thus, we compute histogons for two cases, and take the smaller one. See Section 4.

2 Preliminaries

Let P be a convex polygon with n vertices, stored in an array in counterclockwise order along the boundary of P. We denote by ∂P the boundary of P. For a point $p \in \mathbb{R}^2$, let $x(p)$ and $y(p)$ be the x-coordinate and the y-coordinate of p, respectively.

For a histogon H, let $w(H)$ be the horizontal width of H and let $|H|$ denote the area of H. We call a line segment connecting two distinct boundary points of P a *chord* of P.

3 Inscribed histogons

We compute a largest inscribed histogon in a convex polygon P with n vertices for three versions of the problem: a unit histogon, a histogon of width k for a given integer k, and a histogon of any integer width.

3.1 Largest inscribed unit histogon

For ease of discussion, we assume that no two edges of P are parallel to each other. The case with parallel edges can be handled with little modification. Observe that not every convex polygon contains a unit histogon. The following lemma shows the condition for P to contain a unit histogon of a positive height.

Lemma 1 *The longest horizontal chord in P has length larger than 1 if and only if there is a unit histogon of a positive height contained in P.*

Proof. Assume that the longest horizontal chord in P has length larger than 1. Let s be a horizontal unit segment contained in the interior of the longest chord. Then s can be translated vertically upward or downward while it is contained in P. Let s' be such a translated copy of s. Since P is convex, the convex hull of s and s' is a unit histogon of a positive height contained in P.

Assume that there is a unit histogon $\bar{H}$ of a positive height contained in P. Let s be a horizontal unit segment contained in $\bar{H}$, other than its top and bottom sides. Since no two edges of P are parallel to each other, one endpoint of s is in the interior of P. By extending s until both endpoints of s meet ∂P, we get a horizontal chord of length larger than 1 in P. □

By Lemma 1, we can determine the existence of a unit histogon contained in P from the length of the longest horizontal chord in P. Since P is convex and the vertices of P are stored in an array in counterclockwise order along ∂P, we apply binary search to find the longest horizontal chord in P in $O(\log n)$ time.

From now on, we assume that the length of the longest horizontal chord in P is larger than 1. Let $\bar{P}$ be the translate of P by vector $(-1, 0)$. Let $Q = P \cap \bar{P}$, which is a convex polygon. Then there is one-to-one correspondence between any vertical chord at $x = t$ of Q and the largest unit histogon with left side at $x = t$ inscribed in P. Moreover, the length of a vertical chord and the height of its corresponding histogon are the same. Thus, the height of any largest inscribed unit histogon of P is the length of a longest vertical chord in Q. See Figure 2(a).

Note that ∂P and $\partial\bar{P}$ intersect each other at most twice. If there is a horizontal edge of length larger than 1 in P, one intersection may appear as a horizontal line segment on the horizontal edge. Then each intersection corresponds to the horizontal chord of unit length in P or a horizontal edge of length larger than 1 of P. We can compute the intersections $\partial P \cap \partial\bar{P}$ in $O(\log n)$ time by binary search using the sorted array of vertices of P. The longest vertical chord in Q, and the horizontal chords of unit length of P, can be computed in $O(\log n)$ time by applying binary search on the boundary of Q using the sorted array of vertices of P.

To sum up, we can determine whether a unit histogon of a positive height inscribed in P exists in $O(\log n)$ time, and if so, we can compute the largest inscribed unit histogon in the same time.

Theorem 2 *Given a convex polygon P with n vertices stored in an array in order along its boundary, we can find in $O(\log n)$ time the largest unit histogon inscribed in P.*

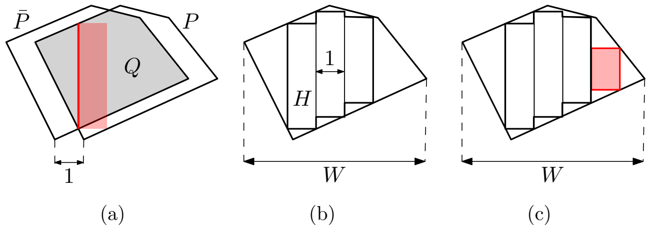

Figure 2: (a) For the translate $\bar{P}$ of P by vector $(-1, 0)$, $P \cap \bar{P}$ is also a convex polygon. (b) $\bar{w}$ is 5.2 and $w(H) = \lfloor\bar{w}\rfloor - 2 = 3$. (c) The largest histogon has width larger than $\lfloor\bar{w}\rfloor - 2$.

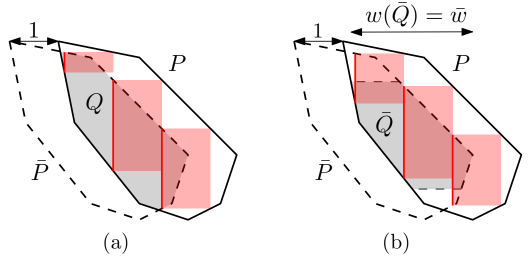

Figure 3: (a) Three unit histogons whose union is not a 3-histogon. (b) Every unit histogon intersects $\bar{Q}$, and every two consecutive unit histogons share a portion along their vertical sides. Their union is a 3-histogon.

3.2 Largest inscribed histogon of a fixed width

Given a positive integer $k > 1$, we compute a largest inscribed histogon H of P with $w(H) = k$. For $t \in \mathbb{R}$, let $H(t)$ denote the largest inscribed k-histogon in P with the leftmost vertical side at $x = t$, and let $\bar{H}(t)$ denote the largest inscribed unit histogon of P with the left side at $x = t$. Then $H(t)$ can be determined by a disjoint union of k unit histogons $\bar{H}(t), \bar{H}(t+1), \ldots, \bar{H}(t+k-1)$.

It is possible that there is no k-histogon that can be inscribed in P even if there are k interior-disjoint unit histogons inscribed in P. Figure 3(a) shows an example with three unit histogons whose union is not a 3-histogon.

Thus, to guarantee a k-histogon inscribed in P, we need the following lemma. Let $\bar{Q}$ be the union of $\ell \cap Q$ over all horizontal lines ℓ with $|\ell \cap Q| \geq 1$. Since Q is a convex polygon, $\bar{Q}$ is also a convex polygon.

Lemma 3 *Assume that $w(\bar{Q}) \neq k - 1$. Then, $w(\bar{Q}) > k - 1$ if and only if there is a k-histogon that can be contained in P.*

Proof. Assume that $w(\bar{Q}) > k - 1$. By letting $t = x(v) + \epsilon$ for the leftmost vertex v of $\bar{Q}$ and sufficiently small $\epsilon > 0$ (smaller than $w(\bar{Q}) - k + 1$), $|\ell_i \cap \bar{Q}| > 0$ for each vertical line $\ell_i : x = t + i$ and $|\bar{H}(t + i)| > 0$ for $i = 0, 1, \ldots, k-1$. Observe that the union of $\bar{H}(t+i)$ for $i = 0, 1, \ldots, k-1$ is a k-histogon $H(t)$ if and only if every two consecutive unit histogons $\bar{H}(t+i)$ and $\bar{H}(t+i+1)$

share a portion (a point or a vertical segment) along their vertical sides at $x = t+i+1$ for $i = 0, 1, \ldots, k-2$. Two consecutive unit histogons $\bar{H}(t+i)$ and $\bar{H}(t+i+1)$ share a portion along their vertical sides if and only if there is a horizontal line ℓ that intersects both $\bar{H}(t+i)$ and $\bar{H}(t+i+1)$.

Suppose that there is no horizontal line that intersects both $\bar{H}(t+i)$ and $\bar{H}(t+i+1)$. Then, we can find a horizontal line ℓ' such that $\ell_i \cap \bar{Q}$ and $\ell_{i+1} \cap \bar{Q}$ are on the opposite sides of ℓ', and the length $|\ell' \cap \bar{Q}|$ must be smaller than 1. This contradicts to the definition of $\bar{Q}$. Therefore, the union of $\bar{H}(t+i)$ for $i = 0, 1, \ldots, k-1$ is a k-histogon contained in P. See Figure 3(b) for an illustration.

Let H be a k-histogon that is inscribed in P. Then H can be partitioned into k interior-disjoint unit histogons $\bar{H}(t+i)$ of positive heights, each corresponding to a vertical line $\ell_i : x = t+i$ for $i = 0, 1, \ldots, k-1$, where t is the x-coordinate of the leftmost side of H. Since every two consecutive unit histogons $\bar{H}(t+i)$ and $\bar{H}(t+i+1)$ share a portion (a point or a vertical segment) along their vertical sides at $x = i+1$, there is a horizontal line ℓ that intersects both $\bar{H}(t+i)$ and $\bar{H}(t+i+1)$. Then $|\ell \cap Q| \geq 1$ which means both ℓ_i and ℓ_{i+1} intersects $\bar{Q}$. Thus $w(\bar{Q}) > k-1$. □

By Lemma 3, we can check whether a k-histogon exists in P by computing $w(\bar{Q})$ using binary search. First, we compute $Q = P \cap \bar{P}$ as we do in Section 3.1 in $O(\log n)$ time. Since the vertices of Q are stored in an array in order, $\bar{Q}$ can be computed in $O(\log n)$ time by locating the two horizontal chords of unit length in Q by binary search on the array.

By applying binary search on the array of $\bar{Q}$, we can compute $w(\bar{Q})$ in $O(\log n)$ time, and decide whether a k-histogon can be inscribed in P or not by Lemma 3 if $w(\bar{Q}) > k-1$. If $w(\bar{Q}) = k-1$, we have to check the existence of $\bar{H}(x(v)+i)$ in P for $i = 0, 1, \ldots, k-1$, where v is the leftmost vertex of $\bar{Q}$. From the convexity of P, the existence of $\bar{H}(x(v)+i)$ in P for $i = 0, 1, \ldots, k-1$ can be confirmed from the existence of two unit histogons $\bar{H}(x(v))$ and $\bar{H}(x(v)+k-1)$ in P, which can be done in $O(\log n)$ time by binary search on the array of Q.

Let I be the set of x-coordinates of all points in $\bar{Q}$, so I is equivalent to the projection of $\bar{Q}$ onto the x-axis. If a k-histogon $H(t)$ can be inscribed in P, there are k unit histogons $\bar{H}(t), \bar{H}(t+1), \ldots, \bar{H}(t+k-1)$ inscribed in P such that any two consecutive unit histogons share a portion along their vertical sides. This implies that $t, t+1, \ldots, t+k-1$ must be contained in the interval I.

We define a function $f \colon \mathbb{R} \to \mathbb{R}$ such that $f(t) = |\bar{H}(t)|$ for any $t \in I$ and $f(t) = -\infty$ for any $t \notin I$. Observe that f is a concave function consisting $O(n)$ linear pieces.

Let $F(x) = \sum_{0 \leq i < k} f(x+i)$. If there exists a k-histogon $H(x)$ inscribed in P, $F(x)$ is the area of $H(x)$. Otherwise, $F(x)$ is $-\infty$. Observe that F is also a concave, piecewise linear function with $O(kn)$ complexity. Our goal is to maximize the function $F(x)$ over $x \in \mathbb{R}$. Let $x^* \in \mathbb{R}$ be a value at which F attains the maximum. If there are more than one such value, we choose the least one as x^*.

Let F'_- be the left-hand derivative of F. There exists a real value $\hat{x} \in \mathbb{R}$ such that $F'_-(\hat{x}) > 0$ and $F'_-(\hat{x}+1) \leq 0$, since F is a concave function. If we restrict the domain of F to $[\hat{x}, \hat{x}+1)$, the function consists of $O(n)$ pieces and we find x^* on it.

We present two algorithms for finding x^*, one using $O(n)$ time which is optimal for $k = \Omega(n)$ (Section 3.2.1) and one using $O(k \log^2 \frac{n}{k})$ time for $k = O(n)$ (Section 3.2.2).

3.2.1 An $O(n)$-time algorithm

We present an $O(n)$-time algorithm for finding x^*, which is optimal for $k = \Omega(n)$. Recall that we can get the interval I in $O(\log n)$ time. We compute the function f that maps $t \in I$ to $|\bar{H}(t)|$ and $t \notin I$ to $-\infty$ by traversing Q in $O(n)$ time. Assume that f consists of $m+2$ linear pieces, where $m = O(n)$. $h_0, h_1, \ldots, h_m, h_{m+1}$ denote the linear functions of these pieces in the order of their domain. Let h'_i denote the derivative of h_i and $(a_i, a_{i+1}]$ denote the domain of h_i for $i = 0, 1, \ldots, m+1$. From the construction of f, $a_0 = -\infty$, $a_{m+2} = +\infty$, $h'_0 = +\infty$, and $h'_{m+1} = -\infty$.

Lemma 4 *We can find $\hat{x}$ such that $F'_-(\hat{x}) > 0$ and $F'_-(\hat{x}+1) \leq 0$ in $O(n)$ time using $O(n)$ space.*

Proof. Let t_i be the smallest integer in $(a_i, a_{i+1}]$ and let N_i be the number of integers in $(a_i, a_{i+1}]$ for $i = 1, 2, \ldots, m$. Then $F'_-(t_i) = \sum_{i \leq j < r_i} (h'_j \cdot N_j) + h'_{r_i} \cdot (k - \sum_{i \leq j < r_i} N_j)$, where r_i is the largest integer such that $\sum_{i \leq j < r_i} N_j < k$. Note that $r_i \leq r_{i+1}$ for each integer i. Then we compute $\sum_{i \leq j < r_i} (h'_j \cdot N_j)$ for all integers i in $O(n)$ time in total. Thus we compute $F'_-(t_i)$ for all integers i in $O(n)$ time. We find an index L such that $F'_-(t_L) > 0$ and $F'_-(t_{L+1}) \leq 0$ in $O(n)$ time. Similarly, we find an index R such that $F'_-(t_R - k) > 0$ and $F'_-(t_{R+1} - k) \leq 0$.

Then there exists an integer s_L with $0 \leq s_L < N_L$ such that $F'_-(t_L + s_L) > 0$ and $F'_-(t_L + s_L + 1) \leq 0$, and there exists an integer s_R with $0 \leq s_R < N_R$ such that $F'_-(t_R - k + s_R) > 0$ and $F'_-(t_R - k + s_R + 1) \leq 0$. This means that $\hat{x} = t_L + s_L = t_R - k + s_R$.

Observe that $F'_-(t_L + a) = \sum_{L \leq j < R} (h'_j \cdot N_j) - h'_L \cdot a + h'_R \cdot b$, where $b = a + k - (t_R - t_L)$. Since the first term $\sum_{L \leq j < R} (h'_j \cdot N_j)$ remains the same for varying a, we can find s_L satisfying $F'_-(t_L + s_L) > 0$ and $F'_-(t_L + s_L + 1) < 0$ in $O(n)$ time. Then $s_R = s_L + k - (t_R - t_L)$. Thus we compute $\hat{x}$ such that $F'_-(\hat{x}) > 0$ and $F'_-(\hat{x}+1) \leq 0$ in $O(n)$ time. □

Then $x^* = \max\{x \in [\hat{x}, \hat{x}+1) \mid F'_-(x) > 0\}$. For each integer i with $1 \le i \le m$, let $k_i(x)$ be a function $k_i : [0,1) \mapsto \mathbb{Z}$ that maps $x \in [0,1)$ to the number of integers j with $0 \le j < k$ satisfying $\hat{x}+x+j \in (a_i, a_{i+1}]$. Note that k_i is a step function having at most three steps in domain $[0,1)$. Let $g_i(x) = k_i(x) \cdot h'_i$ be a function $g_i : [0,1) \to \mathbb{R}$ for each integer i with $1 \le i \le m$. Then g_i is also a step function with at most three steps in domain $[0,1)$. For given an integer i, functions k_i, h'_i, and g_i can be computed in $O(1)$ time. In a step function, each step has an interval as its domain and endpoints of the interval are called *breakpoints* of the step function. Then $\sum_{1\le i \le m} g_i(x) = F'_-(\hat{x}+x)$ and x^* is a breakpoint of g_i's. For a breakpoint s of g_i's, we can compute $F'_-(\hat{x}+s)$ in $O(m) = O(n)$ time by computing $g_i(s)$ for each i and taking the sum of them. Since F'_- is decreasing, we can find x^* in the set of breakpoints by using the median of the set.

Lemma 5 *We can find $x^* \in [\hat{x}, \hat{x}+1)$ in $O(n)$ time using $O(n)$ space.*

Proof. We can construct all g_i functions in $O(n)$ time since each g_i can be computed in $O(1)$ time. Let X be the set of breakpoints in all g_i's. Then $|X| = O(n)$

We find x^* in X iteratively by using the medians of X. The number of breakpoints of X halves over each iteration, and thus the total time spent for computing the median s and $F'_-(\hat{x}+s)$ is $O(n)$. The median s of X can be computed by a selection algorithm that takes time linear to the cardinality of X using Hoare's selection algorithm [11]. Note that $F'_-(\hat{x}+x) = \sum_{1\le i\le m} g_i(x)$ and g_j remains constant in the rest of iterations if X contains no breakpoint of g_j. Let G be the sum of g_j's values such that X contains no breakpoint of g_j. In each iteration, we compute the sum of $g_i(s)$ if X contains a breakpoint of g_i, and compute $F'_-(\hat{x}+s)$ from the sum and G. Then we update X by removing those breakpoints larger than s if $F'_-(\hat{x}+s) \le 0$, and removing those breakpoints smaller than s if $F'_-(\hat{x}+s) > 0$. Finally, we update G. This can be done in time linear to the number of breakpoints in X. We repeat this until X consists of at most two breakpoints.

Observe that F'_- has a positive value at one of the breakpoints. We return the breakpoint as x^*. Since the size of X halves over each iteration, the total time spent over all iterations is $O(n)$. Therefore, x^* can be found in $O(n)$ time using $O(n)$ space. □

Combining Lemma 4 and Lemma 5, we have the following theorem.

Theorem 6 *Given a convex polygon P with n vertices given in order along its boundary and an integer $k > 1$, we can find the largest inscribed k-histogon H in P in $O(n)$ time using $O(n)$ space.*

3.2.2 An $O(k \log^2 \frac{n}{k})$-time algorithm

We present another algorithm for finding x^* in $O(k \log^2 \frac{n}{k})$ time for $k = O(n)$. From now on, we assume that $n \ge 4k$. If $n < 4k$, we apply the algorithm in Section 3.2.1 taking $O(k)$ time. We partition $Q = P \cap \bar{P}$ into two parts along the line ℓ through the leftmost vertex and the rightmost vertex of Q. Let Q^+ denote the upper part and let Q^- denote the lower part of it. Observe that any vertical chord of Q can be partitioned into two pieces by ℓ, one vertical chord of Q^+ and one vertical chord of Q^-.

We group the edges of Q^+ into blocks $B_1, B_2, \ldots, B_m$ of size $\lfloor \frac{n}{k} \rfloor$ consecutively in order from left to right. Similarly, we group the edges of Q^- to blocks $C_1, C_2, \ldots, C_l$ of size $\lfloor \frac{n}{k} \rfloor$ consecutively in order from left to right. Both m and l are $O(k)$. Every block has size $\lfloor \frac{n}{k} \rfloor$, except that the last blocks, B_m and C_l, may consist of less number of edges. For an edge e of P, we say e *contains* an x-coordinate t if the vertical line at $x = t$ intersects e. We say a block B *contains* an x-coordinate t if B contains an edge e and e contains an x-coordinate t.

Our algorithm works as follows. It first computes $\bar{x}$ that maximizes $f(x)$, and sets $D = f'_-(\bar{x})$. It initializes indices $i = 0$ and $j = 0$. Then it searches $\hat{x}$ linearly from $\bar{x}$ by updating D value $k-1$ times as follows. It increases i by 1 and sets $w = \bar{x}+i$ if $D > 0$, and it increases j by 1 and sets $w = \bar{x}-j$ if $D \le 0$. Then it finds blocks B_s and C_t that contain w, computes $f'_-(w)$ using the edges containing w, and updates $D = D + f'_-(w)$. After $k-1$ iterations, we have $D = F'_-(\bar{x}-j)$. The algorithm returns $\hat{x} = \bar{x}-j$ if $D > 0$, and $\hat{x} = \bar{x}-j-1$ if $D \le 0$.

Lemma 7 *We can find $\hat{x}$ such that $F'_-(\hat{x}) > 0$ and $F'_-(\hat{x}+1) \le 0$ in $O(k \log \frac{n}{k})$ time using $O(n)$ space.*

Proof. First we compute $\bar{x}$ that maximizes $f(x)$ in $O(\log n)$ time using binary search. Since $F(x) = \sum_{0\le i<k} f(x+i)$ and f is a concave, piecewise linear function, we can get a larger k-histogon than $H(x)$ by decreasing x if $\bar{x} < x$ or by increasing x if $x+k-1 < \bar{x}$. Then $\bar{x}-k+1 \le x^* \le \bar{x}$. At the end of iterations, $D = F'_-(\bar{x}-j)$. If $D > 0$, $F'_-(\bar{x}-j) > 0$ and $F'_-(\bar{x}-j+1) \le 0$, that is $\hat{x} = \bar{x}-j$. If $D \le 0$, $F'_-(\bar{x}-j-1) > 0$ and $F'_-(\bar{x}-j) \le 0$, that is $\hat{x} = \bar{x}-j-1$. Note that the indices s and t of blocks B_s and C_t containing $w = \bar{x}+i$ monotonically increase while i increases. The indices s' and t' of blocks $B_{s'}$ and $C_{t'}$ containing $w = \bar{x}-j$ monotonically decrease while j increases. Then the step for finding the blocks takes $O(k)$ time in total, since the number of blocks is $O(k)$. Moreover, we can find two edges containing $w = \bar{x}+i$ or $w = \bar{x}-j$ in the blocks in $O(\log \frac{n}{k})$ time using binary search. Thus the time complexity of the algorithm is $O(k \log \frac{n}{k})$ time using $O(n)$ space. □

We define a function $q_i : [0,1) \mapsto \mathbb{R}$ for each integer i with $0 \le i < k$ by $q_i(x) = f'_-(\hat{x}+i+x)$. Then

$\sum_{0\le i<k} q_i(x) = F'_-(\hat{x}+x)$. Note that q_i is a step function on its domain and the total number of breakpoints of all q_i's is $O(n)$. Let b^* be the largest breakpoint of q_i's such that $\sum_{0\le i<k} q_i(b^*) = F'_-(\hat{x}+b^*) > 0$. Then x^* is $\hat{x}+b^*$ since $x^* = \max\{x \in [\hat{x},\hat{x}+1) \mid F'_-(x) > 0\}$.

Note that each breakpoint is induced by a vertex of Q. Consider the sequence of the breakpoints of q_i induced by the vertices on Q^+ from left to right. Let $b_{i,j}$ denote the j-th breakpoint of q_i in the sequence. Similarly, in the sequence of the breakpoints of q_i induced from the vertices on Q^- from left to right, let $c_{i,j}$ be the j-th breakpoint of q_i in the sequence. Then there are $2k$ sequences, two for each q_i, and there are $O(n)$ breakpoints in total.

Lemma 8 *After $O(k\log\frac{n}{k})$-time preprocessing, we can get $b_{i,j}$ and $c_{i,j}$ in $O(1)$ time for any given indices i and j.*

Proof. We show how to get $b_{i,j}$. We can get $c_{i,j}$ similarly. Let $u_{i,j}$ denote the vertex corresponding to $b_{i,j}$. By the definition of q_i, $x(u_{i,j}) = \hat{x}+i+b_{i,j}$. Thus, for given indices i and j, we can get $b_{i,j}$ in $O(1)$ time if we can get $x(u_{i,j})$ in $O(1)$ time. We group the vertices of Q^+ into blocks of size $\lfloor\frac{n}{k}\rfloor$ consecutively in order from left to right. Let B and B' be the two leftmost blocks containing some $t \in [\hat{x}+i, \hat{x}+i+1)$. Then $u_{i,1}$ is the leftmost vertex on edges of B and B' satisfying $x(u_{i,1}) \in [\hat{x}+i, \hat{x}+i+1)$. We search for B and B' for every i linearly in $O(k)$ time. For each i, we find $u_{i,1}$ using binary search in $O(\log\frac{n}{k})$ time. Thus, we can find $u_{i,1}$ for every q_i in $O(k\log\frac{n}{k})$ time. Then we can get $x(u_{i,j})$ for $j>1$ for each q_i in $O(1)$ time as the vertices of Q are stored in an array in order along its boundary. □

By Lemma 8, we can construct the collection of $2k$ sequences implicitly in $O(k\log\frac{n}{k})$ time such that each breakpoint can be accessed in constant time. Our goal is to find the largest breakpoint b^* in the collection such that $\sum_{0\le i<k} q_i(b^*) > 0$.

Kaplan et al. [13] gave a selection algorithm for a row-sorted matrix A with m rows that computes the k smallest items of A in $O(m+k)$ time. Frederickson et al. [9] also gave an $O(m)$-time algorithm for finding the k-th smallest item of A. We describe an algorithm that finds b^* in the collection of $2k$ sequences in $O(k\log^2\frac{n}{k})$ time. Recall that $n \ge 4k$. We partition each sequence of the collection into blocks of size $\lfloor\frac{n}{4k}\rfloor$. They are partitioned into a number of full blocks, followed possibly by one block of size less than $\lfloor\frac{n}{4k}\rfloor$. Then the number of blocks in the collection is $\Theta(k)$. We set the last element in each block as the representative of the block. We select k smallest representatives among all representatives in $O(k)$ time using the selection algorithm by Kaplan et al. We claim that the k-th smallest representative r is an approximated median of the collection. First, the rank of r in the collection is at least $\frac{n}{8}$, since $\frac{n}{8} \le k\lfloor\frac{n}{4k}\rfloor$ for $n \ge 4k$. Second, the number of blocks containing a breakpoint less than r is at most $3k-1$ in the collection, since r is the k-th smallest representative. Then the rank of r in the collection is less than $n-(3k-1)\lfloor\frac{n}{4k}\rfloor \le n-\frac{n}{2} = \frac{n}{2}$.

We evaluate $F'_-(r) = \sum_{1\le i\le k} q_i(r)$. If $F'_-(r) \le 0$, we shrink the search range of each sequence of the collection to the range of the elements smaller than r. If $F'_-(r) > 0$, we shrink the search range of each sequence of the collection to the range of the elements larger than or equal to r. The number of breakpoints in the collection decreases by a constant factor at each iteration.

To evaluate $\sum_{1\le i\le k} q_i(r)$, we need to locate the position of r in each sequence of the collection, except the sequence where r was selected. For each sequence, we already know the block containing r, and we can find the position of r in the block using binary search in $O(\log\frac{n}{k})$ time. Thus it takes $O(k\log\frac{n}{k})$ time to compute the positions of r in all sequences in total.

After $O(\log\frac{n}{4k})$ iterations, the number of remaining breakpoints in the collection becomes smaller than $4k$. Then we use the algorithm in Section 3.2.1 with all elements in the collection to find b^* taking $O(k)$ time.

Taken together, there are $O(\log\frac{n}{k})$ iterations, each of which takes $O(k\log\frac{n}{k})$ time. Thus it takes $O(k\log^2\frac{n}{k})$ time using $O(n)$ space to find $x^* = \hat{x}+b^*$.

Theorem 9 *Given a convex polygon P with n vertices stored in an array in order along its boundary and an integer $k = O(n)$, we can find the largest inscribed k-histogon H in P in $O(k\log^2\frac{n}{k})$ time using $O(n)$ space.*

3.3 Largest inscribed histogon

Now we consider the variation that no restriction is imposed on the width of a largest inscribed histogon in P. We find the largest inscribed histogon H in P.

Recall that $\bar{Q}$ is the union of $\ell\cap Q$ with $|\ell\cap Q| \ge 1$ for all horizontal lines ℓ. Let $\bar{w} = w(\bar{Q})$. Then the largest inscribed histogon has width at most $\lfloor\bar{w}\rfloor+1$ by Lemma 3.

We now claim that the width of the largest inscribed histogon in P is either $\lfloor\bar{w}\rfloor-1$, $\lfloor\bar{w}\rfloor$ or $\lfloor\bar{w}\rfloor+1$. Suppose that the largest inscribed histogon H in P has width $k \le \lfloor\bar{w}\rfloor-2$. Then there are k vertical lines intersecting $\bar{Q}$ such that all distance between two consecutive lines is 1. Since $\bar{w}-k+1 \ge 3$, the leftmost vertical line is at distance larger than 1 from the leftmost point of $\bar{Q}$ or the rightmost vertical line is at distance larger than 1 from the rightmost point of $\bar{Q}$. Thus, we can always attach a unit histogon with positive height to the left or right of H and get an inscribed histogon with a larger area in P. Thus, the largest histogon has width $\lfloor\bar{w}\rfloor-1$, $\lfloor\bar{w}\rfloor$ or $\lfloor\bar{w}\rfloor+1$. See Figure 2(b–c). Once we compute $\bar{w}$ in $O(\log n)$ time, we can compute the largest histogons (of width $\lfloor \bar{w} \rfloor - 1$, $\lfloor \bar{w} \rfloor$ and $\lfloor \bar{w} \rfloor + 1$) using the algorithms in Section 3.2 and choose the largest one.

In conclusion, we can compute the largest inscribed histogon of P in $O(n)$ time using $O(n)$ space by Theorem 6. For $\bar{w} = O(n)$, we can compute it in $O(\bar{w} \log^2 \frac{n}{\bar{w}})$ time using $O(n)$ space by Theorem 9.

Theorem 10 *Given a convex polygon P with n vertices stored in an array in order along its boundary, we can find the largest inscribed histogon in P in $O(\min\{n, \bar{w} \log^2 \frac{n}{\bar{w}}\})$ time using $O(n)$ space, where W denotes the width of the largest inscribed histogon in P.*

4 Smallest circumscribed histogon

We consider the problem of covering a convex polygon P with n vertices by a histogon with smallest area and present algorithms for computing the smallest circumscribed histogon of P.

We denote by $\bar{H}(t)$ the smallest unit histogon with the left side at $x = t$ that covers the part of P between $x = t$ and $x = t + 1$. Let l_t denote the intersection between P and the vertical line $x = t$. Observe that $\bar{H}(t)$ is defined if l_t or l_{t+1} has a positive length, and $|\bar{H}(t)| = \max\{|l_t|, |l_{t+1}|\}$.

Let H^* denote the smallest histogon covering P, and let x^* be the x-coordinate of the leftmost vertical side of H^*. Then H^* can be represented by the disjoint union of $w(H^*)$ unit histogons, $\bar{H}(x^*), \bar{H}(x^* + 1), \ldots \bar{H}(x^* + w(H^*) - 1)$.

We denote by $\hat{P}$ the Minkowski sum of P and the horizontal segment with endpoints $(-1, 0)$ and $(0, 0)$. Note that the length of the longest vertical segment contained in $\hat{P}$ at $x = t$ is the same as $|\bar{H}(t)|$.

We define a function $g : \mathbb{R} \mapsto \mathbb{R}$ by $g(t) = |\bar{H}(t)|$ if $\bar{H}(t)$ is defined, otherwise $g(t) = -\infty$. Then g is a concave, piecewise linear function, since $\hat{P}$ is convex and $g(t)$ is the length of the longest vertical segment at $x = t$ contained in $\hat{P}$.

Lemma 11 *Let H^* be a smallest histogon covering P. Then, $w(H^*) = \lceil W \rceil$, where W is the (horizontal) width of P. There is a smallest histogon covering P whose leftmost vertical side contacts the leftmost vertex of P or whose rightmost vertical side contacts the rightmost vertex of P.*

Proof. Let x_l and x_r be the x-coordinates of the leftmost vertex and the rightmost vertex of P, respectively. Any smallest histogon H^* covering P with its leftmost vertical side with $x = x^*$ satisfies $x_l - 1 < x^* \leq x_l$ and $x_r \leq x^* + w(H^*) < x_r + 1$. Thus, $w(H^*) \geq \lceil x_r - x_l \rceil = \lceil W \rceil$ and $w(H^*) \leq \lceil W \rceil + 1$. Suppose that $w(H^*) = \lceil W \rceil + 1$. We define a function $G : I \mapsto \mathbb{R}$ by $G(x) = \sum_{0 \leq i < \lceil W \rceil + 1} g(x + i)$, where I is a maximal interval such that $g(x + i) > 0$ for all integers i with $0 \leq i < \lceil W \rceil + 1$. We minimize $G(x)$ subject to $x \leq x_l$ and $x_r \leq x + \lceil W \rceil + 1$, which P can be circumscribed by the union of $\bar{H}(x), \bar{H}(x+1), \ldots \bar{H}(x + \lceil W \rceil)$. If $G(x)$ has a minimum, $G(x)$ is minimized at $x = x_l$ or $x = x_r - \lceil W \rceil - 1$ since G is also concave and piecewise linear as g. Thus $x^* = x_l$ or $x^* = x_r - \lceil W \rceil - 1$ which means H^* touches either the leftmost vertex or the rightmost vertex of P. Then either $\bar{H}(x^*)$ or $\bar{H}(x^* + \lceil W \rceil)$ does not intersect P, a contradiction. Thus, $w(H^*) = \lceil W \rceil$ and H^* touches either the leftmost vertex or the rightmost vertex of P. ☐

By Lemma 11, $w(H^*) = \lceil W \rceil$ and there are only two candidate locations for H^*, one with $x^* = x_l$ and one with $x^* = x_r - \lceil W \rceil$. To compute their areas, we can use the method for computing the area of the largest inscribed histogon in Section 3.3. More precisely, we show how to compute the area of the smallest circumscribed histogon with $x^* = x_l$. We construct function g in $O(n)$ time using the vertices of $\hat{P}$ stored in an array in order along its boundary which can be computed in $O(\log n)$ time. For each piece of g, we find in $O(1)$ time the smallest integer s and the largest integer t such that $x_l + s$ and $x_l + t$ are contained in the domain of the piece. Then we can compute $\sum_{s \leq i \leq t} g(x_l + i)$ in $O(1)$ time. By summing the values over all pieces, $\sum_{0 \leq i < \lceil W \rceil} g(x_l + i)$ can be computed in $O(n)$ time. If $W = O(n)$, similar to Lemma 8, we find two edges containing $x = x_l + i$ for all $0 \leq i < \lceil W \rceil$ in $O(W \log \frac{n}{W})$ time and compute $\sum_{0 \leq i < \lceil W \rceil} g(x_l + i)$ in $O(W)$ time. Among two candidates for H^*, the smaller one is the smallest circumscribed histogon of P.

Theorem 12 *Given a convex polygon P with n vertices stored in an array in order along its boundary, we can find the smallest circumscribed histogon of P in $O(\min\{n, O(W \log \frac{n}{W})\})$ time using $O(n)$ space where W denotes the width of the smallest circumscribed histogon in P.*

5 Discussion

We present algorithms for computing the largest inscribed histogon and the smallest circumscribed histogon for a convex polygon. The histogons are required to be axis-aligned. A direction for future work is to consider a generalization of the problem in which the histogons can be of arbitrary orientations.

References

[1] H.-K. Ahn, S. W. Bae, O. Cheong, and J. Gudmundsson. Aperture-angle and hausdorff-approximation of convex figures. *Discrete & Computational Geometry*, 40:414–429, 2008.

[2] H.-K. Ahn, P. Brass, O. Cheong, H.-S. Na, C.-S. Shin, and A. Vigneron. Inscribing an axially symmetric polygon and other approximation algorithms for planar convex sets. *Computational Geometry*, 33:152–164, 2006.

[3] H. Alt, D. Hsu, and J. Snoeyink. Computing the largest inscribed isothetic rectangle. In *Proc. 7th Canad. Conf. Comput. Geom. (CCCG 1995)*, pages 67–72, 1995.

[4] R. P. Boland and J. Urrutia. Finding the largest axis aligned rectangle in a polygon in $O(n \log n)$ time. In *Proc. 13th Canad. Conf. Comput. Geom. (CCCG 2001)*, pages 41–44, 2001.

[5] S. Cabello, O. Cheong, C. Knauer, and L. Schlipf. Finding largest rectangles in convex polygons. *Computational Geometry*, 51:67–74, 2016.

[6] Y. Choi, S. Lee, and H.-K. Ahn. Maximum-area and maximum-perimeter rectangles in polygons. *Computational Geometry*, 94:101710, 2021.

[7] K. Daniels, V. Milenkovic, and D. Roth. Finding the largest area axis-parallel rectangle in a polygon. *Computational Geometry*, 7(1):125–148, 1997.

[8] A. DePano, Y. Ke, and J. O'Rourke. Finding largest inscribed equilateral triangles and squares. In *Proc. 25th Allerton Conf. Commun. Control Comput.*, pages 869–878, 1987.

[9] G. N. Frederickson and D. B. Johnson. Generalized selection and ranking: Sorted matrices. *SIAM Journal on Computing*, 13(1):14–30, 1984.

[10] H. Freeman and R. Shapira. Determining the minimum-area encasing rectangle for an arbitrary closed curve. *Commun. ACM*, 18(7):409?413, jul 1975.

[11] C. A. R. Hoare. Algorithm 65: Find. *Commun. ACM*, 4(7):321?322, jul 1961.

[12] K. Jin and K. Matulef. Finding the maximum area parallelogram in a convex polygon. In *Proc. 23rd Canad. Conf. Comput. Geom. (CCCG 2011)*, 2011.

[13] H. Kaplan, L. Kozma, O. Zamir, and U. Zwick. Selection from heaps, row-sorted matrices and x+y using soft heaps. In *Proc. 2nd Sympos. Simplicity Algo. (SOSA 2019)*, pages 5:1–5:21, 2019.

[14] S. Lee, T. Eom, and H.-K. Ahn. Largest triangles in a polygon. *Computational Geometry*, 98:101792, 2021.

[15] J. O'Rourke, A. Aggarwal, S. Maddila, and M. Baldwin. An optimal algorithm for finding minimal enclosing triangles. *Journal of Algorithms*, 7(2):258–269, 1986.

[16] G. Toussaint. Solving geometric problems with the rotating calipers. *In Proceedings of IEEE MELECON'83*, 83, 02 2000.