The Largest Contained Quadrilateral and the Smallest Enclosing Parallelogram of a Convex Polygon

Günter Rote

July 4, 2019

Abstract

We present a linear-time algorithm for finding the quadrilateral of largest area contained in a convex polygon, and we show that it is closely related to an old algorithm for the smallest enclosing parallelogram of a convex polygon.

1 Introduction

A linear-time algorithm for the *largest quadrilateral contained in a convex polygon* was proposed in 1979 by Dobkin and Snyder [3]. This algorithm stood until 2017, when Keikha, Löffler, Mohades, Urhausen, and van der Hoog [4] constructed a counterexample for which it fails.

A simple linear-time algorithm for the *smallest parallelogram enclosing a convex polygon* was published in a technical report by Schwarz, Teich, Welzl, and Evans [8] in 1994, see also [7].

We will show that the two problems are closely related, in particular when they are constrained by *anchoring* them to some specified direction. The solution of one problem provides an optimality certificate for the other problem. We present a conceptually simple algorithm that treats both problems in a symmetric way and solves them simultaneously in linear time. The algorithm is based on the "rotating calipers" technique from the early days of computational geometry. Proofs are included, so that there can be no doubts about its correctness.

The algorithm becomes very simple when specialized for solving only one of the two problems, see Appendices B and C. Linear-time algorithms for the largest quadrilateral were independently found in 2018 by Vahideh Keikha (personal communication, manuscript in preparation) and by Kai Jin (personal communication, as part of a manuscript previously submitted to a conference), and they are essentially the same as the algorithm given here. According to [1], a linear-time solution is given in unpublished notes of Michael Shamos from 1974 [9]. Given that the solution is so simple, this is plausible, but I have not been able to confirm it.

While the algorithms that we develop were known, the observation that the two problems are so closely connected (Lemma 2) appears to be new. A similar dual connection between the anchored versions of two problems exists between the largest contained and the smallest enclosing *triangle*. This connection was first noted and exploited in the linear-time algorithm of Chandran and Mount [2] for these problems, see also [6, Lemmas 4.i and 14] for a slightly more stringent treatment in the style of Lemma 2.

Acknowledgements. I thank Kai Jin and in particular Vahideh Keikha for comments on previous versions of this manuscript.

2 Conjugate Pairs

A *direction* is given by a nonzero vector $\mathbf{u} \in \mathbb{R}^2$. Parallel vectors represent the same direction, and opposite directions are considered equal. Directions are conveniently parameterized by the polar angle θ: $\mathbf{u}(\theta) = \binom{\cos\theta}{\sin\theta}$.

We denote the quadrilateral contained in P conventionally by its four corners $ABCD$. For the parallelogram that surrounds P, it will be better to denote it by the four *sides abcd*, leaving the corners anonymous, see Figure 1.

Definition 1. a) A quadrilateral $ABCD$ is *D-anchored* to $\mathbf{u}$ if the diagonal AC is parallel to $\mathbf{u}$.

b) A parallelogram $abcd$ is *S-anchored* to $\mathbf{u}$ if the two sides b and d are parallel to $\mathbf{u}$.

The letter D stands for "diagonal", and S stands for "side". We will sometimes just say "*anchored*" if it is clear from the context which version we mean.

Definition 2. Let $F = ABCD$ be a quadrilateral, and let $G = abcd$ be a parallelogram. We say that F and G are *conjugate*, or (F, G) form a *conjugate pair*, if

1. the diagonal AC is parallel to the sides b and d, and
2. each corner A, B, C, D of F lies on the corresponding side a, b, c, d of G.

Because of the first condition, the two elements F and G of a conjugate pair are anchored to the same direction. Because of the second condition, F is a convex quadrilateral contained in G. It is possible that F degenerates to a triangle because it is not necessarily strictly convex, and it may even happen that some corners coincide.

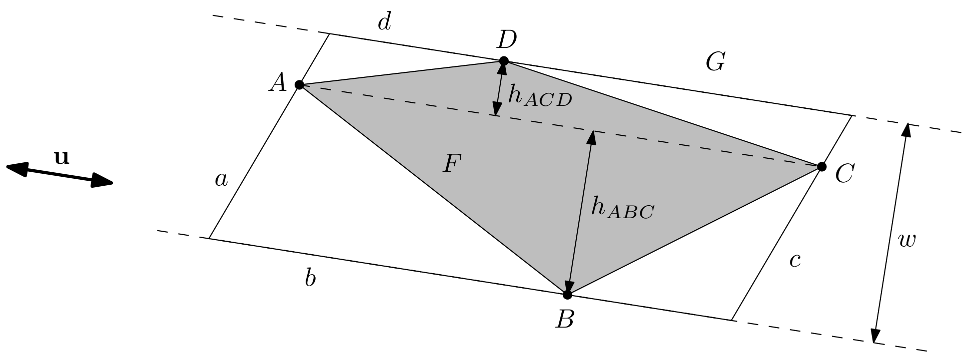

Figure 1: A conjugate pair (F, G). The quadrilateral $F = ABCD$ is D-anchored and the parallelogram $G = abcd$ is S-anchored to the direction $\mathbf{u}$. The heights h_{ACD} and h_{ABC} of the two triangles into which $ABCD$ is decomposed by the diagonal AC sum up to the distance w between the lines through b and d.

The following basic geometric lemma considers a conjugate pair (F, G) in isolation and proves some optimality properties of F and G with respect to each other.

Lemma 1. *a) Let G be a parallelogram, S-anchored to some direction $\mathbf{u}$. Then a quadrilateral F that is contained in G and is D-anchored to $\mathbf{u}$ is a largest quadrilateral with these properties if and only if (F, G) is a conjugate pair.*

b) Let F be a quadrilateral, D-anchored to some direction $\mathbf{u}$. Then a parallelogram G that contains F and is S-anchored to $\mathbf{u}$ is a smallest parallelogram with these properties if and only if (F, G) is a conjugate pair.

c) If (F, G) is a conjugate pair, the area of G is twice the area of F.

Proof. See Figure 1. Since F and G are required to be anchored to the same direction, the first condition for a conjugate pair is always satisfied. The question is whether the four sides of G are incident to the four corresponding corners of F.

Let $|b| = |d|$ denote the length of the two sides of G that are parallel to $\mathbf{u}$. Then, given that the diagonal AC should be parallel to $\mathbf{u}$ and contained in G, it is clear that

$$|AC| \le |b| = |d|,$$

with equality if and only if the sides a and c touch A and C.

Moreover, if F is contained in G, the distance between B and D, when projected to the direction perpendicular to $\mathbf{u}$, is at most the distance w between the lines through b and d:

$$\left|\mathbf{u}^{\perp} \cdot (D - B)\right| \le w,$$

with equality if and only if the sides b and d touch B and D.

(a) The quadrilateral $F = ABCD$ is composed of the triangles ABC and ACD, which share the common base AC. Therefore, the area of F is expressed in terms of the heights h_{ABC} and h_{ACD} of these triangles as

$$\tfrac{1}{2}|AC| \times (h_{ABC} + h_{ACD}) = \tfrac{1}{2}|AC| \times \left|\mathbf{u}^{\perp} \cdot (D - B)\right| \le \tfrac{1}{2}|b|w, \tag{1}$$

and we have just seen that equality holds if and only if the four sides of G touch the corresponding corners of F. This proves (a). The area of the parallelogram G is

$$|b|w = |d|w, \tag{2}$$

which equals twice the area of F in (1), and this proves (c).

To prove (b), we use (1) in the other direction, giving a lower bound on the area (2) of any anchored parallelogram G containing F. Again, since equality in (1) holds if and only if F and G are conjugate, (b) has been proved. $\square$

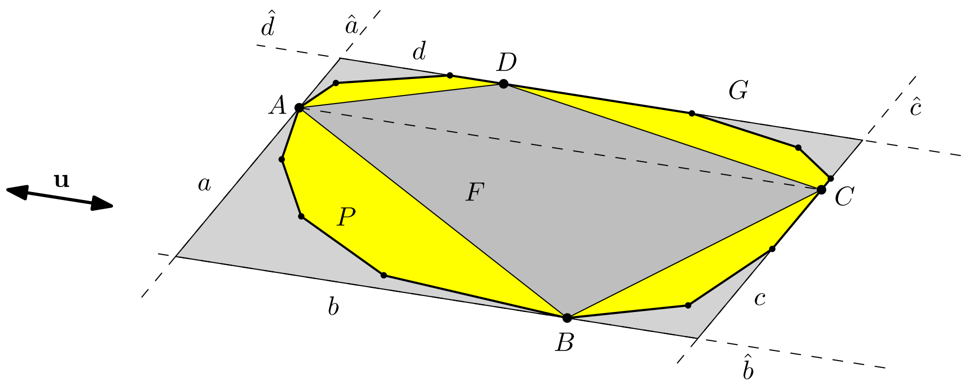

Figure 2: A conjugate pair $(F, G) = (ABCD, abcd)$ anchored to the direction $\mathbf{u}$ and sandwiching a convex polygon P.

The following crucial lemma gives the optimality condition for the anchored versions of the two problems.

Lemma 2 (Characterization of Optimality by Conjugate Pairs)**.** *Let P be a convex polygon in the plane, and let $\mathbf{u}$ be a direction.*

a) A quadrilateral F that is D-anchored to $\mathbf{u}$ and contained in P is a largest quadrilateral with these properties if and only if there is a parallelogram containing P that is conjugate to F.

b) A parallelogram G enclosing P and S-anchored to $\mathbf{u}$ is a smallest parallelogram with these properties if and only if there is a quadrilateral contained in P that is conjugate to G.

Proof of sufficiency. In both cases, there is a conjugate pair (F, G) such that the convex region P is sandwiched between them: $F \subseteq P \subseteq G$.

a) By Lemma 1a, F is even the largest D-anchored quadrilateral inside the larger region $G \supseteq Q$. Thus, there cannot be a larger anchored quadrilateral in P.

b) By Lemma 1b, G is even the smallest S-anchored parallelogram that encloses the smaller region $F \subseteq Q$. Thus, there cannot be a smaller anchored parallelogram enclosing P. ☐

Necessity of the conditionsis not needed for the correctness of our algorithm, and it will only be proved later as an easy consequence of Lemma 4, see page 7. Alternatively, there are easy direct proofs (cf. [8, Lemma 2]), even for arbitrary convex regions.

The lemma is also a manifestation of linear programming duality, since the problem of finding the longest chord AC with a given direction can be formulated as a linear program.

3 Constructing all Conjugate Pairs in Linear Time

The idea is to construct conjugate pairs $(F(\theta), G(\theta))$ with $F(\theta) \subseteq P \subseteq G(\theta)$, for all directions $\mathbf{u}(\theta)$ in the range $0° \le \theta \le 180°$. By the sufficient criterion of Lemma 2, these are largest anchored contained quadrilaterals and smallest anchored enclosing parallelograms. Hence, the overall largest contained quadrilaterals and smallest enclosing parallelograms will be among them.

The following straightforward observation separates the task of finding an anchored conjugate pair (F, G) into two subtasks. The first involves A, C, a, and c, and it is concerned with the direction of the diagonal AC. The other task involves B, D, b, and d, and it is concerned with the direction of the sides b and d. A pair of points on the boundary of a convex region P that admits parallel supporting lines is called *antipodal*.

Lemma 3. *Let P be a convex region in the plane and $\mathbf{u}$ be a direction. A conjugate pair $(ABCD, abcd)$ with $ABCD \subseteq P \subseteq abcd$ and anchored to $\mathbf{u}$ is found as follows, see Figure 2. Here, the parallelogram abcd is defined by two pairs of parallel lines $\hat{a}, \hat{c}$ and $\hat{b}, \hat{d}$:*

a) AC is an antipodal pair of P parallel to $\mathbf{u}$, with supporting lines $\hat{a}$ and $\hat{c}$,

b) $\hat{b}$ and $\hat{d}$ are the two opposite lines of support parallel to $\mathbf{u}$, and B and D are points where these lines touch P. (Thus, BD is also an antipodal pair.) ☐

As stated in the following lemma, whose proof will be given in Section 5, both tasks can easily be carried out with the classical rotating-calipers technique. We assume that P is a convex polygon, given by the ordered list of its n vertices.

Lemma 4. *a) In $O(n)$ time, one can find a sequence of direction angles $0° = \theta_0 < \theta_1 < \cdots < \theta_{i-1} < \theta_i < \cdots < \theta_n < \theta_{n+1} = 180°$, and a corresponding sequence of vertex-edge pairs $(Q_1, e_1), (Q_2, e_2), \ldots, (Q_{n+1}, e_{n+1})$, such that for any θ in each closed interval $[\theta_{i-1} \mathinner{..} \theta_i]$, an antipodal segment $A(\theta)C(\theta)$ parallel to $\mathbf{u}(\theta)$ can be found by intersecting the line through Q_i parallel to $\mathbf{u}(\theta)$ with the edge e_i. The lines parallel to e_i through Q_i and e_i are the corresponding supporting lines.*

b) In $O(n)$ time, one can find a sequence of direction angles $0° = \phi_0 < \phi_1 < \cdots < \phi_{i-1} < \phi_i < \cdots < \phi_k < \phi_{k+1} = 180°$, and a corresponding sequence of antipodal pairs of vertices $(B_1, D_1), (B_2, D_2), \ldots, (B_{k+1}, D_{k+1})$, with $k \le n$, such that for any ϕ in each closed interval $[\phi_{i-1} \mathinner{..} \phi_i]$, the lines through B_i and D_i parallel to the direction $\mathbf{u}(\phi)$ are supporting lines.

We remark that the sequence (B_i, D_i) does not necessarily include *every* pair of antipodal vertices: For each pair of opposite parallel edges of P, there are two pairs of antipodal vertices which admit parallel supporting lines of only one direction. These pairs don't appear in the list.

It is now clear how to proceed with the help of Lemma 4. Since the areas of a conjugate pair are related by Lemma 1c, let us ignore the enclosing parallelograms $a(\theta)b(\theta)c(\theta)d(\theta)$ and concentrate on the inner quadrilaterals $A(\theta)B(\theta)C(\theta)D(\theta)$. We merge the lists of breakpoints $\theta_0, \theta_1, \ldots$ and $\phi_0, \phi_1, \ldots$ and obtain a list of $O(n)$ intervals such that in each interval, there are largest anchored quadrilaterals $A(\theta)B(\theta)C(\theta)D(\theta)$ with a fixed structure: The points $B(\theta) = B$ and $D(\theta) = D$ are fixed vertices. On the diagonal $A(\theta)C(\theta)$, one point, say $A(\theta) = A$, is fixed to a vertex Q_i, while the other point $C(\theta)$ moves on a fixed edge e_i.

In a quadrilateral $ABC(\theta)D$ with one moving point C, the area is a linear function of C. As θ increases, the corner $C(\theta)$ moves monotonically on some edge e_i, and therefore, the extremes are attained at the endpoints of the interval. We thus just need to evaluate the area at all interval endpoints θ_i and ϕ_i of the merged sequence and pick the largest or smallest one. Since each endpoint belongs to two intervals, the quadrilateral $A(\theta)B(\theta)C(\theta)D(\theta)$ prescribed by Lemma 4 may be ambiguous, but this does not matter. All these quadrilaterals have the same area.

Theorem 5. *a) The quadrilateral of largest area contained in a convex polygon can be found in linear time.*

b) The parallelogram of smallest area enclosing a convex polygon can be found in linear time. □

Pseudocode for the algorithm is given in Appendix A, and prototype implementations of the algorithms in Appendices A to C in PYTHON are contained in the source files of this preprint.

4 Discussion

It is perhaps instructive to reflect on some features of this algorithm and compare it to other approaches. An easy property of largest quadrilaterals (in fact, largest k-gons for any k) contained in a polygon P is the *vertex property*: Their corners must be vertices of P. Our algorithm does not use this property at all. It considers an infinite family $A(\theta)B(\theta)C(\theta)D(\theta)$ of quadrilaterals. Even after reducing them to a discrete set of directions (the interval endpoints θ_i and ϕ_i), many of these candidates don't fulfill the vertex property. Most previous algorithms for largest contained k-gons, and in particular, the algorithms of Dobkin and Snyder [3], consider only k-gons with the vertex property. By concentrating on the vertex property too early, one may miss useful avenues to finding good and simple algorithms.

We may of course still use the vertex property as an "afterthought" to introduce shortcuts and simplify the algorithm. For example, once the point $C(\theta)$ lies in the middle of an edge, one can skip the area computations and fast-forward θ until $C(\theta)$ arrives at a vertex. (For the problem of the largest contained triangle, the analogous step is described in [6, Section 8].)

There are other possible simplifications. The two lists of breakpoints $\theta_0, \theta_1, \ldots$ and $\phi_0, \phi_1, \ldots$ need not be computed separately in advance. They can be generated on the fly as they are processed, after an appropriate initialization. We have described the algorithm in terms of angles θ for convenience. When implementing the algorithm on a computer, it is better to avoid angle calculations and use direct comparisons of vector directions or signed areas, see Appendix A.1. (Anyway, since the problem is invariant under affine transformations, angular quantities are not really suited to the problem.)

In Appendix B, we show the whole simplified algorithm for the largest contained quadrilateral. This algorithm is actually so simple that one can as well derive it directly from the property that AC must form an antipodal vertex pair, without going through the continuous family $A(\theta)B(\theta)C(\theta)D(\theta)$. The same remark holds for the smallest enclosing parallelogram. Appendix C shows a variation of the algorithm following [8] that is just as simple.

5 Rotating Calipers

Proof of Lemma 4. For part (a), we need antipodal points for all directions. An algorithm for listing all antipodal pairs of *vertices* of a convex polygon P is given in [5, Section 4.2.3]. We just need to "fill the gaps" in order to get antipodal pairs for a continuous range of directions

Let f and g be two opposite lines of support in direction $\mathbf{u}(\phi)$, see Figure 3. We will increase ϕ from $\phi = 0°$ to $\phi = 180°$ and maintain the points A and C where they touch P. Since we want these points to move continuously, we parameterize the process by a new parameter $t = \phi + s$, where s it the combined distance moved by $A(t)$ and $C(t)$ along the boundary of P since the beginning. We start with $A(0)$ and $C(0)$ as the lowest and highest points of P. In case of ties, we take the leftmost lowest and the rightmost highest point. Figure 3a shows ϕ and the distances s_A and s_C moved by A and C, from which s is computed as $s = s_A + s_C$.

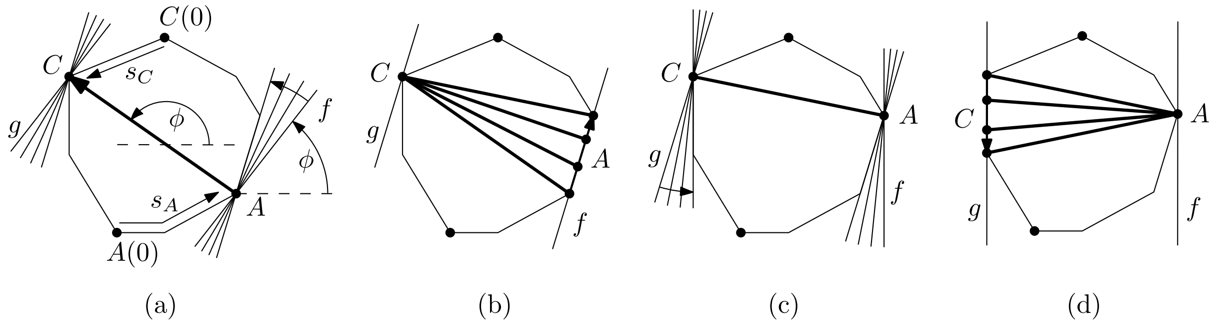

Figure 3: Four successive stages of the circular sweep: (a) The antipodal points $A = A(t)$ and $C = C(t)$ together with the parallel support lines $f = f(t)$ and $g = g(t)$, for the parameter $t = \phi + s_A + s_C$. The angle ϕ increases until f or g hits an edge. (b) The line f has hit an edge. C is stationary and A slides along this edge. (c) ϕ increases further, and g hits an edge. (d) A is stationary and C moves.

Now we start to increase t. Whenever $\mathbf{u}(\phi)$ is parallel to an edge of P, we continuously advance $A(t)$ or $C(t)$ to the other endpoint of this edge, increasing s while leaving ϕ constant. If P has two sides parallel to $\mathbf{u}(\phi)$, we arbitrarily use the convention that we first advance $A(t)$ and then $C(t)$. Now, $f(t)$ and $g(t)$ are ready to tilt around the vertices $A(t)$ and $C(t)$, increasing ϕ while s remains constant, until $f(t)$ or $g(t)$ hits the next edge.

We continue this process in a loop until $\phi = 180°$. At this point, A and C have swapped places, and s equals the perimeter of P. The segment $A(t)C(t)$ has completed a rotation by 180°.

The points $A(t)$ and $C(t)$ move continuously in counterclockwise direction as a function of t, and for every t, the points $A(t)$ and $C(t)$ are antipodal, as witnessed by supporting lines $f(t)$ and $g(t)$. Thus we have achieved our primary goal of finding an antipodal pair for every direction.

The parameter range of t is decomposed into intervals where A remains stationary, C remains stationary, or both points remain stationary. We cut out those intervals where none of the points move. For the remaining intervals, we choose yet another parameterization, namely by the direction $\mathbf{u}(\theta)$ pointing from $A(t)$ to $C(t)$.

Each of the remaining intervals is characterized by one stationary point, Q_i, while the other point moves on a fixed edge, e_i. If $\mathbf{u}(\theta)$ is the direction pointing from $A(t)$ to $C(t)$, The breakpoints θ_{i-1} and θ_i are the directions at the end of the intervals, when both $A(t)$ and $C(t)$ are at vertices. It only remains to rearrange the interval breakpoints cyclically modulo 180° in order to start with $\theta_0 = 0°$. Since each interval advances either A or C by one vertex and A and C together make a full tour around P, the number of interval breakpoints θ_i is n.

Part (b) of the lemma is straightforward. In fact, it can be obtained by the same circular sweep as above, with the straightforward parameterization by the angle ϕ, concentrating only on the points $A(\phi)$ and $C(\phi)$ where the supporting lines in direction $\mathbf{u}(\phi)$ touch P. (These points will take the roles of B_i and D_i in the lemma.)

The breakpoint directions ϕ_i are therefore the directions where $A(\phi)$ or $C(\phi)$ jumps. These are the directions for which $\mathbf{u}(\phi_i)$ is parallel to some edge of P. There are at most n such angles. The sequence $\phi_1, \phi_2, \ldots$ is obtained by merging the two lists of edge directions obtained from traversing the left boundary of P and the right boundary of P counterclockwise, between the extreme points in vertical direction. $\square$

Proof of necessity in Lemma 2. a) Assume that F' is a largest quadrilateral that is D-anchored to $\mathbf{u}$ and contained in P. Lemma 4 together with Lemma 3 implies that, for this direction $\mathbf{u}$, there exists an anchored conjugate pair (F, G) with $F \subseteq P \subseteq G$. By the sufficiency part of Lemma 2, which has already been proved, F is a largest anchored quadrilateral contained in P, and therefore of the same area as F'. By Lemma 1a, F is even a largest anchored quadrilateral contained in the larger area G. By the necessity statement in the same lemma, since F' is also contained in G, F' can only have the same area as F if it forms a conjugate pair (F', G) with G. This proves the necessity for Part (a). The proof of Part (b) is completely analogous. $\square$

References

[1] James E. Boyce, David P. Dobkin, Robert L. (Scot) Drysdale III, and Leo J. Guibas. Finding extremal polygons. *SIAM J. Computing*, 14(1):134–147, 1985. doi:10.1137/0214011.

[2] Sharat Chandran and David M. Mount. A parallel algorithm for enclosed and enclosing triangles. *International Journal of Computational Geometry & Applications*, 2(2):191–214, 1992. doi:10.1142/S0218195992000123.

[3] David P. Dobkin and Lawrence Snyder. On a general method for maximizing and minimizing among certain geometric problems. In *20th Annual Symposium on Foundations of Computer Science*, pages 9–17. IEEE, 1979. doi:10.1109/SFCS.1979.28.

[4] Vahideh Keikha, Maarten Löffler, Ali Mohades, Jérôme Urhausen, and Ivor van der Hoog. Maximum-area triangle in a convex polygon, revisited. Preprint, 2017. arXiv:1705.11035v2.

[5] Franco P. Preparata and Michael Ian Shamos. *Computational Geometry. An Introduction.* Springer, 1985.

[6] Günter Rote. The largest inscribed triangle and the smallest circumscribed triangle of a convex polygon: An overview of linear-time algorithms. Class notes, June 2019. URL: https://kvv.imp.fu-berlin.de/x/w77mPE.

[7] Christian Schwarz, Jürgen Teich, Alek Vainshtein, Emo Welzl, and Brian L. Evans. Minimal enclosing parallelogram with application. In *Proceedings of the Eleventh Annual Symposium on Computational Geometry*, SoCG'95, pages 434–435, New York, NY, USA, 1995. ACM. doi:10.1145/220279.220338.

[8] Christian Schwarz, Jürgen Teich, Emo Welzl, and Brian L. Evans. On finding a minimal enclosing parallelogram. Technical Report TR-94-036, International Computer Science Institute (ICSI), August 1994. URL: http://www.icsi.berkeley.edu/ftp/global/pub/techreports/1994/tr-94-036.pdf.

[9] Michael Ian Shamos. Problems in computational geometry. Manuscript, 1974–1977.

A The Algorithm in Pseudocode

For completeness, we give the pseudocode for our algorithm. We assume that the convex polygon $P = (p_1, p_2, \ldots, p_n)$ is given by the ordered list of its n vertices in counterclockwise order. We assume that $n \geq 3$, and we look for a largest contained quadrilateral $ABCD = p_a p_b p_c p_d$ in counterclockwise order, and a smallest enclosing parallelogram, also in counterclockwise order. Indices of polygon vertices are considered modulo n.

In contrast to the algorithm that is sketched in Section 3, we don't start with $\theta_0 = 0°$, but we start more conveniently with the antipodal pair defined by $A = p_1$ and the point C opposite to the edge $p_1 p_2$.

A.1 Primitive Operations

The basic predicate of this algorithm is a comparison between two directions $\mathbf{u} = \binom{x_1}{y_1}$ and $\mathbf{v} = \binom{x_2}{y_2}$, which can be calculated with only two multiplications as the sign of a 2×2 determinant that expresses the signed area of the parallelogram spanned by $\mathbf{u}$ and $\mathbf{v}$:

$$\det(\mathbf{u}, \mathbf{v}) := \begin{vmatrix} x_1 & x_2 \\ y_1 & y_2 \end{vmatrix} = x_1 y_2 - x_2 y_1 = -\det(\mathbf{v}, \mathbf{u})$$

This is positive if $\mathbf{v}$ lies counterclockwise from $\mathbf{u}$. The area of a quadrilateral $ABCD$ is $\pm\frac{1}{2}\det\big((C - A), (D - B)\big)$.

Frequently, the algorithm makes comparisons between triangle areas over a common basis. This should also be calculated as a 2×2 determinant. For example, $\operatorname{area} p_a p_{a+1} p_{c+1} - \operatorname{area} p_a p_{a+1} p_c = \frac{1}{2}\det\big((p_{a+1} - p_a), (p_{c+1} - p_c)\big)$ if the two triangles are oriented counterclockwise. Since we find the formulation involving triangle areas geometrically more appealing, we have not replaced it in our pseudocode.

A.2 Largest and smallest anchored quadrilaterals

Lemma 6 ([8, Lemma 1]). *There is a smallest enclosing parallelogram abcd such that*

1. *at least one of the sides a and c touches an edge of P, and*
2. *at least one of the sides b and d touches an edge of P.*

Proof. A smallest enclosing parallelogram $abcd$ must be a smallest enclosing parallelogram *anchored* to the direction of b and d, and hence there must be a conjugate pair $(ABCD, abcd)$, see Figure 2. If the side a or c doesn't already touch an edge of P, these sides can be tilted around A and C without changing the area, until one of the sides hits an edge of P.

Afterwards, we can apply the same argument to the direction of a and c and ensure that b or d touches an edge of P. □

As a consequence of part 2, when looking for the smallest enclosing parallelogram, it is sufficient to look at parallelograms that are S-anchored to the directions of the edges of P. We have already mentioned that a largest contained quadrilateral can be found among those quadrilaterals that use only vertices of P. Thus it is sufficient to look at anchored quadrilaterals for which A and C lie at vertices. This explains the places where areas are compared against the current minimum or maximum in the following program.

A.3 Pseudocode

```
$a_0 := a := 1$
$c := 2$
**while** area $p_a p_{a+1} p_{c+1} >$ area $p_a p_{a+1} p_c$:
    $c_0 := c := c + 1$ (find the point $p_c$ with supporting line parallel to $p_a p_{a+1}$.)
$next\_AC :=$ "A" (The corner $A$ slides on the edge $p_a p_{a+1}$.)
$\mathbf{u}^{AC} := p_c - p_{a+1}$ (the direction $\mathbf{u}$ where $A$ hits the next vertex)
$b := a$
**while** area $p_c p_a p_{b+1} >$ area $p_c p_a p_b$:
    $b := b + 1$ (find the point $p_b$ with supporting line parallel to $p_a p_c$.)
$d := c$
**while** area $p_a p_c p_{d+1} >$ area $p_a p_c p_d$:
    $d := d + 1$ (find the other point $p_d$ with supporting line parallel to $p_a p_c$.)
**if** area $p_b p_{b+1} p_{d+1} \leq$ area $p_b p_{b+1} p_d$:
    $next\_BD :=$ "B" (The parallelogram side $b$ hits an edge of $P$ before $d$ does.)
    $\mathbf{u}^{BD} := p_{b+1} - p_b$
**else:**
    $next\_BD :=$ "D" (The parallelogram side $d$ hits an edge of $P$ before $b$ does.)
    $\mathbf{u}^{BD} := p_d - p_{d+1}$
$maxarea := 0$ (the area of the largest contained quadrilateral)
$minarea := \infty$ (the area of the smallest enclosing parallelogram)
**repeat**
    **if** $\det(\mathbf{u}^{BD}, \mathbf{u}^{AC}) \geq 0$:
        (The parallelogram side $b$ or $d$ touches an edge of $P$.)
        **if** $next\_AC =$ "A":
            construct the point $A$ on the line $p_a p_{a+1}$ such that $p_c A$ is parallel to $\mathbf{u}^{BD}$
(*)         $minarea := \min\{minarea, 2 \cdot \text{area}\, A p_b p_c p_d\}$
        **else**:
            construct the point $C$ on the line $p_c p_{c+1}$ such that $p_a C$ is parallel to $\mathbf{u}^{BD}$
(**)        $minarea := \min\{minarea, 2 \cdot \text{area}\, p_a p_b C p_d\}$
        **if** $next\_BD =$ "B":  $b := b + 1$
        **else**:                $d := d + 1$
        **if** area $p_b p_{b+1} p_{d+1} \leq$ area $p_b p_{b+1} p_d$:
            $next\_BD :=$ "B" (The parallelogram side $b$ hits an edge of $P$ before $d$ does.)
            $\mathbf{u}^{BD} := p_{b+1} - p_b$
        **else:**
            $next\_BD :=$ "D" (The parallelogram side $d$ hits an edge of $P$ before $b$ does.)
            $\mathbf{u}^{BD} := p_d - p_{d+1}$
    **else**: (The sliding corner $A$ or $C$ reaches a vertex of $P$.)
        **if** $next\_AC =$ "A":  $a := a + 1$
        **else**:                $c := c + 1$
        $maxarea := \max\{maxarea, \text{area}\, p_a p_b p_c p_d\}$
        **if** area $p_a p_{a+1} p_{c+1} \leq$ area $p_a p_{a+1} p_c$: (Which of $A$ and $C$ slides on an edge of $P$?)
            $next\_AC :=$ "A" (The corner $A$ slides on the edge $p_a p_{a+1}$.)
            $\mathbf{u}^{AC} := p_c - p_{a+1}$ (the direction $\mathbf{u}$ where $A$ hits the next vertex)
        **else:**
            $next\_AC :=$ "C" (The corner $C$ slides on the edge $p_c p_{c+1}$.)
            $\mathbf{u}^{AC} := p_{c+1} - p_a$ (the direction $\mathbf{u}$ where $C$ hits the next vertex)
**until** $(a, c) = (c_0, a_0)$
```

The area of $Ap_bp_cp_d$ in line $(*)$ can be computed by the formula

$$\pm\frac{1}{2}\cdot\frac{\det\bigl(p_{a+1}-p_a, p_c-p_a\bigr)\cdot\det\bigl(\mathbf{u}^{\mathrm{BD}}, p_d-p_b\bigr)}{\det\bigl(p_{a+1}-p_a, \mathbf{u}^{\mathrm{BD}}\bigr)},$$

and for the area of $p_ap_bCp_d$ in $(**)$, we replace $p_{a+1}-p_a$ by $p_{c+1}-p_c$ in two places.

B The Largest Contained 4-Gon

We give here the specialized algorithm for computing the area of the largest 4-gon contained in a convex polygon P.

In contrast to the algorithm that is sketched in Section 3, and also differently from Appendix A.3, we start as in Section 5 with the points $A(0)$ and $C(0)$ that have horizontal supporting lines, see Figure 3.

> Let p_{a_0} be the leftmost vertex among the lowest vertices of P
> Let p_{c_0} be the rightmost vertex among the highest vertices of P
> $a := b := a_0$
> $c := d := c_0$
> $maxarea := 0$
> **repeat**
>   **while** area $p_cp_ap_{b+1} >$ area $p_cp_ap_b$:
> $b := b+1$ (find the point p_b with supporting line parallel to p_ap_c.)
>   **while** area $p_ap_cp_{d+1} >$ area $p_ap_cp_d$:
> $d := d+1$ (find the other point p_d with supporting line parallel to p_ap_c.)
> $maxarea := \max\{maxarea, \text{area}\, p_ap_bp_cp_d\}$
>   **if** area $p_ap_{a+1}p_{c+1} \le$ area $p_ap_{a+1}p_c$: (advance (a,c) to the next antipodal pair)
> $a := a+1$
>   **else**:
> $c := c+1$
> **until** $(a,c) = (c_0, a_0)$

The main loop is driven by the antipodal pair (a,c). In each iteration, either a or c is advanced to the next vertex. This is essentially the program for reporting all antipodal pairs of vertices from [5, Section 4.2.3], except that we need not be careful about getting all such pairs if P has parallel edges. In the two inner loops, the points b and d that are farthest from the line ac are updated.

C The Smallest Enclosing Parallelogram According to Schwarz, Teich, Welzl, and Evans [8]

The algorithm of Schwarz et al. [8] is similar in spirit to our algorithm in constructing a sequence of parallelograms $abcd$ by advancing the direction to which b and d are parallel, following the rotating-calipers technique. They also sketch an application of smallest enclosing parallelograms to signal compression [8, Section 4], and the appendix gives details about a C++ implementation.

There is one difference in the setup. We explain it with our notation: By Lemma 6 ([8, Lemma 1]), it suffices to look for parallelograms where at least one of the sides b and d touches a whole edge of P, *and* at least one of the sides a and c touches a whole edge of P. This means that two adjacent parallelogram sides must touch edges of P. Now, the algorithm of [8] only considers those anchored parallelograms where these two sides are b and c, like in Figure 4a. This restriction is compensated by sweeping over an angular range of 360° instead of 180°.

Figure 4 illustrates a few steps of the algorithm. After finding the parallelogram of Figure 4a and computing its area, the algorithm of Section A.3, when specialized for the smallest containing parallelogram, would next look at the parallelogram of Figure 4b. This parallelogram is skipped in Schwarz et al. [8] at this point, but this omission is no mistake: This parallelogram was already considered before with rotated labels, when b touched p_2p_3 and c touched p_6p_7. The next parallelogram is not shown: d touches the edge $p_{12}p_{13}$ and a touches p_2p_3. This parallelogram is also skipped by Schwarz et al. [8] at this point, but it is considered later when b touches $p_{12}p_{13}$ and c touches p_2p_3. Figure 4b shows the next parallelogram. It is a largest S-anchored parallelogram when the side b is anchored, but it is not a largest S-anchored parallelogram when the side a is anchored, because the dashed antipodal pair p_7p_{13} is not parallel to a and c. Hence it cannot be a largest enclosing parallelogram. The algorithm of Schwarz et al. [8] skips this parallelogram and does not consider it at all.

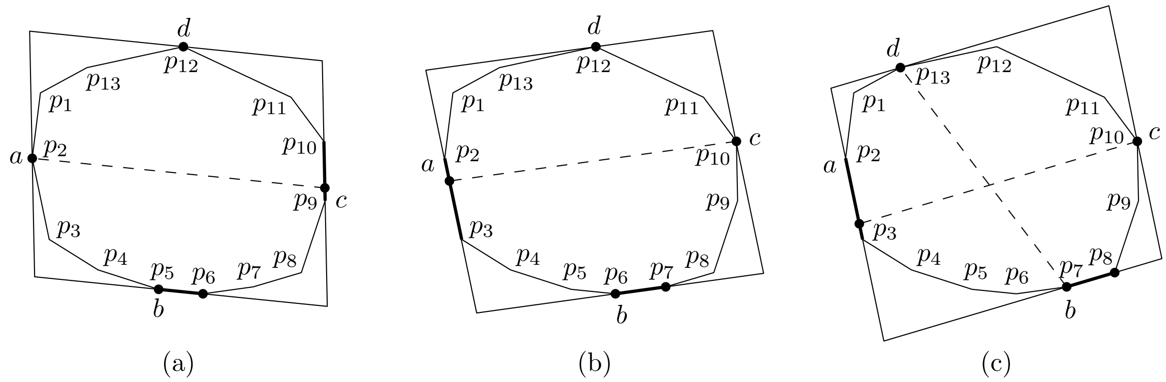

Figure 4: Three snapshots of the algorithm

This setup makes the algorithm simple and elegant: Most case distinctions of Section A.3 disappear, and the flags *next_AC* and *next_BD* can be eliminated. Like in Section B, the algorithm is structured into two nested loops. The outer loop iterates over the edges of P through which b goes, and the inner loop updates the antipodal pair AC parallel to the direction of b.

$b := 1$; $c := 2$; $d := 2$; $a := 3$
while area $p_cp_{c+1}p_{a+1} >$ area $p_cp_{c+1}p_a$: (initialization)
  $a := a + 1$ (find the opposite point p_a with supporting line parallel to p_cp_{c+1})
$minarea := \infty$
for $b := 1 \,..\, n$:
  while area $p_bp_{b+1}p_{d+1} >$ area $p_bp_{b+1}p_d$:
    $d := d + 1$ (update the point p_d opposite to p_bp_{b+1})
  while area $p_bp_{b+1}p_a >$ area $p_bp_{b+1}p_{c+1}$:
    $c := c + 1$ (search for antipodal pair AC parallel to $\mathbf{u} = p_bp_{b+1}$, with C on an edge)
    while area $p_cp_{c+1}p_{a+1} >$ area $p_cp_{c+1}p_a$
      or (area $p_cp_{c+1}p_{a+1} =$ area $p_cp_{c+1}p_a$ **and** area $p_bp_{b+1}p_{a+1} \geq$ area $p_bp_{b+1}p_c$):
        $a := a + 1$ (update the point p_a opposite to p_cp_{c+1})
  if area $p_bp_{b+1}p_a \geq$ area $p_bp_{b+1}p_c$:
    construct the point C on the line p_cp_{c+1} such that p_aC is parallel to $\mathbf{u} = p_bp_{b+1}$
    $minarea := \min\{minarea, 2 \cdot \text{area}\, p_ap_bCp_d\}$

The algorithm of [8] actually uses a precomputed list $L = ((p_i, p_{i+1}), p_{q_i})_{i=1..n}$ that stores for each edge (p_i, p_{i+1}) of P an antipodal vertex p_{q_i} that is farthest away from the line through (p_i, p_{i+1}). By contrast, the algorithm above updates the vertex p_d opposite to p_bp_{b+1} and the vertex p_a opposite to p_cp_{c+1} on the fly. The treatment of degenerate cases is also different from [8].