88-26

ALGORITHMS FOR COMPUTING THE CENTER OF AREA OF A CONVEX POLYGON

MATTHEW DIAZ

JOSEPH O'ROURKE

TECH REPORT 88-26

Algorithms for Computing the Center of Area of a Convex Polygon

Matthew Díaz* Joseph O'Rourke†

October 17, 1988

Abstract

Given a convex polygon P, associate with each point $p \in P$ the minimum area of the polygon to the left of any chord through p. The maximum over all points in P is known as "Winternitz's Measure of Symmetry" and the point p^* that achieves this maximum we call the center of area. We will show that p^* is unique and derive geometric properties of minimum-area chords. These properties lead to two algorithms for computing the center of area of a convex polygon with n vertices. The first is a combinatorial algorithm that runs in time $O(n^6 \log^2 n)$. The second is a numerical algorithm that computes the coordinates to K bits of precision in time $O(nK)$. We conclude with a discussion of our implementation of the second algorithm, extensions to higher dimensions and other generalizations.

1 Introduction

Given a set of points, Q, in the plane, one may wish to find a point (not necessarily in Q), such that regardless of how you pass a line through this point, the ratio of the number of points on either side is not very large or very small. This "balance" point is called a *centerpoint*, and algorithms for computing it and the related k-hulls are known [Ede87], [CSY87].

Moving from the discrete to the continuous domain, we ask the following: Given a convex polygon P, does there exist a point such that regardless of how the polygon is cut through that point the ratio of the areas of the resulting pieces is "balanced"? The center of mass can be considered "balanced"

* Dept. of Computer Science, The Johns Hopkins University, Baltimore, MD 21218
† Dept. of Computer Science, Smith College, Northampton, MA 01060

since any chord passing through it cuts off $\geq 4/9$ of the area (Winternitz's Theorem [YB61]). However, the center of mass is not always the most "balanced" point: the point that maximizes the minimum area cut off by any chord through it we call the *center of area.*

We first introduce the notation that will be used throughout this paper, and then discuss certain geometric properties of the center of area and related geometric structures. The most important property to be established is that the center of area of a convex polygon is a unique point. This is followed by two algorithms for computing the coordinates of the center of area – one combinatorial and the other numerical. The combinatorial algorithm has high complexity, $O(n^6 \log^2 n)$, but the numerical algorithm only requires $O(n)$ work per bit of accuracy in the location of the center of area. We conclude with implementation details and open problems.

2 Preliminaries

We will be considering only objects in the plane. Define $L(p,\theta)$ to be a directed line through the point p, and forming an angle θ with the positive x axis, and let $L^+(p,\theta)$ be the closed halfplane bounded by, and to the left of, $L(p,\theta)$ (Figure 1a).

For a convex polygon P, define $C(p,\theta)$ to be the chord formed by the intersection of P with $L(p,\theta)$. Define the function $\alpha(p,\theta) = |L^+(p,\theta) \cap P|$, the area of the region to the left of the chord $C(p,\theta)$ (Figure 1b).

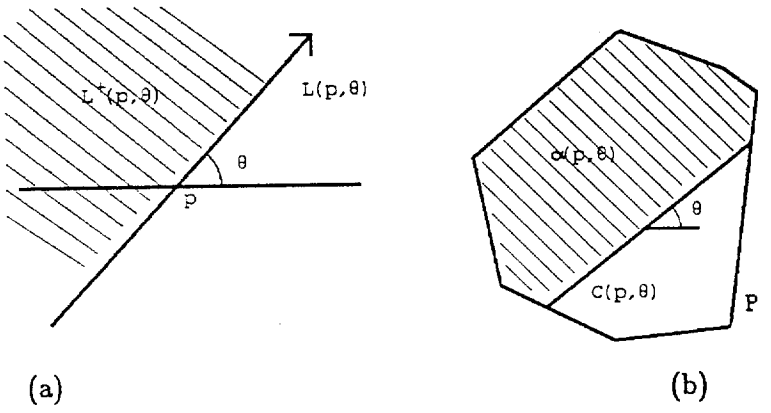

Figure 1: Notation

Define

$$\alpha(p) = \min_{\theta} \alpha(p, \theta).$$

$\alpha(p)$ is thus the minimum area cut off by *any* chord through p. Let θ^* be one of the angles that minimizes $\alpha(p, \theta)$. In general it is possible for there to be an arbitrary number of angles that minimize $\alpha(p)$ (for example, any chord through the center of a regular polygon will be minimal). Unless stated otherwise, we will use θ^* to represent any one of the possible choices. We will refer to $L(p, \theta^*)$ and $C(p, \theta^*)$ as a min-line and a min-chord, respectively. We can now formally define the center of area as the set of all points p^* that satisfy

$$\alpha(p^*) = \max_{p} \alpha(p).$$

Also let $\delta^* = \alpha(p^*)$. The area ratio of the pieces of P determined by $C(p^*, \theta^*)$ is known as *Winternitz's Measure of Symmetry* [Grü63].

The center of area and related concepts have been studied by geometers for some time, and the mathematical properties we derive below have either appeared in the literature, or can be derived from equivalent results.[1] Nevertheless we feel it is useful to have a unified presentation, both for the algorithms to follow and for generalizations to non-convex polygons.

2.1 Properties

$\alpha(p)$ is a function defined over the convex polygon P. The contours (level surfaces) of this function, are defined as:

$$\Gamma_\delta \equiv \{p : \alpha(p) = \delta\}$$

The following properties of Γ_δ are easily verified:

1. Γ_δ is a simple, connected curve.
2. If $\delta_1 < \delta_2$ then Γ_{δ_2} is strictly contained in Γ_{δ_1}.
3. If $p \in \Gamma_\delta$ then each $L(p, \theta^*)$ supports Γ_δ, $\Gamma_\delta \subset P - L^+(p, \theta^*)$.
4. Γ_δ is convex. [SS67]

Figure 2 shows the contours for a non-regular polygon, taken at 5% intervals. The contours Γ_δ can be considered "δ-hulls", the continuous analog of k-hulls [Ede87]. It can be shown algebraically that the contours are comprised of pieces of hyperbolic curves determined by the edges of P [Hog62], [Día89].

[1] We thank Branko Grünbaum for enlightening us [Grü88].

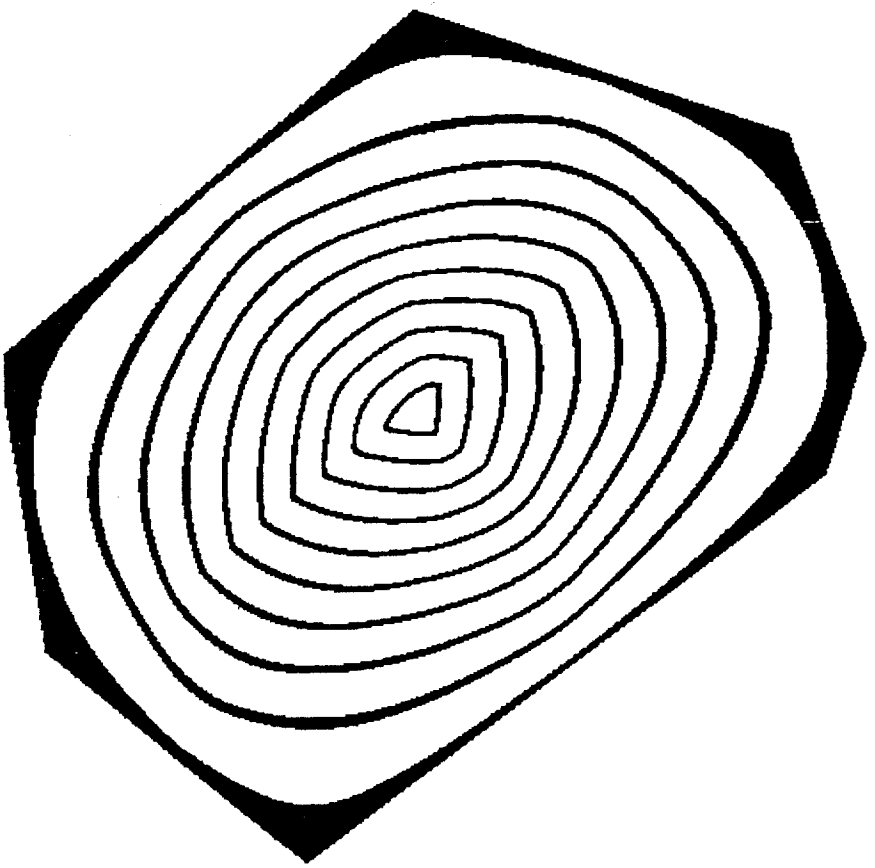

Figure 2: The contours for a convex polygon P.

The contours are shown at 5 percent intervals, each representing a range of $[\delta, \delta + 1)$ percent due to integer truncation. Thus the outer contour extends from $[0, 1)$ percent and the innermost contour extends over the range $[45, 46)$ percent. The 0 percent contour is the boundary of P. The maximum possible value for δ^* is 50 percent, and must always be greater than 4/9 (44) percent by Winternitz's Theorem. This figure was generated by dividing P into "pixels" and calculating $\alpha(p)$ for each pixel p by spinning a line about p and searching for the minimum area cut off.

If $C(p,\theta)$ has endpoints a and b on the boundary of P, and if the length of the line segment ap is the same as the length of bp, $\mid ap \mid = \mid bp \mid$, then we say that $C(p,\theta)$ is a *balanced chord.* The following lemma has been known for convex sets as far back as 1889 [Bru89], and has appeared more recently in [CY86]. We present our own proof in order to facilitate extension to non-convex polygons [DO].

Lemma 2.1 *The chord $C(p,\theta)$ is a min-chord only if it is balanced.*

Proof Assume that $C(p,\theta)$ is not balanced, and let a and b be the points at which $C(p,\theta)$ intersects the boundary of polygon P. We examine the rotation of the line $L(p,\theta)$ both clockwise and counterclockwise by a small angle ϵ with the restriction that the chord, $C(p,\theta)$, associated with $L(p,\theta)$ does not become balanced at any point during the rotation. (If it did become balanced it would contradict our assumption.) In particular, consider the chord $C(p,\theta-\epsilon)$ that intersects the boundary of P at points a_- and b_- and $C(p,\theta+\epsilon)$ that intersects at a_+ and b_+. (Figure 3).

We know that $C(p,\theta)$ is not balanced, so without loss of generality let $\mid ap \mid < \mid bp \mid$. It must also be the case then that $\mid pa_- \mid < \mid pb_- \mid$ and $\mid pa_+ \mid < \mid pb_- \mid$ for if they were not, then the chord would have passed through a balanced state.

The area of $\triangle apa_-$ is strictly less than the area of $\triangle bpb_-$ since they share the common angle ϵ and the former has two sides of length strictly less than that of the latter. Similarly, the area of $\triangle apa_+$ is strictly less than the area of $\triangle bpb_+$. This implies $\alpha(p,\theta-\epsilon) > \alpha(p,\theta)$ and $\alpha(p,\theta) > \alpha(p,\theta+\epsilon)$, which implies there is always a direction that $L(p,\theta)$ can be rotated to decrease $\alpha(p,\theta)$. Thus $C(p,\theta)$ cannot be a min-chord. □

The converse of this is not necessarily true: not every balanced chord determines a min-line.

Using Lemma 2.1 we can now prove the following uniqueness result, which had been proved previously in [Süs50] by a different method.

Lemma 2.2 *For a convex polygon the center of area is a unique point.*

Proof From the properties of the contours we know that the center of area must be a convex figure. If we assume that it is not a point, then it must be either a line segment or region. We will discuss these in turn.

Assume that the center of area is a line segment, ab, and thus

$$\alpha(p) = \delta^* \quad \forall\, p \in ab$$

We distinguish two cases:

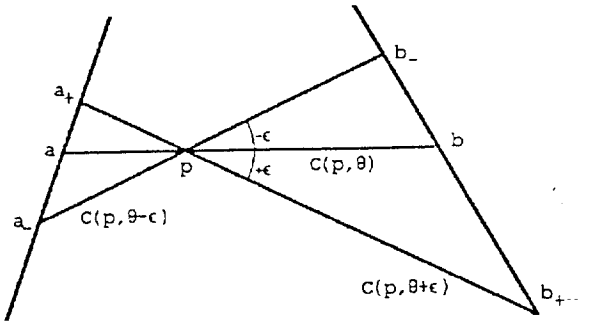

Figure 3: Rotating $L(p,\theta)$ for Lemma 2.1

Case 1. $\exists\ p,\ p \in ab$, such that $L(p,\theta^*)$ is not collinear with ab. Take a point, $p' \in ab$, such that p' is to the left of p. (When p is the left endpoint case 2 holds). Since P is a connected region, it must be the case that

$$\alpha(p',\theta^*) < \alpha(p,\theta^*).$$

But

$$\alpha(p,\theta^*) = \alpha(p) = \alpha(p') = \delta^*.$$

Hence

$$\alpha(p',\theta^*) < \alpha(p'),$$

which contradicts the definition of $\alpha(p')$ as a minimum.

Case 2. It must be the case that there exist two distinct points, p and p', $p, p' \in ab$ such that both $L(p,\theta^*)$ and $L(p',\theta^*)$ are collinear with ab. Then both $L(p,\theta^*)$ and $L(p',\theta^*)$ intersect the boundary of P at the same points. By Lemma 2.1 it must be that both p and p' are midpoints of the same chord, contradicting the assumption that p and p' are distinct.

The situation where the center of area is a region is handled by case 1 since it is always possible to find a point p' to the left of $L(p,\theta^*)$ through which to construct $L(p',\theta^*)$.

The center of area is a connected region, and having thus disallowed the possibilty of the center of area being a line or region, it must be the case that it is a single point. □

This lemma remains true for non-convex polygons [DO] but it is false for subsets of the plane that are not simply-connected (Figure 4).

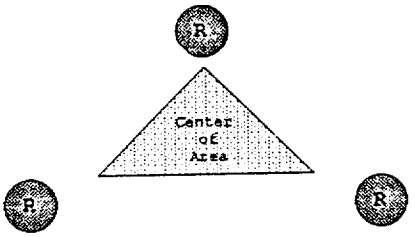

Figure 4: Center of area for a disconnected region R.

The minimum area of R which is cut off by any point in the center of area is $1/3$, which is best possible.

3 A Combinatorial Algorithm

The first algorithm to be described is combinatorial in nature. Intuitively it partitions the polygon into regions where $\alpha(p)$ is simple and then solves a minimization problem over each piece. We call this partition the *chord diagram* of P. This diagram contains much information in addition to the center of area data, but finding the center of area exactly without constructing the diagram would require sophisticated pruning. It should be noted that the goal of this algorithm is to indicate that the solution could be found in polynomial time, and as such the bounds used are brute force estimates.

3.1 Chord Diagram

Through every point in a convex polygon, P, there exists at least one balanced chord [Grü72]. While any given point may be the midpoint of several balanced chords, no two of the balanced chords through any given point may intersect the same pair of edges. We know from Lemma 2.1 that one or more of these balanced chords will be min-chords. If we label the edges of P as $e_1 ... e_n$, we can assign to each min-chord a *label pair*, (e_i, e_j), corresponding to the labels of the edges it intersects.[2] Then for each point, p, in the polygon, we can associate with it the set of label pairs, $\{(e_{i_1}, e_{j_1}), ..., (e_{i_k}, e_{j_k})\}$ representing edges intersected by min-chords through p. We then partition

[2]If we define e_i to include its left endpoint but not its right one, then every chord will be assigned a unique label pair.

P into equivalence classes such that p_1 and p_2 are in the same class iff their label sets are equivalent. This partitioning we call the *chord diagram.* Partitions with label sets of size 1 (one label pair) correspond to regions of P, sets of size 2 represent boundaries between these regions, and sets of size 3 or more, vertices. In particular the center of area will be one of these vertices.

We construct the chord diagram in two stages. First we identify the regions in P where there *might* be a point p such that $C(p, \theta^*)$ intersects a specific pair of edges. Second, we compute the pointwise minimum of the functions over each of these regions to find $\alpha(p)$. We begin by examining the set of balanced chord midpoints for a particular pair of edges. Let $R_{i,j}$ be the set of points such that for any $p \in R_{i,j}$, p is the midpoint of a balanced chord through edges e_i and e_j. $R_{i,j}$ is the set of points through which a min-chord might pass (by Lemma 2.1) and touch e_i and e_j.

Lemma 3.1 *$R_{i,j}$ is a parallelogram with sides parallel to e_i and e_j, and half as long.*

Proof Let a_i and b_i be the left and right endpoints of e_i, respectively, and likewise for a_j and b_j for edge e_j. We wish to enumerate all possible chords with endpoints on e_i and e_j and examine the locus of midpoints. We start with the set of chords with one endpoint at a_j and the other on e_i. These chords sweep through the triangle $a_i b_i a_j$ and the midpoints of these chords sweep out a line parallel to the base of the triangle, e_i, with length $\mid e_i/2 \mid$. (Figure 5a). We next move the endpoint at a_j towards b_j by a distance ϵ, and again sweep the second endpoint from a_i to b_i. The midpoints of these chords follow another line segment parallel to e_i and shifted towards b_i by $\epsilon/2$ in a direction parallel to e_i. Continuing in a like manner we generate the series of line segments shown in Figure 5b. As we let ϵ shrink to 0, the locus of midpoints converges to a solid parallelogram. □

Each pair of edges gives rise to a parallelogram with sides parallel to those edges, half their length and with vertices located at the midpoints of the segments connecting the endpoints of the edges. The set of parallelograms generated by all $O(n^2)$ pairs of edges of a polygon P is shown in Figure 6. Note that this same diagram can be constructed by placing a half-size copy of P at each vertex of P.

Over each parallelogram $R_{i,j}$ one can define the function $\alpha_{i,j}(p)$ to be the minimum area cut off by a chord with endpoints on e_i and e_j with midpoint p, $p \in R_{i,j}$. This function has the form:

$$c_\theta y^2 + 2xy + c_d y, \tag{1}$$

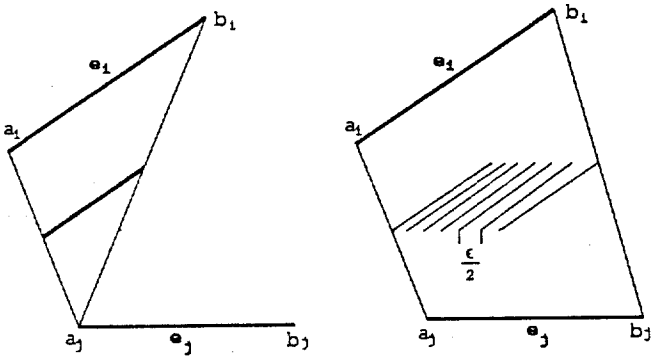

Figure 5: Parallelogram generated by two edges.

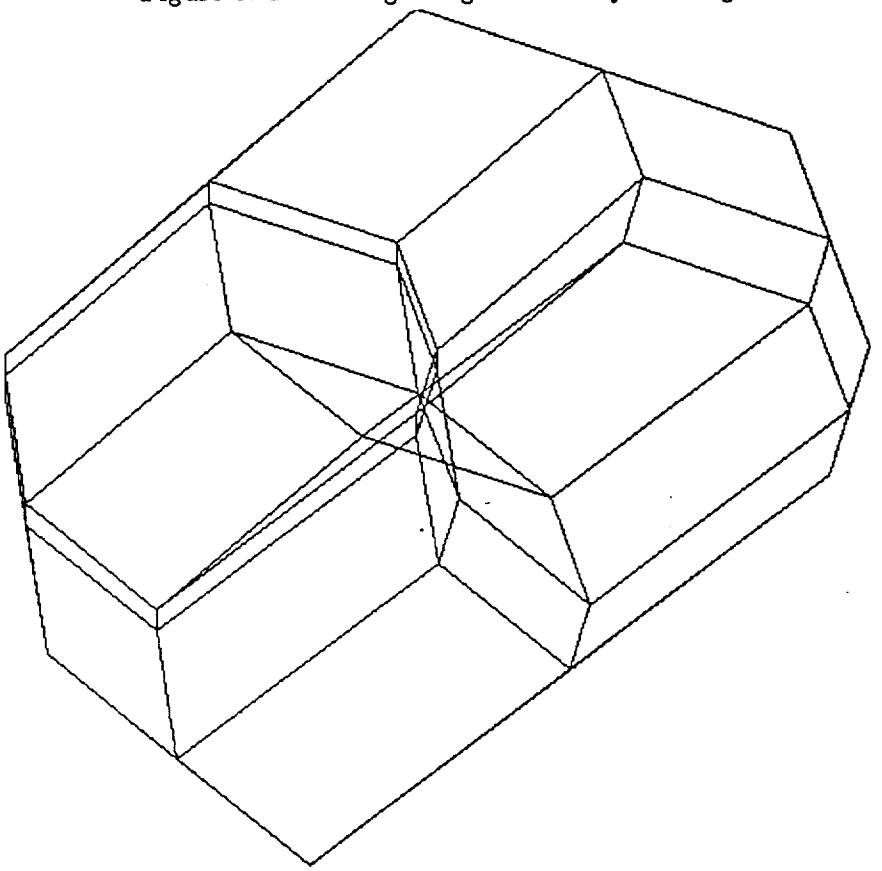

Figure 6: Parallelograms for a polygon.

where the constant c_θ depends on the angle formed by e_i and e_j and c_d depends upon the distance from e_i to the intersection of e_i and e_j. (The equation for the special case where e_i and e_j are parallel has a much simpler form). The important point is that these functions are simple and depend only on constants determined by the edges of P. This is the reason for starting with the regions $R_{i,j}$.

By Lemma 2.1 we know that $C(p, \theta^*)$ must be balanced, and hence each parallelogram in which a point p is contained determines a candidate for $C(p, \theta^*)$. For a point p, there are only a linear number of pairs of edges that may be intersected by a line through p, yielding a linear upper bound to the number of parallelograms, $R_{i,j}$ to which p can belong, and thus a linear number of candidates for $C(p, \theta^*)$.

The value of $\alpha(p)$ is then determined as the minimum among these candidates:

$$\alpha(p) = \min \alpha_{i,j}(p) \quad \forall\ R_{i,j} \ni p$$

As mentioned earlier, boundaries between two regions of the chord diagram occur when $\alpha(p)$ for a point is realized by two or more $\alpha_{i,j}(p)$'s (the label set has size greater than 1). Explicit algebraic computation of these boundaries, by intersecting two surfaces with equations of the form Equation (1), show that they are actually pieces of hyperbolic curves.[3]

It is known [Grü63] that at least 3 min-chords must pass through the center of area, so given the chord diagram the center of area can be found by examining points at which 3 or more boundaries meet.

Figure 7 represents a sample chord diagram, Figure 8 is an enlarged view of the region surrounding the center of area, and Figure 9 is an overlay of the chord diagram of Figure 8 on the contours at 2% intervals. These figures were generated using the same "raster-line" approximation as the contours in Figure 2.

3.2 The Algorithm

The algorithm proceeds as follows:

Algorithm 3.1

1. *For each pair of edges, e_i and e_j generate $R_{i,j}$. This yields $O(n^2)$ parallelograms.*

[3] See [Dia89] for more details.

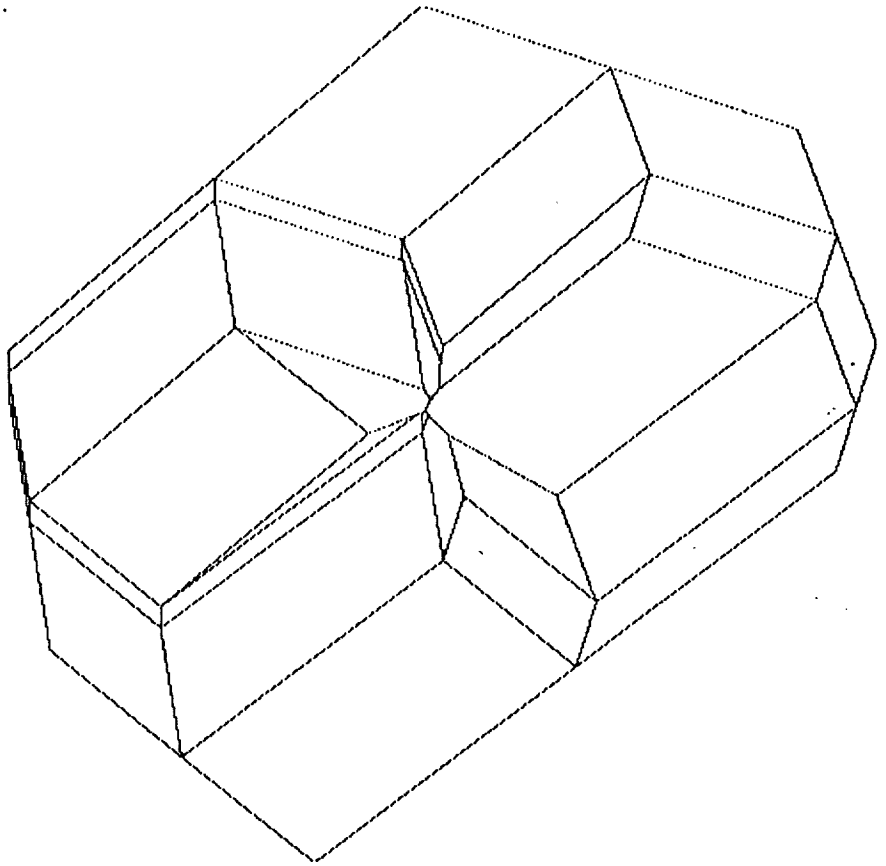

Figure 7: Chord diagram (for the same polygon shown in Figure 6).

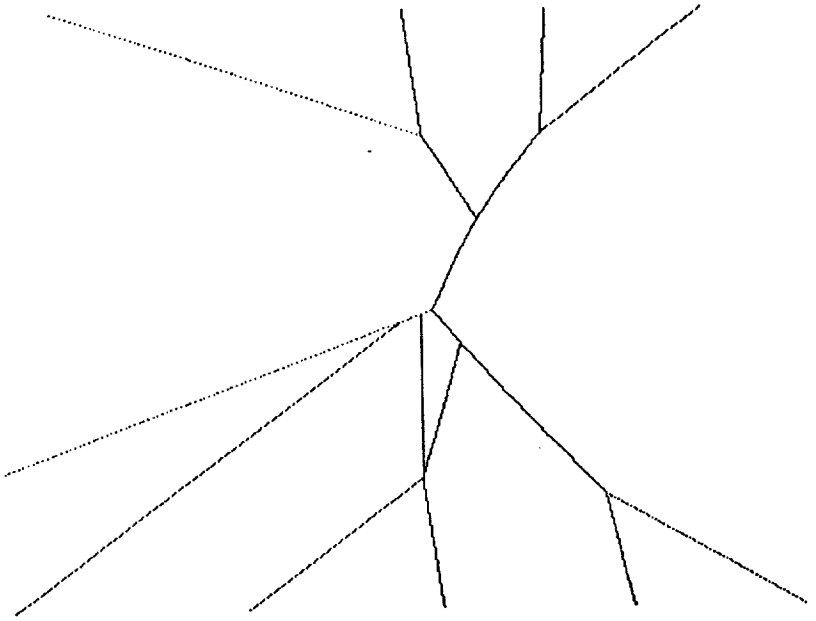

Figure 8: Chord diagram enlargement.

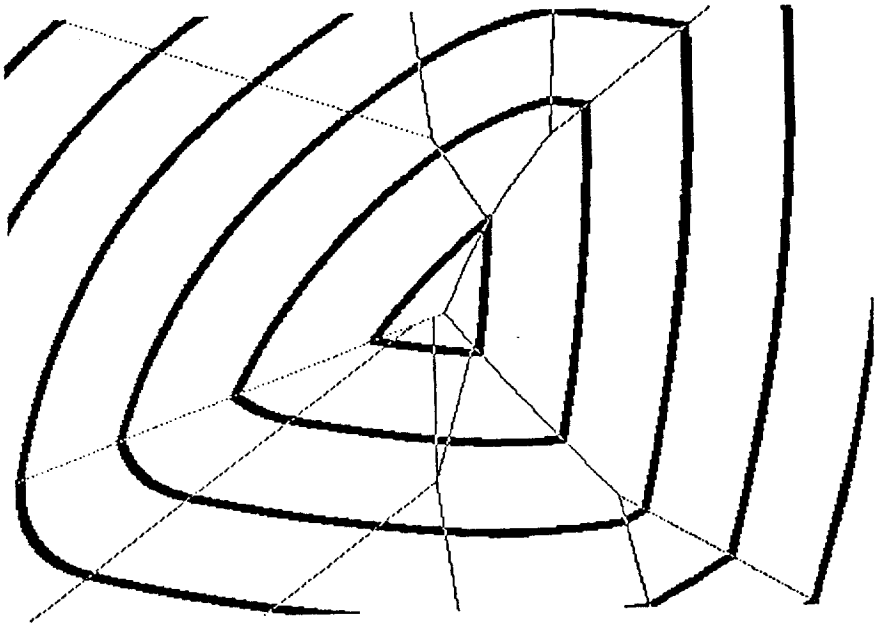

Figure 9: Combined chord diagram and contours.

2. *Determine all the regions formed by the intersection of these parallelograms. This generates $O(n^4)$ regions.*

3. *Over each region, determine the lower envelope of the $\alpha_{i,j}(p)$'s. We will show below that for the $O(n)$ functions associated with each of the $O(n^4)$ regions, this step has complexity $O(n^6 \log^2 n)$.*

4. *Check the value for each of the $O(n^6)$ vertices at constant time per vertex to determine the center of area.*

For Step 3 there are two issues to consider: the combinatorial complexity of the lower envelope of the $\alpha_{i,j}(p)$'s, and the complexity of determining this envelope. Work on Davenport-Schinzel sequences provides bounds for the former given that the functions satisfy certain requirements [EPSS87], [SS87]. The two requirements particular to this case are that:

1. Any two of the functions intersect in either a closed curve or are unbounded.

2. Any three of the functions intersect at no more than two points.

For N functions satisfying these requirements, the complexity of the lower envelope will be $O(N^2)$.

We first define the function $\alpha'_{i,j}(p)$ as $\alpha_{i,j}(p)$ but not restricted to the parallelogram $R_{i,j}$. It can then be shown that the intersection of any two of the $\alpha'_{i,j}(p)$'s results in one branch of a hyperbola [Día89], satisfying the first requirement, and since any two (single branch) hyperbolas can intersect in at most two points, the second requirement is also satisfied. This indicates that for $O(N)$ $\alpha'_{i,j}(p)$'s the complexity of the lower envelope for the entire plan is $O(N^2)$, and thus the complexity restricted to the region over which $\alpha_{i,j}(p)$ is defined can be no worse than $O(N^2)$. The complexity of finding this envelope, however, is $O(N^2 \log^2 N)$ [Sha88].

We can now compute the time complexity for Step 3 of the algorithm. We have $O(n^4)$ regions of the polygon, and over each region we have $O(n)$ functions (since, from Section 3.1, each point can belong to only a linear number of $R_{i,j}$'s). Computing the lower envelope for each of these regions requires $O(n^2 \log^2 n)$ time, giving a total complexity for step 3 of $O(n^6 \log^2 n)$.

We believe that the complexity of this algorithm could be significantly reduced, but the effort involved would be of questionable value due to the ease of implementation and speed of the following numerical algorithm.

4 A Numerical Approach

Because finding the center of area can be viewed as finding the maximum of the bivariate function $\alpha(p)$, it is natural to try procedures for finding maxima of functions. We implemented a hill-climbing algorithm, but it was highly unstable due to the discontinuities in the derivative of the surface: each seam between a pair of functions $\alpha_{i,j}(p)$ adjacent on the lower envelope is such a discontinuity, and Grünbaum's result guarantees that there will be at least three such seams meeting at the center of area (see Figure 9). Therefore we developed a more robust approach specific to this problem.

To understand how this numerical algorithm works we will first consider a simpler algorithm. This algorithm starts with a search space, S, picks a point $p \in S$, and determines $C(p, \theta^*)$. We know from contour property 3 that $C(p, \theta^*)$ supports a contour at p. Hence the center of area must be to the *right* of $C(p, \theta^*)$, permitting the search space to be reduced by the area to the *left* of $C(p, \theta^*)$. One necessary condition for linear convergence is for a fixed percentage of the area of S to be removed at each iteration. Although we have no control over which line through p will be the min-line, by Winternitz's Theorem if we pick p to be the center of mass, then *any* line through p is guaranteed to cut off $\geq 4/9$ of the area of S. This leads to the following algorithm:

Algorithm 4.1 *Start with polygon P and search polygon $S_0 = P$.*

1. *Set p_i to the center of mass of S_i.*
2. *Find $C(p_i, \theta^*)$ using P. If more than one min-chord, choose one arbitrarily.*
3. *Set $S_{i+1} = S_i - L^+(p_i, \theta^*)$.*
4. *If not at desired precision, goto 1.*

Both steps 1 and 3 are straightforward and can be done in $O(n)$ time. Step 2 can also be done in time $O(n)$ as follows: By Lemma 2.1 we know that for a given point p, $C(p, \theta^*)$ must be balanced. Hence there are only a linear number of choices of pairs of edges from P that need to be checked. By scanning the edges of P in order it is possible to find the balanced chord and update the area to the left of $C(p, \theta)$ in constant time per edge. Thus the minimum can be found in $O(n)$ time.

Although after K iterations the area of S has been reduced by $(4/9)^K = O(1/2^K)$, this does not necessarily mean that the coordinates of the center of area are converging. In fact, there exist polygons such that S may converge to a line segment and not a point. (Figure 10.)

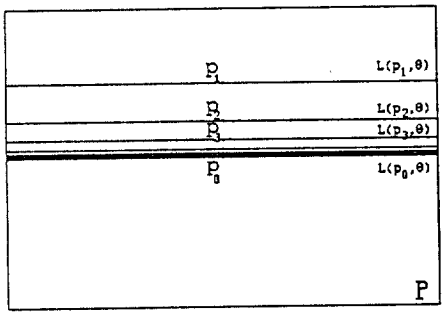

Figure 10: Convergence to a line using Algorithm 4.1

We found two methods to remedy this problem. The first is a procedure for determining to which side of an arbitrary chord of P the center of area must lie. Unfortunately, this leads to a slower and less stable algorithm. The second method follows the basic structure of Algorithm 4.1, but ensures that the diameter as well as the area of S converges. This is achieved by splitting S into two pieces, and selecting appropriate points within each piece. An outline of the algorithm follows.

Algorithm 4.2 *Start with polygon P and search polygon $S_0 = P$.*

1. *Set $D = diam(S_i)$, and split S_i into R_i and L_i along the perpendicular bisector to D.*
2. *Set r_i and l_i to be the centers of mass of R_i and L_i respectively.*
3. *Find $C(l_i, \theta^*)$ and $C(r_i, \theta^*)$ using P.*
4. *Set $S_{i+1} = S_i - L^+(l_i, \theta^*)$.*
5. *If $r_i \in S_{i+1}$, set $S_{i+1} = S_{i+1} - L^+(r_i, \theta^*)$.*

6. If not at desired precision, goto 1.

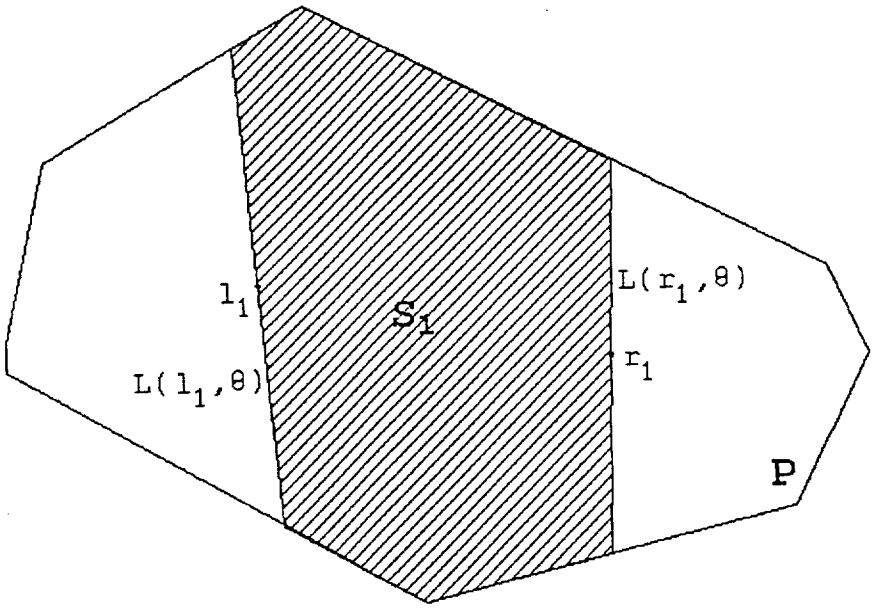

Figure 11: Algorithm 4.2 after the first iteration.

Step 1 can be done in $O(n)$ time [PS85] and Steps 2-5 are essentially the same as in Algorithm 4.1. The benefit of this algorithm is that it can be shown that the diameter of S_i converges, in the limit, as $i \to \infty$.

Theorem 4.1 *Using Algorithm 4.2, the diameter of S_i converges linearly to a point as $i \to \infty$.*

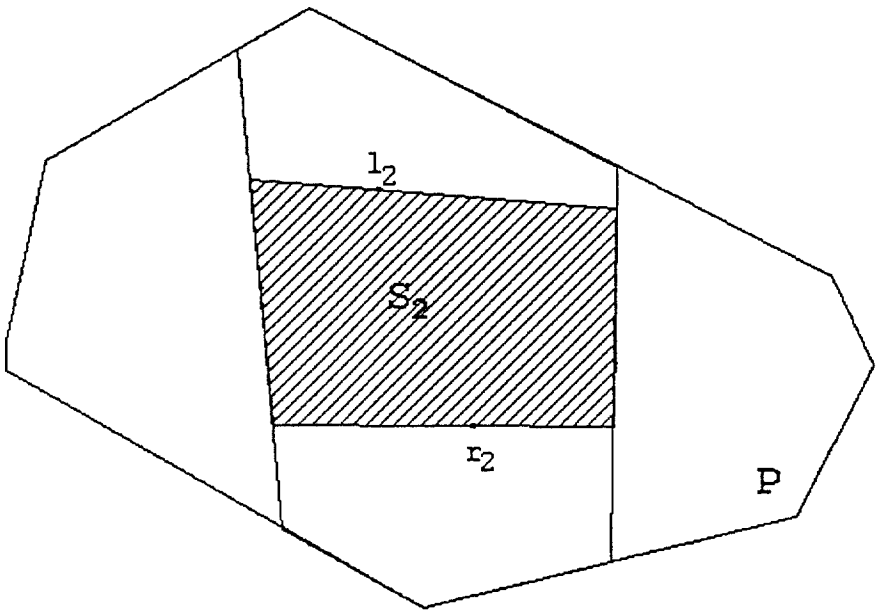

Figure 12: Algorithm 4.2 after the second iteration.

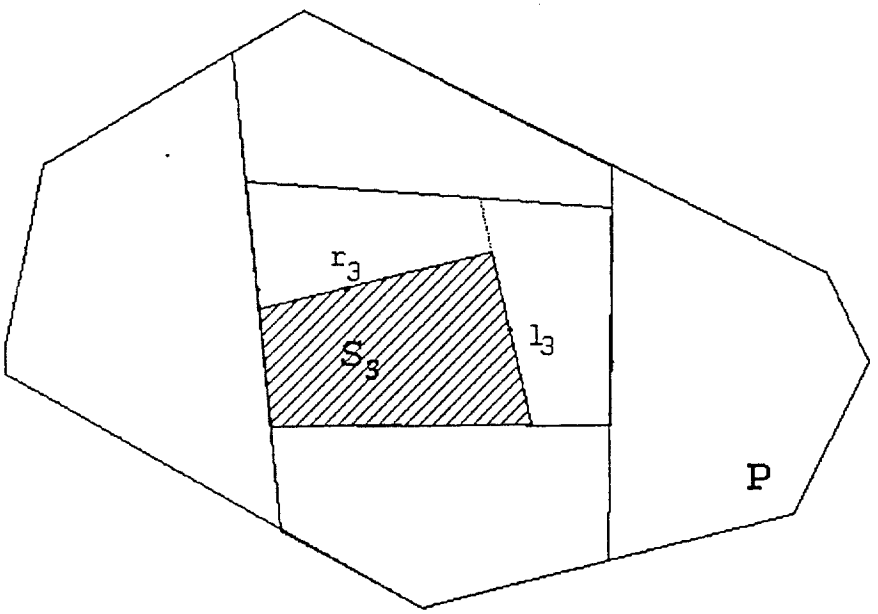

Figure 13: Algorithm 4.2 after the third iteration.

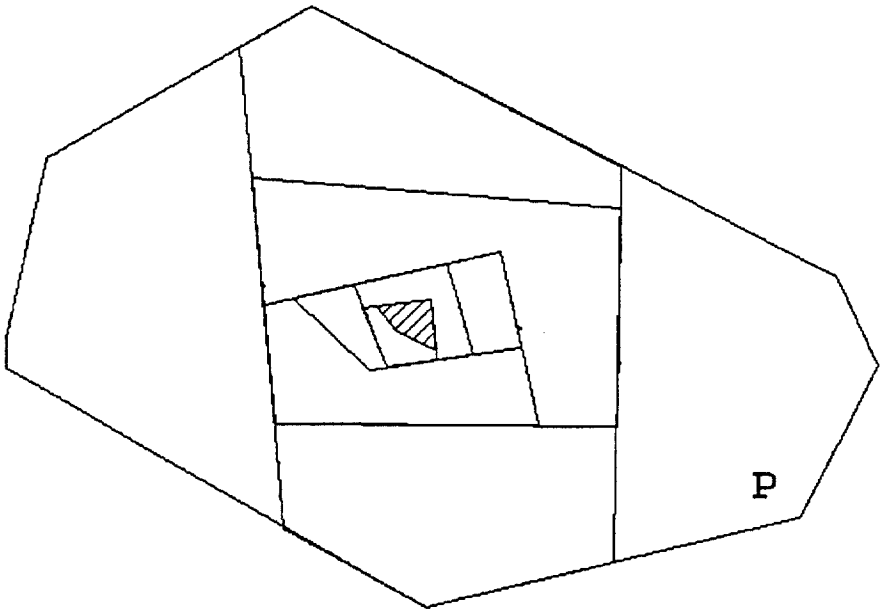

Figure 14: Algorithm 4.2 after several iterations.

Proof We first show that S_i must converge to a point. To do this, assume the contrary. Then it must be the case that there exists some non-zero length, l_{min}, that forms a lower bound on the lengths of the diameters of S_i for all i. (In other words if we let $d_i = diam(S_i)$ then $\mid d_i \mid \geq l_{min}\ \forall\ i$.) The algorithm in steps 4 and 5, by removing a fixed percentage of the area of S_i, forces the area to decrease linearly at each iteration. With the area decreasing and the diameter length fixed by l_{min}, it it must be the case that S_i converges to a line segment (To see this, let B_i be the smallest box surrounding S_i with length l_{min} and width ϵ. Since each S_i is convex, and their area approaches 0, it must be the case that ϵ approaches 0, leaving S_i as a line segment of length l_{min}.)

As S_i converges to a segment of length l_{min}, the centers of mass of each side, r_i and l_i, converge to a separation of $l_{min}/2$. The min-lines, $L(r_i, \theta^*)$ and $L(l_i, \theta^*)$, determine the portion of S_i to be removed, and hence are restricted by the requirement that the length of d_i stay above l_{min}. In particular, the angle of $L(p, \theta^*)$ with respect to d_i can be at most $\tan^{-1}(2\epsilon/l_{min})$ (Figure 15).

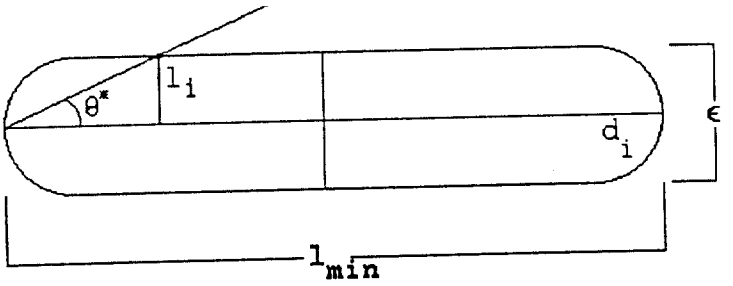

Figure 15: Maximum permissible angle for $L(p, \theta^*)$

As ϵ approaches 0, this $\tan^{-1}$ approaches 0, and hence $L(r_i, \theta^*)$ becomes colinear with $L(l_i, \theta^*)$. By Lemma 2.1, it must be the case that both r_i and l_i are the midpoints of $L(r_i, \theta^*)$ and $L(l_i, \theta^*)$ respectively, and since these lines are colinear, we have that $r_i = l_i$. However, we have shown that when the diameter does not converge, r_i and l_i are separated by $l_{min}/2$, a contradiction. Hence the diameter of S_i converges to a point. That it converges linearly follows from the linear convergence of the area. (If the area goes to zero and the diameter converges, then it too must converge at the same asymptotic rate). □

The implication of Theorem 4.1 is that Algorithm 4.2 can compute the coordinates of the center of area to K bits of precision in $O(nK)$ time,

in other words, $O(n)$ work per bit of precision. Note that the theorem does not claim a linear diameter convergence rate for *each* iteration, but only eventually. Our experience is, however, that "eventually" happens immediately, as discussed below.

5 Empirical Results

Algorithm 4.2 was implemented in the C programming language on a VAX 8530. Tests were performed to examine the expected performance of the algorithm for random convex polygons. The input polygons were created by picking points uniformly from a square and taking their convex hull. Figure 16 shows a plot of the area of the search polygon S and the length of the diameter on a logarithmic y axis versus iteration. The graph shows that, as expected, both the area and the diameter of the search polygon decrease a constant number of bits per iteration, with the area decreasing roughly twice as fast as the diameter.

6 Concluding Remarks

We have presented two algorithms for computing the center of area the point that characterizes "Winternitz's Measure of Symmetry" for a convex polygon P. The second algorithm (4.2) is easily implemented, fast and robust in practice. This is an interesting case where an "exact" combinatorial algorithm would not only be more complicated than the simple numerical algorithm, but would likely yield inferior numerical results if actually implemented.

There are several generalizations [Grü63] that have not been completely explored, particularly algorithmically. One is to extend the definition of the center of area to non-convex polygons. It can be shown that the contour properties still hold, that again the center of area is a unique point and that Lemma 2.1 can be extended to provide a generalized balance condition [DO].

We can also move to higher dimensions by replacing area with volume and min-lines with min-hyperplanes, to get the center of volume. Additionally we can replace area with surface area to get (in 2 dimensions) the center of perimeter. We expect that the numerical techniques developed here will prove useful in these generalizations.