Hausdorff Core of a One Reflex Vertex Polygon

Robert Fraser* Patrick K. Nicholson†

Abstract

In this paper we present a polynomial time algorithm for computing a Hausdorff core of a polygon with a single reflex vertex. A Hausdorff core of a polygon P is a convex polygon Q contained inside P which minimizes the Hausdorff distance between P and Q. Our algorithm essentially consists of rotating a line about the reflex vertex; this line defines a convex polygon by cutting P. To determine the angle at which the line should be rotated, our algorithm searches for a global minimum on the upper envelope of n continuous piecewise functions, where n is the number of vertices of P.

1 Introduction

Our work is inspired by the problem of providing an approximation of a polygon which is as close as possible to the original polygon. There are a number of plausible measures of 'closeness' that one can choose to use, such as an approximation which minimizes the Hausdorff distance, Fréchet distance, or difference in area between the approximation and the original polygon. In this work, we are interested in minimizing the Hausdorff distance, while additionally restricting that the approximating polygon is convex and fully contained in the original polygon. We formalize this notion presently with the definition of the *Hausdorff core* of a polygon.

Given a simple polygon P, a *Hausdorff core* of P is a convex polygon Q, such that $Q \subseteq P$ and $H(P, Q)$ is minimized, where $H(P, Q)$ denotes the Hausdorff distance between P and Q. Throughout this work, we will restrict ourselves to simple polygons, where we define a polygon P as a closed region of the plane, with the boundary of the polygon ∂P represented as a polygonal chain with n vertices $\partial P = \{p_1, p_2, \ldots, p_n\}$.

1.1 Related Work

The problem of approximating polygons may be broadly classified as approximations which are allowed to intersect the original polygon P, approximations which enclose P, or approximations which are entirely included in P. We restrict our discussion to this latter class, of which the best known example is the potato-peeling problem of Chang and Yap [2]. The 'potato' is formally known as the maximum area convex subset (MACS)[3], which is the largest area convex polygon contained in P. There is an $O(n^7)$ time algorithm for this problem, and an $O(n^6)$ time algorithm for maximizing the perimeter of a contained convex polygon, where n is the number of vertices of P [2]. Recently, Dorrigiv et al. [5] provided a PTAS for the general Hausdorff core problem on simple polygons. Their technique noted that the problem may be reduced to a disk stabbing problem with the restriction that the stabbing sequence does not cross outside of the original polygon. A triangle of maximal area contained in a convex polygon can be found in linear time [4], while the regular k-gon of maximal area can be found in time $O(kn + n \log n)$ [1].

Studies which use the Hausdorff measure for approximation have generally used restricted classes of polygons. Lopez and Reisner [6] presented polynomial-time approximation algorithms for approximating a convex polygon with a convex polygon of lower complexity. The dual problem was also studied, where the solution provides the minimum number of vertices of the approximating polygon given a maximum allowed Hausdorff distance. They show that both inclusion and enclosure problems can be approximated to within one vertex of the optimal in $O(n \log n)$ time and $O(n)$ time, respectively.

Chassery and Coeurjolly [3] presented an algorithm for determining a Hausdorff core of a polygon by shrinking the input polygon P until its convex hull is contained in the original P. If the shrunken polygon P' is not convex, then the convex hull of P' contains a vertex of P which lies on an edge e of P'. e is used as a cutting line upon P to obtain a new polygon P_1 to be shrunk. The procedure is repeated to obtain P_i' from P_i until P_i' is convex. If the Euclidean 1-centre of P is not contained in P, it is possible to construct examples where this algorithm would not return a Hausdorff core of P, as shown in [5].

2 Hausdorff Distance

We define the Hausdorff distance $H(P, Q)$ between polygons P and Q as

$$H(P,Q) = \max\left\{\max_{p\in P}\min_{q\in Q}\operatorname{dist}(p,q), \max_{q\in Q}\min_{p\in P}\operatorname{dist}(p,q)\right\},$$

*David R. Cheriton School of Computer Science, University of Waterloo, r3fraser@cs.uwaterloo.ca

†David R. Cheriton School of Computer Science, University of Waterloo, p3nichol@cs.uwaterloo.ca

where $\text{dist}(p, q)$ denotes the Euclidean distance between points p and q. Given a simple polygon P, a *Hausdorff core* of P is a polygon Q where Q is convex, $Q \subseteq P$, and $H(P, Q)$ is minimized. There is not necessarily a unique Hausdorff core solution; our objective is to determine any arbitrary optimal solution.

Lemma 1 *One edge of a Hausdorff core Q will be a segment of a line ℓ which intersects the reflex vertex r.*

Proof. Suppose that this is not the case. ℓ intersects ∂P at two points, call them ℓ_1 and ℓ_2. Choosing ℓ_1 w.l.o.g., rotate ℓ about ℓ_1 until it intersects r; call the resulting polygon Q'. Q' remains convex because ℓ remains a straight line defining the cut, and $Q \subseteq Q'$. Therefore, $H(P, Q) \geq H(P, Q')$. □

Given that ℓ intersects r, we can regard the resulting partition of the polygon as being composed of three convex regions (one of which may be empty), called Q_a, Q_b, and Q_c. We will define these in counter-clockwise order around r so that the line segment $\overline{\ell_1 r}$ defines an edge of Q_a, $\overline{\ell_1 \ell_2}$ defines an edge of Q_b, and $\overline{r \ell_2}$ defines an edge of Q_c. This arrangement is illustrated in Figure 1.

Lemma 2 *The optimal solution polygon Q can always be defined by some Q_b.*

Proof. Suppose that this is not the case, and instead Q is defined by Q_a w.l.o.g.. Choosing ℓ so that $\overline{rp_1}$ lies on ℓ creates an arrangement where Q'_a is empty and $Q_a \subset Q'_b$ for all possible polygons Q_a. Therefore, $H(P, Q_a) \geq H(P, Q'_b)$. □

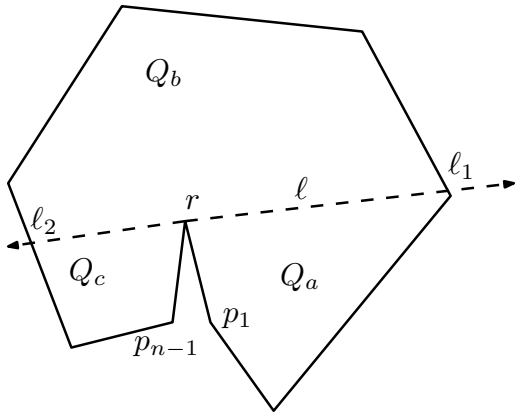

Figure 1: The line ℓ defines a cut through polygon P, which defines three convex regions Q_a, Q_b, and Q_c.

2.1 Measuring the Hausdorff Distance in Polygons

In our problem, we are interested in measuring the distance from the vertices of polygon P to the approximating polygon Q. We divide the vertices of ∂P into two sets ∂P_I and ∂P_E, where ∂P_I are the vertices of ∂P in $P \cap Q$, and $\partial P_E = \partial P - \partial P_I$.

Lemma 3 *The maximum distance from P to Q occurs at a vertex of P.*

This is a weaker version of Lemma 1 in [5], which states that the maximum distance from P to Q occurs at a vertex of the convex hull of P.

Since the vertices of P_I do not need to be considered for the Hausdorff distance measure, as they are contained within Q, the nature of the problem is reduced to finding the distance from the vertices of P_E to the closest point on $\ell \cap P$. There are two cases to consider: either the nearest point on $\ell \cap P$ is somewhere on the interior of the line segment, or the nearest point lies at one of the endpoints.

If the former case is true, then the technique for measuring the distance is trivial. Suppose we are considering vertex p_i, and the nearest point on ℓ is ℓ_i. In this case the points (r, p_i, ℓ_i) define a right-angled triangle, and the distance between p_i and r is fixed. Therefore, the distance from p_i to ℓ is defined as $\text{dist}(p_i, \ell_i) = \text{dist}(p_i, r) \sin\theta$, where $\theta = \angle(p_i, r, \ell_i)$.

In the latter case, describing the nearest point is slightly more difficult, as it is either ℓ_1 or ℓ_2 . For our purposes here we will assume w.l.o.g. that the nearest point is ℓ_1. We express $\text{dist}(p_i, \ell_1)$ in terms of the intersection between the ray $\overrightarrow{r\ell_1}$ and ∂P. We formalize this case further in the next section.

3 Finding the Optimal Solution

Our algorithm essentially consists of sweeping ℓ through P by rotating it about r to discover the angle at which the distance between vertices of P_E and $\ell \cap P$ is minimized. For convenience of exposition, we rotate P about r by the angle required to align rp_1 horizontally; i.e. $p_1 = (r_x + \text{dist}(r_x, p_1), r_y)$. We then perform the sweep in a counterclockwise fashion, such that $\theta_1 = 0$ is the initial state when ℓ intersects p_1, and $\theta_{n-1} < \pi$ is the final state when ℓ intersects p_{n-1}.

3.1 Expressing the Distance to Each Point

For each point p_i, we define a set of intervals I_i using the circle C_i with diameter $\overline{rp_i}$. Suppose that the circle C_i intersects P at k points, $c_1, ..., c_k$, distinct from any p_i. We define the points $t_1, ..., t_{n+k-1}$ to be $\left(\cup_{d=1}^{k} c_d\right) \cup \left(\cup_{e=1}^{n-1} p_e\right)$, such that $t_1 = p_1$ and each t_j follows t_{j-1} by moving counterclockwise along the perimeter of P. Then $I_i = \{[t_1, t_2), [t_2, t_3), ..., [t_{n+k-2}, t_{n+k-1}]\}$. Thus, each non-reflex vertex in P and each intersection point between C_i and P appear as endpoints of the intervals in I_i.

Each interval, $[t_j, t_{j+1})$, is either contained entirely within C_i, or entirely outside C_i. In the former case we refer to the interval as an *inner interval*, and in the latter case an *outer interval*. Let I_i^+ be the set containing all inner intervals from I_i, and I_i^- be set containing all outer intervals from I_i.

Observation 1 *Suppose for some value of θ, ℓ_1 is contained within the interval $[t_j, t_{j+1})$. Then if $[t_j, t_{j+1}) \in I_i^+$, the nearest point on ℓ to p_i is ℓ_1; otherwise the nearest point lies on the interior of the line segment $\overline{r\ell_1}$. An identical claim can be made when the nearest point is ℓ_2.*

We now define the function $g(p_i, \theta)$ to be the distance between $p_i \in P_E$ and ℓ, when ℓ is rotated clockwise by an angle of θ from its starting orientation θ_1. Since P_E changes as ℓ rotates through a vertex, we actually keep track of all $p_i \in P$, noting that $g(p_i, \theta) = 0$ when $p_i \notin P_E$. As we discussed earlier, $g(p_i, \theta)$ for fixed p_i is a piecewise continuous function. As we vary θ, the function used to compute the distance changes depending on whether the nearest point from p_i to ℓ is contained in $\overline{r\ell_1}$ or $\overline{r\ell_2}$; w.l.o.g., we assume the nearest point p_i is always contained within the segment $\overline{r\ell_1}$. Next, suppose that ℓ_1 lies within the interval (u, v). If (u, v) is an outer interval, then we can compute the distance to $\ell \cap P$ using a right triangle. If (u, v) is an inner interval, then the minimum distance to $\ell \cap P$ occurs at the endpoint ℓ_1. Otherwise, we express ℓ_1 as the intersection between the ray $\overrightarrow{rl_1}$ and the segment (u, v). Thus we have:

$$g(p_i, \theta) = \begin{cases} \text{dist}(p_i, r)\sin(|\theta_i - \theta|) & \text{if } (u, v) \in I_i^- \\ \text{dist}(p_i, \ell_1) & \text{otherwise} \end{cases} \quad (1)$$

where

$$\ell_1 = (u_x + z(v_x - u_x), u_y + z(v_y - u_y)) \ , \quad (2)$$

and

$$z = \frac{r_y - u_y + \tan\theta(u_x - r_x)}{v_y - u_y - \tan\theta(v_x - u_x)} \ . \quad (3)$$

Since C_i can intersect P at most $O(n)$ times, $k \in O(n)$. Thus, for each p_i, $g(p_i, \theta)$ consists of $O(n)$ portions. We also note that $g(p_i, \theta)$ is continuous throughout the domain $[0, \theta_{n-1}]$, and that each portion of $g(p_i, \theta)$ is unimodal for its domain. Thus, a minima of one portion can be found in constant time.

3.2 Minimizing the Maximum Distance

We have discussed how to compute the distance between a single vertex of P, p_i, and the line ℓ. However, our objective is to minimize the maximum distance between P_E and $\ell \cap P$. To do this, we need to compute:

$$\min_{0 \le \theta \le \theta_{n-1}} G(\theta) \ , \quad (4)$$

where

$$G(\theta) = \max_{1 \le i \le n-1} g(p_i, \theta) \ . \quad (5)$$

We now describe how to merge two piecewise continuous functions $g(p_j, \theta)$ and $g(p_{j+1}, \theta)$, to create a new piecewise continuous function $g'(\theta) = \max\{g(p_j, \theta), g(p_{j+1}, \theta)\}$. To compute the intersection points between two overlapping portions of $g(p_j, \theta)$ and $g(p_{j+1}, \theta)$, we observe that there are three cases: i) both portions represent inner intervals, ii) both portions represent outer intervals, and iii) one portion is an inner interval and the other is an outer interval. In all three cases the intersection points have analytic solutions[1]. Furthermore, since the domain of each function is $[0, \theta_{n-1}]$ and $\theta_{n-1} < \pi$, the number of intersection points between any two portions is constant. This means that the number of intersection points between $g(p_j, \theta)$ and $g(p_{j+1}, \theta)$ is $O(n)$.

Continuing this process, we can construct $G(\theta)$ by performing $n-2$ merge steps like the one just described. Since we have $n-1$ continuous functions such that each pair have $O(n)$ intersections, we can apply the upper bound from [7, p.8]. Therefore, the upper envelope of the functions, $G(\theta)$, consists of at most $O(n^3)$ portions. Each of these pieces can be examined in constant time to find the minimum value. Thus the overall running time is polynomial in n.

4 Example

In this section, we provide an example (Figure 2) to illustrate the operation of our algorithm. Consider the vertex p_1 in Figure 2. To begin with, we set $\theta = 0$ and the current solution ℓ intersects p_1. As θ is increased, the minimum distance to ℓ follows along $\overline{p_1 p_2}$ until point t_3, when the minimum distance function changes to the sin function to follow the path of the circle through the interior of P until it meets the boundary at point A. At this point, the minimum distance function changes again to follow $\overline{p_3 p_4}$ to point p_4, at which point our sweep concludes.

The point of intersection between the two functions converging on the right of Figure 3 lie just outside of the range of the graph, ($\theta = 161.58$, while viable solutions lie in the range $\theta = 0 \ldots 161.565$). The minimal value of the upper boundary of the graph is at $\theta = 161.565$, where the Hausdorff distance $H(P, Q) = \sqrt{(}17)$, corresponding to the distance between points p_1 and p_4.

[1]The closed form solutions are quite large, which is why we do not explicitly state them.

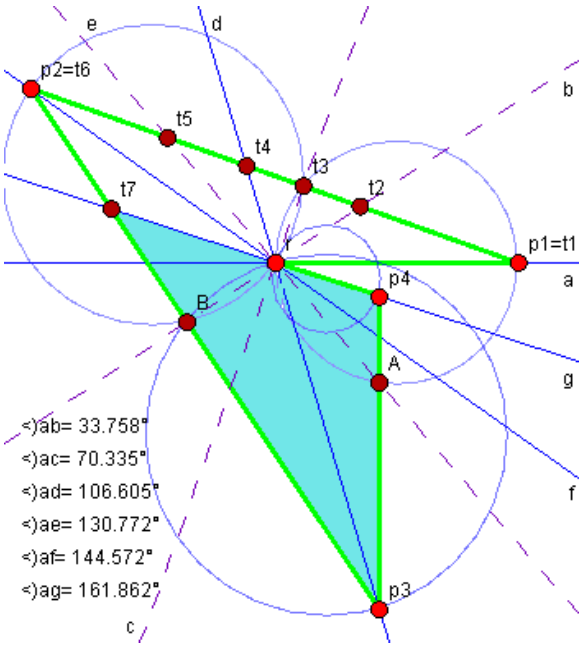

Figure 2: A polygon P with reflex vertex r is shown. Lines indicate boundaries between intervals, where solid lines indicate interval boundaries defined by vertices of P, and dashed lines indicate boundaries defined by the transition between distance functions for a vertex. The latter transitions occur at the intersection points of the boundary of polygon with the diametrical circles defined between r and each other vertex $p_i \in P$. The shaded area indicates the optimal solution Q.

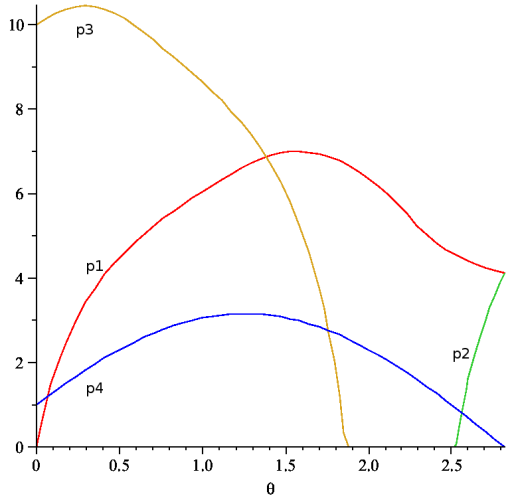

Figure 3: A plot showing the curves corresponding to the distances from the vertices of P to the solution polygon Q for all possible values of θ.

5 Conclusions

In this work we have presented an exact algorithm for determining a Hausdorff core of a polygon with one reflex vertex in polynomial time. The algorithm measures the distance from the excluded vertices of the original polygon P to the cut line defining the approximating polygon Q. The Hausdorff distance between the polygons $H(P,Q)$ for a given cut is described by the maximum such distance, and so the problem may be formulated in terms of a set of piecewise functions whose maximum for a given line corresponds to $H(P,Q)$. By determining the minimum maximum value over all possible cuts, we determine the optimal Hausdorff approximation for P.

The algorithm is not immediately extendable to polygons containing more than one reflex vertex, since the problem is complicated by the interaction between the cutting lines. It is also interesting to compare our problem to the potato peeling problem. In the potato peeling problem, if P has only one reflex vertex then finding the optimal solution is trivial; only three cases need to be considered [2]. This seems to suggest that finding the exact Hausdorff core is a more difficult problem in general.

Acknowledgements

We thank Joseph Cheriyan, Nathan Krislock, and Marcel Silva for discussions on optimization problems, and the members of the algorithms group at Waterloo in general. We gratefully acknowledge the detailed revisions and suggestions provided by the reviewers.

References

[1] A. Aggarwal, M.M. Klawe, S. Moran, P. Shor, and R. Wilber. Geometric applications of a matrix-searching algorithm. *Alg.*, 2(1):195–208, 1987.

[2] J. S. Chang and C. K. Yap. A polynomial solution for the potato-peeling problem. *Disc. & Comp. Geom.*, 1(1):155–182, 1986.

[3] J.-M. Chassery and D. Coeurjolly. Optimal shape and inclusion. In *Mathematical Morphology: 40 Years On*, volume 30, pages 229–248. Springer, 2005.

[4] D.P. Dobkin and L. Snyder. On a general method for maximizing and minimizing among certain geometric problems. In *Proc. SFCS*, pages 9–17, 1979.

[5] R. Dorrigiv, S. Durocher, A. Farzan, R. Fraser, A. López-Ortiz, J. Ian Munro, A. Salinger and M. Skala. Finding a Hausdorff Core of a Polygon: On Convex Polygon Containment with Bounded Hausdorff Distance. *Proc. of WADS*, LNCS 5664, pp. 218-229, 2009.

[6] M.A. Lopez and S. Reisner. Hausdorff approximation of convex polygons. *Comp. Geom. Theory & App.*, 32(2):139–158, 2005.

[7] M. Sharir and P.K. Agarwal. Davenport-Schinzel sequences and their geometric applications. Cambridge University Press, 1995.