# Robust Point Location in Approximate Polygons

A. James Stewart
Computer Science Department
Cornell University
jstewart@cs.cornell.edu

## Abstract

This paper presents a framework for reasoning about robust geoemtric algorithms. *Robustness* is formally defined and a data structure called an *approximate polygon* is introduced and used to reason about polygons constructed of edges whose positions are uncertain.

A robust algorithm for point location in an approximate polygon is presented. The algorithm uses only the *signature* of the point (not its location) to determine whether the point is inside or outside the polygon.

An approximate polygon could, by shifting its edges back and forth within their error bounds, induce a large number of different line arrangements. The cell $C_\alpha$ with signature $\alpha$ in one such arrangment will be different than the cell $C'_\alpha$ with signature $\alpha$ in another arrangement. This paper proves that, regardless of their positions and shapes, the cells $C_\alpha$ and $C'_\alpha$ are always to the same side of the polygons which induce their respective arrangements.

## Introduction

Most geometric algorithms assume that perfect "real" arithmetic is available. When these algorithms are implemented they often fail because this assumption is not borne out; that is, these algorithms are not *robust*. This failure occurs because either the input or the intermediate calculations are imprecise, leading to inconsistent decisions by the algorithm.

This paper presents a framework for reasoning about robust geometric algorithms which operate on polygons. *Robustness* is formally defined and a data structure called an *approximate polygon* is introduced and used to reason about polygons constructed of edges whose positions are uncertain.

A robust algorithm for point location in an approximate polygon is described. The interesting aspect of this algorithm is that in addition to the polygon's position being uncertain, the point's position in the plane does not have to be known; only the point's *signature* is important (that is, its left/right relations to the edges of the polygon). The point location algorithm has immediate practical application to solid modeling, particularly in the robust intersection of polyhedra.

An approximate polygon could, by shifting its edges back and forth within their error bounds, induce a large number of different line arrangements. In each of these arrangements some points with a given signature $\alpha$ may or may not appear, and if they appear, they may be to the interior or to the exterior of the polygon which induces the arrangement. An interesting *uniqueness theorem* is presented which states that in all such line arrangements, the points with signature $\alpha$ in each arrangement are always to the same side of the polygon which induces that arrangement.

### Practical Applications

The point location algorithm has immediate practical application to solid modeling. In particular, a solid modeler performing an intersection operation needs to determine whether an edge of one polyhedron intersects a face of another. This is achieved by calculating the intersection of the edge with the plane in which the face lies, and then asking whether this point of intersection is on the interior of the polygon representing the face. If the boundary of the face and the location of the point of intersection are known precisely then this is a trivial problem.

However, polyhedra usually have overconstrained faces and vertices, and the *exact* locations of the vertices, edges, and faces of the polyhedra can require a very large number of bits to represent. Since the input is rounded off to a small number of bits the locations of these features are imprecise. In addition, the location of the point of intersection can be completely unknown, particularly in ill-conditioned cases in which the intersecting edge lies very close to the plane of the face. Thus there is an important practical application for a point location algorithm which handles uncertainty in the face boundary and in the point location.

An approximate polygon can represent a face whose boundaries are not known exactly, and the point location algorithm presented in this paper can determine whether a point whose location is also uncertain lies on the interior of such a face. Since both the location of the boundary and the location of the point are uncertain, the algorithm must make use of some other information. This other information consists of the *signature* of the point; that is, its position (left or right) with respect to each edge of the boundary. It is a reasonable assumption that such information exists since the signature is often derivable from logical information available in the solid modeler (for example, see Karasick's modeler [5]).

## Background

The theory of approximate polygons is based upon the "representation and model" approach of Hoffmann, Hopcroft, and Karasick [4]. In this approach the algorithm operates on a computer representation, but presents output as though it were operating on some mathematical model corresponding to the representation.

An approximate polygon is a computer *representation* of some real, mathematical polygon, the *model*. The model is rarely explicitly constructed by the algorithm. An approximate polygon $P_{rep}$ can be thought of as a set of constraints on the topology and position of the implicit model polygon. Any real polygon $P$ satisfying these constraints is considered a model for $P_{rep}$.

Under the representation and model approach, the definition of robustness is very close to that of Fortune [2]. Consider a geometric problem $\mathcal{P}$ as a function from an input space consisting of *models* to an output space, $\mathcal{P} : \mathcal{I} \rightarrow \mathcal{O}$, and consider an algorithm $\mathcal{A}$ as function from a different input space consisting of *representations* to the same output space, $\mathcal{A} : \mathcal{R} \rightarrow \mathcal{O}$. Given a representation $x_{rep}$, the set of its models is denoted $\textsc{models}(x_{rep})$. This leads to a definition of robustness:

> An algorithm $\mathcal{A}$ for a problem $\mathcal{P}$ is *robust* if
>
> $$\forall\ x_{rep} \in \mathcal{R},\ \exists\ x \in \textsc{models}(x_{rep})$$
>
> $$\text{such that } \mathcal{A}(x_{rep}) = \mathcal{P}(x).$$

Note that we can pick an arbitrary $x \in \textsc{models}(x_{rep})$. It could be that there are two models $x^1$ and $x^2$ such that $\mathcal{P}(x^1) \neq \mathcal{P}(x^2)$. In this case the algorithm could choose to output either $\mathcal{P}(x^1)$ or $\mathcal{P}(x^2)$ and would still be considered to be robust. This leads to a definition of consistency:

> A problem $\mathcal{P}$ and a representation $\mathcal{R}$ are *consistent* if
>
> $$\forall\ x_{rep} \in \mathcal{R},\ \ \forall\ x^1, x^2 \in \textsc{models}(x_{rep}),$$
>
> $$\mathcal{P}(x^1) = \mathcal{P}(x^2).$$

A definition of *correctness* would be similar to that of robustness, except that the model and representation spaces would be identical and $\textsc{models}(x_{rep}) = \{x_{rep}\}$.

In evaluating geometric algorithms which use the representation and model approach, the criteria of robustness and consistency should be used in place of the usual criterion of correctness.

Note that, unlike in Fortune's work [2], there is no notion of *stability* in the definition of robustness. That is, there is no notion of the distance between the representation $x_{rep}$ and the model $x$ which allows us to say "the implementation is stable if $x$ is near $x_{rep}$". However, bounds on the models can be achieved by ensuring that $\textsc{models}(x_{rep})$ is sufficiently small.

## Other approaches

The approach with approximate polygons is most similar to that of Milenkovic's hidden variable method [6]. His method constructs arrangements of pseudolines which are constrained to lie within strips of fixed width. The pseudolines can be considered as a model and the strips as a representation. Milenkovic's pseudoline arrangement algorithm can then be said to be provably robust in the sense of the above definition. It is interesting to note that, as with many algorithms of the "representation and model" variety, the model is never explicitly constructed.

There are several other approaches to building robust algorithms (where "robust" is defined differently). Sugihara [10, 11, 12] emphasizes removing redundant decisions from the algorithm in order to maintain topological consistency. Salesin, Stolfi, and Guibas [8] use what they call *epsilon geometry* to reason about the amount of perturbation of the input required for certain *epsilon predicates* to be true. Construction of robust algorithms is based upon these epsilon predicates. Dobkin and Silver [1] keep track of roundoff error and, when the error becomes too large, increase precision and backtrack to some earlier point in the computation. Segal and Sequin [9] alter the symbolic data to make it more amenable to precise computation, and signal the user when tolerances on the input become too large. (Milenkovic also alters the symbolic data in his "data normalization" approach [6].) Greene and Yao [3] discretize the problem domain, allowing the algorithm to perform exact computations.

In the remainder of the paper approximate polygons are defined, some of their properties are enumerated, the point location algorithm is outlined, and the uniqueness theorm for point location in an approximate polygon is presented.

# Definitions

A *polygon* $P = (e_1, e_2, ...e_n)$ is an ordered list of directed edges, where each edge $e_i$ lies on a line $\ell_i$ and only intersects the edges $e_{i-1}$ and $e_{i+1}$ at its endpoints. Each edge is *directed* such that the interior of the polygon it to its right.

An approximate polygon closely mirrors the appearance of a real polygon, as shown in Figure 1. The approximate polygon consists of an ordered list of *bands* corresponding to the edges of the model. The position of the bands in the plane constrains the line equations of the model.

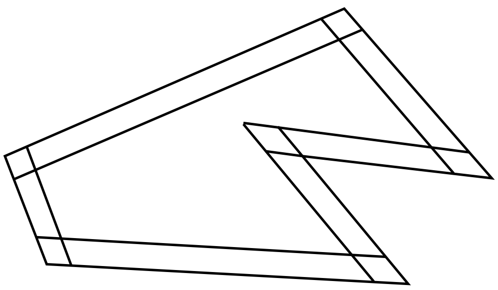

Figure 1: An Approximate Polygon

Just as real polygons are based upon lines, approximate polygons are based on swaths: A *swath* $S_i$ is the region between two lines $\ell_i^{out}$ and $\ell_i^{in}$. These lines have the restriction that $\forall x\ \ell_i^{in}(x) \geq \ell_i^{out}(x)$. The restriction causes the lines to be parallel and conveniently defines the region between them as

$$S_i = \{x \mid \ell_i^{in}(x) \geq 0\ \wedge\ \ell_i^{out}(x) \leq 0\}.$$

Just as an edge is part of a line, a band is part of a swath. Assuming for now that an approximate polygon is represented by an ordered list of swaths, a *band* $B_i$ of an approximate polygon $P_{rep}$ having swaths $S_i$ is the shaded region in Figure 2, defined as

$$B_i = S_i \cap E_{i-1}^i \cap E_{i+1}^i,$$

$$\text{where } E_j^i = \begin{cases} \{x \mid l_j^{out}(x) \leq 0\} & \text{if } i/j \text{ is convex} \\ \{x \mid l_j^{in}(x) \geq 0\} & \text{if } i/j \text{ is reflex.} \end{cases}$$

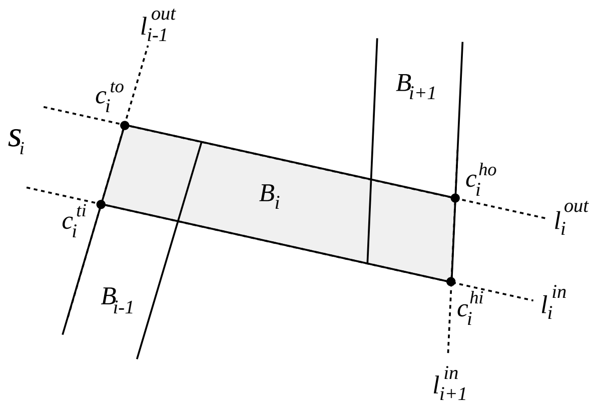


Figure 2: The Pieces of a "Band"

It will be useful to define the corners of a band as $c^{ti}$, $c^{to}$, $c^{hi}$, and $c^{ho}$, where $h$, $t$, $i$, and $o$ denote head, tail, in and out. The definitions are shown in the following table, and depend on whether the bands adjacent to the corner make a convex or a reflex turn. In Figure 2 the tail of $B_i$ is convex, so the definitions for $c^{ti}$ and $c^{to}$ are chosen from the "convex" column of Table 1. Since the head of $B_i$ is reflex, the definitions of $c^{hi}$ and $c^{ho}$ come from the "reflex" column.

| | CONVEX | REFLEX |
|---|---|---|
| $c^{ti}$ | $\ell_i^{in} \cap \ell_{i-1}^{out}$ | $\ell_i^{in} \cap \ell_{i-1}^{in}$ |
| $c^{to}$ | $\ell_i^{out} \cap \ell_{i-1}^{out}$ | $\ell_i^{out} \cap \ell_{i-1}^{in}$ |
| $c^{hi}$ | $\ell_i^{in} \cap \ell_{i+1}^{out}$ | $\ell_i^{in} \cap \ell_{i+1}^{in}$ |
| $c^{ho}$ | $\ell_i^{out} \cap \ell_{i+1}^{out}$ | $\ell_i^{out} \cap \ell_{i+1}^{in}$ |

Table 1: Defining the Band Corners

An *approximate polygon* $P_{rep}$ is an ordered list of bands[1] $B_i$ which lie on swaths $S_i$. $B_i \cap B_j = \emptyset$ *iff* $i$ and $j$ differ by more than one. A real polygon $P$ is a *model* for an approximate polygon $P_{rep}$ if the following constraints are met:

[1]When given as input to an algorithm, the bands are defined exactly with floating point numbers. Subsequent computation on the bands is also done exactly (with extended precision, if necessary).

1. There is a one-to-one correspondence between the lines $\ell_i$ of $P$ and the swaths $S_i$ of $P_{rep}$. Assume that $\ell_i$ corresponds to $S_i$.

2. Each line $\ell_i(x) = 0$ must lie between the corners of band $B_i$. That is, it must satisfy the following four constraints (see Figure 2):

$$\ell_i(c^{ti}) \le 0, \quad \ell_i(c^{to}) \ge 0,$$

$$\ell_i(c^{hi}) \le 0, \quad \ell_i(c^{ho}) \ge 0.$$

It will be useful later on to talk about the *span* of a band. This is the set of points swept out by all lines which fit within the band. The *left* and *right* of a band are the set of those points to the left and right of the span. By convention, the interior of the approximate polygon is to the right of the band. In Figure 3 the shaded region is $\text{SPAN}(B_i)$ and to its left and right are $\text{LEFT}(B_i)$ and $\text{RIGHT}(B_i)$. For a band $B_i$, define the set of lines satisfying Condition 2 above as $\text{LINES}(B_i)$.

$$\begin{array}{rcl} \text{SPAN}(B_i) & = & \{x \mid \exists\, \ell \in \text{LINES}(B_i),\ \ell(x) = 0\} \\ \text{RIGHT}(B_i) & = & \{x \mid \forall\, \ell \in \text{LINES}(B_i),\ \ell(x) < 0\} \\ \text{LEFT}(B_i) & = & \{x \mid \forall\, \ell \in \text{LINES}(B_i),\ \ell(x) > 0\} \end{array}$$

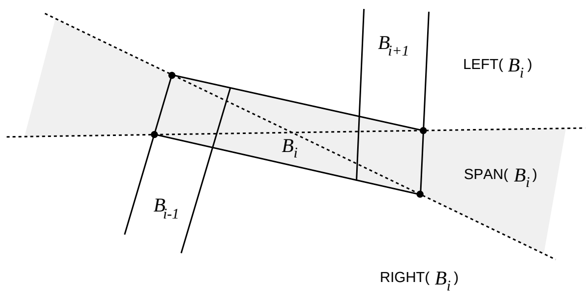


Figure 3: The SPAN of a Band

Some useful properties follow from the previous definitions (these are stated without proof).

1. An approximate polygon is closed and simple.

2. Edge $e_i$ lies completely within $B_i$.

3. Edges $e_i$ and $e_{i+1}$ intersect within $B_i \cap B_{i+1}$.

4. All models of an approximate polygon are simple.

5. $\text{SPAN}(B_i) \cap \text{SPAN}(B_{i+1}) = B_i \cap B_{i+1}$.

6. If $x \notin \text{SPAN}(B_i)$ then in all models, $x$ lies to the same side of $e_i$.

7. If $x \in \text{SPAN}(B_{i+1}) - B_i$ then in all models $x$ lies to the same side of $e_i$.

# Robust Point Location in Approximate Polygons

The point location problem would be simple if the exact location of the point were given. However, in most practical applications the point's location is known only to be within some region of uncertainty. In particularly ill-conditioned situations this region of uncertainty can be as large as the polygon itself.

Some practical applications (geometric modelers, for example) can, from other information, logically deduce the LEFT/RIGHT status of the point with respect to each edge of the polygon. Call this L/R sequence the *signature*. If the polygon's location is known exactly, then in the induced line arrangement a cell decomposition can easily determine whether all points with a given signature lie inside or outside the polygon. It is a different matter, however, when there is uncertainty in the polygon's location. If uncertainty is modeled with an approximate polygon then the following questions must be answered:

> **Question 1 (Robustness)** Given an approximate polygon $P_{rep}$ and a signature $\alpha \in (\text{L}|\text{R})^*$, does $P_{rep}$ have a model $P$ in which the induced line arrangement contains a cell with signature $\alpha$, and is the cell INSIDE or OUTSIDE the model $P$?

> **Question 2 (Consistency)** Consider that an approximate polygon can have two models, $P^1$ and $P^2$, which induce two different line arrangments. These two arrangements each contain a cell with signature $\alpha$ (call them $C^1$ and $C^2$). Then is it possible that $C^1$ is INSIDE $P^1$ and $C^2$ is OUTSIDE $P^2$?

If the answer to Question 2 were affirmative then the signature $\alpha$ and the approximate polygon $P_{rep}$ would not be sufficient information to determine point location, and the problem would not be *consistent*. The Uniqueness Theorem which is presented later proves that this is *not* the case.

## Some final definitions

A *signature* $\alpha^P(v)$ is a string in $(\text{L}|\text{R})^*$. The signature denotes the relation of the point $v$ to each edge $e_i$

of the polygon $P$. The $i_{th}$ element of $\alpha^P(v)$ is the relation of the point $v$ to edge $e_i$ of the polygon $P$. $\alpha_i^P(v) = \text{R}$ means that $v$ is to the right of edge $e_i$ in $P$ and $\alpha_i^P(v) = \text{L}$ means that $v$ is to the left of edge $e_i$ in $P$. The superscripts will be dropped if the polygon in question is evident.

Refer to Figure 3 for the following definitions. A *half-region* is similar to a half-space, except that it has a polygonal boundary. The following half-regions $R_i$ and $L_i$ consist of those points which, in *at least one* model $P$, are either ON $e_i$ or to the RIGHT or LEFT of $e_i$, respectively, in that model. Given some $\alpha_i(v)$, the half-region $H_i$ is that region in whose interior $v$ must lie if it is to have $\alpha_i(v)$ as the $i^{th}$ component of its signature. The *interior* of the cell $\tilde{C}_\alpha$ consists of those points which have signature $\alpha$ in *at least one* model.

$$R_i = \text{SPAN}(B_i) \cup \text{RIGHT}(B_i)$$

$$L_i = \text{SPAN}(B_i) \cup \text{LEFT}(B_i)$$

$$H_i = \begin{cases} R_i & \text{if } \alpha_i = \text{R} \\ L_i & \text{if } \alpha_i = \text{L} \end{cases}$$

$$\tilde{C}_\alpha = \bigcap_{i=1}^{n} H_i$$

The next two lemmas will be used to construct the point location algorithm. The first lemma shows that for each point in $\tilde{C}_\alpha$ there exists some model in which the point has signature $\alpha$; the second lemma shows how to determine whether the point is INSIDE or OUTSIDE that model.

**Lemma 1 (Model Existence)** *Given an approximate polygon $P_{rep}$ and a signature $\alpha$, construct $\tilde{C}_\alpha$ as described above. Then for each point $v$ on the interior of $\tilde{C}_\alpha$, there exists* some *model $P \in \text{MODELS}(P_{rep})$ in which $v$ has signature $\alpha$.*

*Proof* Since $v \in \tilde{C}_\alpha$, for each $i$, $v \in H_i$ and there is some edge $e_i$ in the band $B_i$ which has $v$ to the side specified by $\alpha_i$. These edges join to form a model polygon $P$ in which $v$ has signature $\alpha$. □

**Lemma 2 (Point Location)**
*Given an approximate polygon $P_{rep}$, a model polygon $P \in \text{MODELS}(P_{rep})$, and a point $v$ which has a signature $\alpha$ with respect to $P$, the following are true:*

1. *If $v$ is strictly to the interior of $P_{rep}$ (that is, it does not lie on any band $B_i$) then $\alpha$, $v$* INSIDE *$P$.*

2. *If $v$ is strictly to the exterior of $P_{rep}$ then $v$* OUTSIDE *$P$.*

3. *If $v \in B_i$, but $v \notin B_{i\pm1}$, then $v$* INSIDE *$P$* iff *$\alpha_i = \text{R}$.*

4. *If $v \in B_i \cap B_{i+1}$ and the $i/i+1$ corner is* convex, *then $v$* INSIDE *$P$* iff *$\alpha_i = \text{R}$ and $\alpha_{i+1} = \text{R}$.*

5. *If $v \in B_i \cap B_{i+1}$ and the $i/i+1$ corner is* reflex, *then $v$* INSIDE *$P$* iff *$\alpha_i = \text{R}$ or $\alpha_{i+1} = \text{R}$.*

*Proof* In Figure 4 the cases 1 through 5 are demonstrated by the points $x_1$ through $x_5$. □

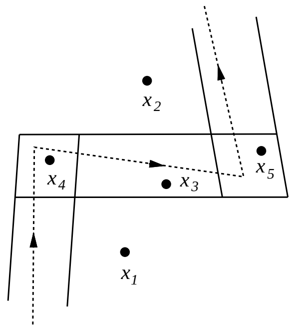


Figure 4: Cases for the Point Location Lemma

Given the Model Existence Lemma and the Point Location Lemma, a point location algorithm can be developed. This algorithm will construct the region $\tilde{C}_\alpha$, pick a point from its interior, and apply the rules of the Point Location Lemma to determine whether the point is INSIDE or OUTSIDE the model in which it has signature $\alpha$. The following Uniqueness Theorem shows that if one such point is INSIDE its model polygon then *all* such points are INSIDE their respective model polygons (similarly for OUTSIDE).

**Theorem 1 (Uniqueness)** *Given an approximate polygon $P_{rep}$ and a signature $\alpha$, if for* some *model polygon in $\text{MODELS}(P_{rep})$ there is a point with signature $\alpha$ which is* INSIDE *the polygon, then, for* every *model polygon, all points which have signature $\alpha$ with respect to that polygon are* INSIDE *that polygon (similarly for* OUTSIDE*).*

*Proof in Appendix A.*

## Point Location Algorithm

The Model Existence Lemma, Point Location Lemma, and Uniqueness Theorem combine to form the point location algorithm shown in Figure 5. Note that the algorithm is quite simple and never actually constructs the model polygon.

**Lemma 3 (Robustness)** *The point location algorithm is robust.*

1. Compute $\tilde{C}_\alpha$.
2. If $\tilde{C}_\alpha = \emptyset$ then no model of $P_{rep}$ induces a cell with signature $\alpha$.
3. Pick a point $w$ on the interior of $\tilde{C}_\alpha$.
4. Apply the Point Location Lemma to determine whether $w$ is INSIDE or OUTSIDE of the models in which it has signature $\alpha$.

Figure 5: Point Location Algorithm

*Proof* This follows directly from the Model Existence Lemma and the Point Location Lemma. □

**Lemma 4 (Consistency)** *The approximate point location problem is consistent.*

*Proof* This follows directly from the Uniqueness Theorem. □

**Lemma 5 (Complexity)** *The point location algorithm has time complexity $\mathcal{O}(n^2)$.*

*Proof* Step 1 of the algorithm can be accomplished by computing the arrangement of half-regions in $\mathcal{O}(n^2)$ time. This is done by computing the arrangement of the $3n$ lines which define the $n$ half-regions $H_i$, then joining adjacent cells which are separated by a line segment which is *not* part of the boundary of some $H_i$. Those cells separated by a line segment which *is* part of the boundary of some $H_i$ will have signatures which differ in a single position, so the signature of each cell can be found in constant time.

The remaining steps of the algorithm take constant time. Step 3 is easily accomplished given the convex decomposition of $\tilde{C}_\alpha$ which is computed simultaneously with $\tilde{C}_\alpha$ itself. Thus, the overall running time is $\mathcal{O}(n^2)$. □

# Summary

Most geometric algorithms are not *robust*; they fail due to inexact input or with inexact intermediate computations. This paper has introduced (a) formal definitions of robustness and consistency, and (b) the notion of an *approximate polygon*, along with several of its properties. With these, one can formally develop robust and consistent algorithms that deal with inexact polygons.

One such algorithm for point location in an approximate polygon has been presented. The algorithm is particularly suited for practical application in a solid modeler because it assumes uncertainty in both the polygon position and the point position. The point location algorithm has been proved robust, and the point location problem has been shown to be consistent.

# Acknowledgments

I am pleased to acknowledge the many helpful discussions which I have had with Chanderjit Bajaj and Paul Chew during the course of this work. Improvements to the algorithm were contributed by Victor Milenkovic, and this line of research was originally suggested by John Hopcroft.

My research has been supported by Nasa Training Grant NGT-50327. This work is also supported by Office of Naval Research Contract N00014-88K-0591, Office of Naval Research Grant N00014-89J-1946, and National Science Foundation Grant IRI-9006137.

# Appendix A

**Theorem 1 (Uniqueness)**
*Given an approximate polygon $P_{rep}$ and a signature $\alpha$, if for* some *model polygon in* MODELS$(P_{rep})$ *there is a point with signature $\alpha$ which is* INSIDE *the polygon, then, for* every *model polygon, all points which have signature $\alpha$ with respect to that polygon are* INSIDE *that polygon (similarly for* OUTSIDE*).*

*Proof (by contradiction):* Let $\alpha^k(x)$ be the signature of point $x$ in model $P^k$. Let $e_i^k$ be edge $e_i$ in model $P^k$. Then assume the following:

$$\exists P^1, P^2 \in \text{MODELS}(P_{rep}), \;\; \exists u, v \in \Re^2,$$

$$u \text{ INSIDE } P^1 \;\;\wedge\;\; v \text{ OUTSIDE } P^2 \;\;\wedge\;\; \alpha^1(u) = \alpha^2(v).$$

The theorem is proved by showing that there is some edge $e_k$ which always separates $u$ and $v$, violating this assumption.

**Lemma 6** *One of $u$ or $v$ must lie within the boundary of $P_{rep}$.*

*Proof* Assume that neither $u$ nor $v$ is within the boundary. Then by the initial assumption and the Point Location Lemma $u$ and $v$ lie on opposite sides of the boundary. Say $u$ is inside and $v$ is outside. Then the segment $\overline{uv}$ must traverse both of the parallel sides of some band $B_i$, as shown in Figure 6. From the definition of the corners of $B_i$ and the definition of SPAN$(B_i)$, $u$ and $v$ lie to different sides of SPAN$(B_i)$. Then by Property 6 $u$ lies to one side of all models of $P_{rep}$ and $v$ lies to the other side. Thus, in the models $P^1$ and $P^2$, $\alpha_i^1(u) \neq \alpha_i^2(v)$ and the initial assumption is contradicted. $\square$

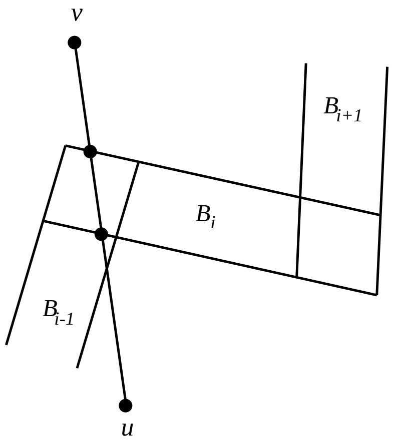


Figure 6: $\overline{uv}$ crosses some $B_i$

From Lemma 6 assume without loss of generality that $u$ lies in the boundary of $P_{rep}$.

**Lemma 7** *The point $u$ lies in some band $B_i$ such that $\alpha_i^1(u) = \mathrm{R}$ and $\alpha_i^2(v) = \mathrm{R}$.*

*Proof* $u$ is INSIDE $P^1$ by the initial assumption and is in some band by Lemma 6. If $u \in B_k$ and $u \notin B_{k\pm1}$, then by the Point Location Lemma $u$ is to the RIGHT of $e_k^1$. Choose $i = k$. If $u \in B_k \cap B_{k+1}$ then by the Point Location Lemma $u$ is to the right of at least one of $e_k^1$ or $e_{k+1}^1$. Choose $i$ to be $k$ or $k+1$ to satisfy the lemma. Then by the initial assumption, $\alpha_i^1(u) = \mathrm{R}$ means that $\alpha_i^2(v) = \mathrm{R}$ also. $\square$

**Lemma 8** *On the segment $\overline{uv}$ there is some point $x \neq v$ which is inside $P^2$. Furthermore, $e_i^2$ does* not *intersect $\overline{uv}$ between $x$ and $v$.*

*Proof*

*Case 1: $u \in B_i$, but $u \notin B_{i\pm1}$.* If $\alpha_i^2(u) = \mathrm{R}$, then $u$ INSIDE $P^2$ (by the Point Location Lemma), so choose $x = u$ and we are done. Otherwise consider the case in which $\alpha_i^2(u) \neq \mathrm{R}$.

Refer to Figure 7. The point $u$ lies in $B_i$ and not in $B_{i-1}$, so by Property 7, in all models $P^k$, $u$ is to the same side of $e_{i-1}^k$. In particular, $\alpha_{i-1}^1(u) = \alpha_{i-1}^2(u)$. By the initial assumption, $\alpha_{i-1}^1(u) = \alpha_{i-1}^2(v)$, so $\alpha_{i-1}^2(u) = \alpha_{i-1}^2(v)$. By a similar argument for $e_{i+1}$,

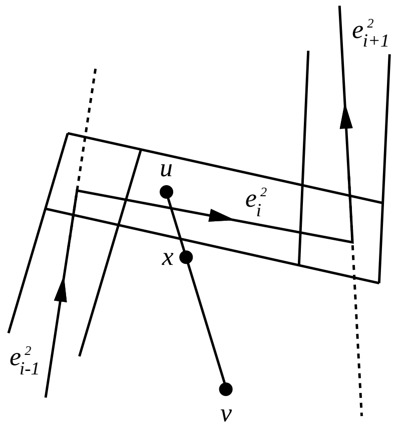


Figure 7: $x$ exists if $u \in B_i$

$\alpha_{i+1}^2(u) = \alpha_{i+1}^2(v)$. Thus $u$ and $v$ lie between two lines touching the endpoints of $e_i^2$.

Since $\alpha_i^2(u) \neq \mathrm{R}$ and since by Lemma 7 $\alpha_i^2(v) = \mathrm{R}$, $\overline{uv}$ must cross the *line* defined by $e_i^2$. But $u$ and $v$ lie between the two lines touching the endpoints of $e_i^2$, so $\overline{uv}$ must cross the *edge* $e_i^2$. Since by Property 4 all models $P$ are simple, there is a small neighborhood to the right of $e_i^2$ which contains only points interior to $P$. The segment $\overline{uv}$ passes through this neighborhood, so there is some point $x \neq v$ on $\overline{uv}$ which is INSIDE $P^2$, and $\overline{xv} \cap e_i^2 = \emptyset$.

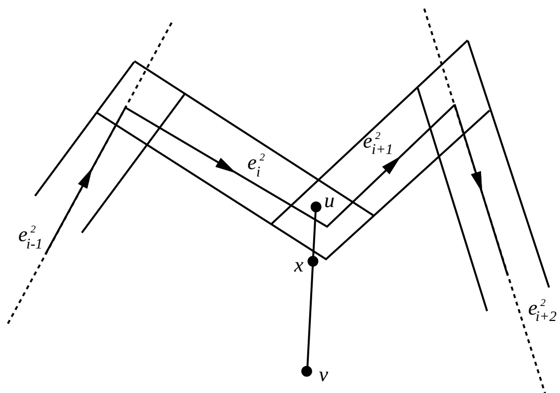


Figure 8: $x$ exists if $u \in B_i \cap B_{i+1}$

*Case 2: $u \in B_i \cap B_{i+1}$.* Refer to Figure 8. By an argument similar to Case 1, $u$ and $v$ must lie to the same side of $e_{i-1}^k$ in all models $P^k$, and must lie to the same side of $e_{i+2}^k$ in all models $P^k$. If $u$ INSIDE $P^2$ then choose $x = u$ and we are done. Otherwise, if $u$

OUTSIDE $P^2$ then the edges $e_i^2$ and $e_{i+1}^2$ must separate $u$ and $v$. Thus the segment $\overline{uv}$ must cross either $e_i^2$ or $e_{i+1}^2$, and in doing so must pass through a neighborhood of interior points to the right of the edge that it crosses. Therefore there is some point $x \neq v$ on $\overline{uv}$ which is INSIDE $P^2$, and $\overline{xv} \cap e_i^2 = \emptyset$. □

**Lemma 9** *In $P^2$ there is some edge $e_j^2$ which crosses $\overline{uv}$ such that $\alpha_j^2(u) = \text{R}$ and $\alpha_j^2(v) = \text{L}$. Furthermore, $e_j^2$ can be chosen such that no other $e_k^2$ crosses $\overline{uv}$ between $e_j^2$ and $v$.*

*Proof* From lemma 8, $\overline{uv}$ contains a point $x$ which is INSIDE $P^2$, and by the initial assumption, $v$ OUTSIDE $P^2$. From the Jordan curve theorem we know that $\overline{xv}$ must cross the boundary of $P^2$ on some edge $e_j^2$ with $x$ to the inside (right) of $e_j^2$ and $v$ to the outside (left) of $e_j^2$. From the ordering of points along $\overline{uv}$ ($u \leq x < v$), if $\alpha_j^2(x) = \text{R}$ then $\alpha_j^2(u) = \text{R}$ also. If there are many candidates for $e_j^2$, choose that which is closest to $v$ to satisfy the second part of the lemma. □

**Lemma 10** $B_j \cap B_i = \emptyset$

*Proof* If $|i-j| > 1$ then by the definition of an approximate polygon $B_i \cap B_j = \emptyset$. So we only have to consider $|i-j| \leq 1$. But by Lemma 9 $e_j^2$ intersects $\overline{uv}$ between $x$ and $v$, and by Lemma 8 $e_i^2$ does *not* intersect $\overline{uv}$ between $x$ and $v$. So $e_i^2 \neq e_j^2$ and hence $i \neq j$.

Assume that $j = i-1$. Then, by the initial assumption, $\alpha_j^2(v) = \text{L}$ means that $\alpha_j^1(u) = \text{L}$. By Lemma 9, $\alpha_j^2(u) = \text{R}$. Since $u$ is to different sides of $e_j$ in $P^1$ and $P^2$, $u$ must lie in $\text{SPAN}(B_j)$. Since $u$ also lies in $B_i$, by Property 5 $u$ lies in the corner $B_i \cap B_j$.

Suppose corner $i/j$ is convex. Since $u$ INSIDE $P^1$, by the Point Location Lemma $\alpha_i^1(u) = \text{R}$ and $\alpha_j^1(u) = \text{R}$. But, by the initial assumption, $\alpha_j^1(u) = \text{R}$ means that $\alpha_j^2(v) = \text{R}$, contradicting lemma 9. So corner $i/j$ is not convex.

Suppose corner $i/j$ is reflex. Refer to Figure 9. By Lemma 9, $\alpha_j^2(u) = \text{R}$ and $\alpha_j^2(v) = \text{L}$, and by Lemma 7, $\alpha_i^2(v) = \text{R}$. But if $\overline{uv}$ is to intersect $e_j^2$ then it must also intersect $e_i^2$ closer to $v$, as shown in the Figure. This contradicts Lemma 9, which states the $e_j^2$ is the closest intersection to $v$. So corner $i/j$ is not reflex.

Thus the assumption is false that $j \neq i-1$. By a similar argument $j \neq i+1$. Therefore $B_j \cap B_i = \emptyset$. □

**Lemma 11** $u \in \text{SPAN}(B_j) - B_j$

*Proof:* By lemma 9, $\alpha_j^2(u) = \text{R}$ and $\alpha_j^2(v) = \text{L}$. By the initial assumption, $\alpha_j^2(v) = \text{L}$ means that $\alpha_j^1(u) = \text{L}$. Since $\alpha_j(u)$ is different in models $P^1$ and $P^2$, $u$ must lie in $\text{SPAN}(B_j)$. From lemma 10, $u$ cannot lie in $B_j$ since it already lies in $B_i$. □

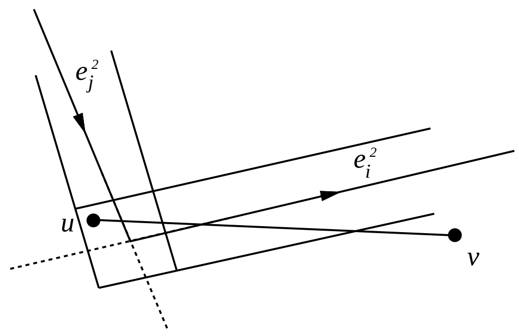


Figure 9: $\overline{uv}$ must intersect $e_i^2$ closer to $v$

**Lemma 12** *Define $k$ such that $B_j \cap B_k$ is closest to $u$. Then $u$ and $v$ are on opposite sides of $e_k^2$.*

*Proof* Refer to Figure 10. Note that $k = j \pm 1$, otherwise $B_j$ and $B_k$ wouldn't intersect at all. In any model $P$ the point $u$ and the edge $e_j$ are on opposite sides of the line defined by $e_k$ ($u \in \text{SPAN}(B_j) - B_j$ by lemma 11, and since $B_k$ is closest to $u$, it separates $u$ from the rest of $B_j$, which contains $e_j$). Thus $\overline{uv}$ must intersect $e_j^2$ at some point $z$ to the side of $e_k^2$ which is opposite to $u$. From the ordering of points along $\overline{uv}$ (($u < z < v$), $v$ must also be opposite to $u$. □

**Lemma 13** $\alpha_k^1(u) = \alpha_k^2(u)$.

*Proof* By Lemma 12, $u \in \text{SPAN}(B_j) - B_j$. By Property 5, $\text{SPAN}(B_j) \cap \text{SPAN}(B_k) = B_k \cap B_k$, so with a bit of algebra we can conclude that $u \notin \text{SPAN}(B_k)$. Then by Property 6, $\alpha_k^1(u) = \alpha_k^2(u)$. □

By lemma 12 there is some edge $e_k^2$ in model $P^2$ such that $\alpha_k^2(u) \neq \alpha_k^2(v)$, and by lemma 13, $\alpha_k^1(u) = \alpha_k^2(u)$. So $\alpha_k^1(u) \neq \alpha_k^2(v)$. But this contradicts the initial assumption, so the theorem is proved by contradiction.

□

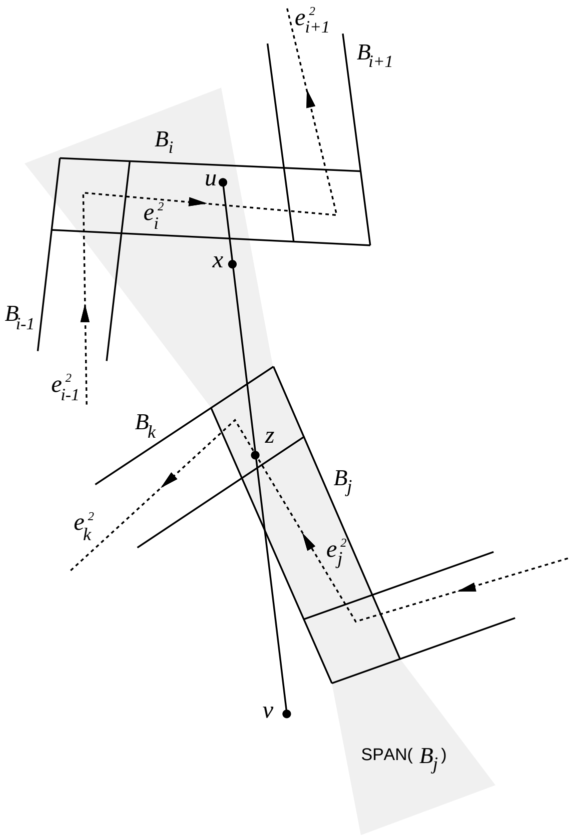


Figure 10: $u$ and $v$ are on opposite sides of $e_k^2$

# References


[1] D. Dobkin and D. Silver. Recipes for geometry & numerical analysis - Part I: An empirical study. In *Annual Symposium on Computational Geometry*, pages 93–105, 1988.

[2] S. Fortune. Stable maintenance of point set triangulations in two dimensions. In *IEEE Annual Symposium on Foundations of Computer Science*, pages 494–499, 1989.

[3] D. H. Greene and F. F. Yao. Finite-resolution computational geometry. In *IEEE Annual Symposium on Foundations of Computer Science*, pages 143–152, October 1986.

[4] C. M. Hoffmann, J. E. Hopcroft, and M. S. Karasick. Towards implementing robust geometric computations. In *Annual Symposium on Computational Geometry*, pages 106–117, June 1988.

[5] M. Karasick. *On the Representation and Manipulations of Rigid Solids*. PhD thesis, McGill University, 1989.

[6] V. J. Milenkovic. *Verifiable Implementations of Geometric Algorithms Using Finite Precision Arithmetic*. PhD thesis, Carnegie Mellon University, 1988.

[7] V. J. Milenkovic. Double precision geometry: A general technique for calculating line and segment intersections using rounded arithmetic. In *IEEE Annual Symposium on Foundations of Computer Science*, pages 500–505, 1989.

[8] D. Salesin, J. Stolfi, and L. Guibas. Epsilon geometry: Building robust algorithms from imprecise calculations. In *Annual Symposium on Computational Geometry*, pages 208–217, 1989.

[9] M. Segal and C. Sequin. Partitioning polyhedral objects into nonintersecting parts. In *IEEE Computer Graphics and Applications*, pages 53–67, January 1988.

[10] K. Sugihara and M. Iri. Construction of the Voronoi diagram for over $10^5$ generators in single-precision arithmetic. In *Abstracts of the First Canadian Conference on Computational Geometry*, page 42, August 1989.

[11] K. Sugihara and M. Iri. Two design principles of geometric algorithms in finite-precision arithmetic. *Appl. Math. Lett.*, 2(2):203–206, 1989.

[12] K. Sugihara, Y. Ooishi, and T. Imai. Topology-oriented approach to robustness and its applications to several Voronoi-diagram algorithms. In *Canadian Conference on Computational Geometry*, pages 36–39, August 1990.