A Robust Algorithm for Point in Polyhedron

C. BRADFORD BARBER*
The Geometry Center
University of Minnesota
Minneapolis, MN 55454
barber@geom.umn.edu

MICHAEL D. HIRSCH†
Department of Computer Science
Princeton University
Princeton, NJ 08544
mhirsch@princeton.edu

June 24, 1993

Abstract

A polyhedron consists of faces, their incidences, and an assignment of faces to point sets in $\mathbf{R}^{d+1}$. A polyhedron separates $\mathbf{R}^{d+1}$ into an unbounded and bounded component. The *point inclusion* test for polyhedra reports whether a point belongs to the bounded component.

We present an algorithm for point inclusion in general dimension polyhedron. Ours is the first combinatorial algorithm without special cases for singular inputs. In addition, we allow imprecisely specified faces and round-off error. This is the first point-in-polyhedron algorithm to work in general dimension with arbitrarily imprecise data and arithmetic.

We represent imprecision by assigning an open, convex set to each face. The open sets are called *boxes* and represent the possible locations of a feature. We then prove that point inclusion in a precise or an imprecise polyhedron can be reduced to the odd/even parity of a vertex subset.

1. Introduction

Polyhedra are geometric objects commonly used to model real-world data on a computer. Point inclusion for polyhedra is a basic geometric operation for point membership classification. In constructive solid geometry, objects are composed by set operations on primitive and previously constructed objects. Point inclusion is the first step in adding an object to a representation. Point classification is also important for clipping, geometric intersection, and ray tracing [13] [5].

Typically, one tests for point inclusion by shooting a ray from the point to infinity. If the ray crosses an odd number of facets, the point is inside the polyhedron. This *crossing parity test* is a corollary to the generalized Jordan-Brouwer Separation Theorem [12]. As often happens with geometric algorithms, there are a number of special cases to worry about. For example, the ray could intersect an edge or vertex, or an edge could be a subset of the ray. These are called *singularities* and are measure zero events. In other words, such singularities almost never occur (in theory). Unfortunately, on a discrete digital computer "almost never" can be "alarmingly often."

We handle singularities and/or imprecision by reducing point inclusion to the odd/even parity of a vertex subset. Each step of the reduction identifies a set of facets that *might* intersect the test ray. By allowing such uncertainty, we can treat singular cases as if they were non-singular. The boundary of this set does *not* intersect the test ray. We project the boundary to a perpendicular hyperplane and perform point inclusion in one lower dimension. We call it the `Flashlight` algorithm because, with imprecision, a test ray is like a flashlight's beam that illuminates many facets, some of which, perhaps, would be missed by a single ray. This is the first point inclusion algorithm to work in general dimension for arbitrarily imprecise data and arithmetic.

*This research was supported in part by the National Science Foundation under Grant Numbers CCR90-02352 and NSF/DMS-8920161.

†Supported in part by an NSF Postdoctoral Fellowship.

Other solutions have been published for point inclusion. In 3-d, Kalay merges faces when a singularity is detected [8], and Horn and Taylor tests the orientation of nearby features [7]. In general dimension, Corkum and Wyllie randomly rotate the polyhedron [2]. Lane, Magedson and Rarick project each facet to a sphere and sum the projected areas [10]. If the arithmetic is sufficiently imprecise or the input sufficiently large, neither of the general dimension algorithms will determine an answer.

Besides solving a geometric problem, `Flashlight` demonstrates a practical approach to handling imprecision. We model imprecision by assigning a convex set, called a *box*, to each geometric feature. A box constrains the possible locations of the feature and the maximum effect of roundoff error. A box is the geometric equivalent of an interval in interval arithmetic [11]. Once an algorithm is designed for boxes, imprecision and roundoff error are simply parameters of the implementation. Imprecision and roundoff errors are active topics in computational geometry. Two other models are *ϵ-geometry* [4] and the *reasoning paradigm* [6]. See [3] for a discussion.

Section 2 reviews the mathematics we will use. Section 3 presents the `Flashlight` algorithm for precisely specified polyhedra. Section 4 presents the `Flashlight` algorithm for imprecise polyhedra. Section 5 proves the correctness of `Flashlight`. For missing proofs, see [1].

2. Definitions and Notation

For simplicity of presentation, all our polyhedra will be simplicial complexes. Our point inclusion algorithm, `Flashlight`, would work equally well with any reasonable definition of polyhedron in which the faces are convex [1].

DEFINITION 2.1. *A d-dimensional polyhedron is a simplicial d-complex in $\mathbf{R}^{d+1}$ satisfying conditions of* twoness *and* minimality *on its d-faces (henceforth called* facets*). The* twoness *condition is that each $(d-1)$-face (called a* ridge*) is a face of exactly two facets. The* minimality *condition is that no non-empty proper subset of the facets satisfies the twoness condition.*

Like the other algorithms, we use crossing parity to determine if a point is inside a polyhedron. Our algorithm is recursive. The recursive step does not preserve the twoness and minimality conditions of a polyhedron nor the intersection properties of a simplicial complex. For this reason we generalize the polyhedron to require an evenness condition and to allow overlapping facets. Given the following definition, the proof of Theorem 2.4 is straight-forward.

DEFINITION 2.2. *A d-dimensional generalized polyhedron in $\mathbf{R}^n$ is the image of a simplicial d-complex K under a map f to $\mathbf{R}^n$. The simplicial complex K satisfies the* evenness *condition, i.e. each ridge occurs in an even number of facets. The map f must be continuous on $\mathbf{R}^n$ and an affine map on each facet.*

LEMMA 2.3. *Let $\mathcal{P}$ be a generalized polyhedron of dimension d in $\mathbf{R}^n$. Let S be a subset of facets of $\mathcal{P}$. Then ∂S is a generalized polyhedron of dimension $d-1$ in $\mathbf{R}^n$.*

PROOF: Suppose that there were some ridge r of ∂S that was a face of an odd number of facets of ∂S. Then r is in $\partial\partial S$. But, for any simplicial complex, $\partial\partial S = \emptyset$ [12]. ∎

THEOREM 2.4. *Given a generalized polyhedron $\mathcal{P}$ of dimension d and a point $q \in \mathbf{R}^{d+1} - \mathcal{P}$, the crossing parity for q is well defined i.e., the crossing parity of a curve from q to the unbounded component is independent of the curve.*

3. The Flashlight algorithm for precise polyhedra

Let $\mathcal{P}$ be a polyhedron in $\mathbf{R}^{d+1}$ and q be a point in $\mathbf{R}^{d+1} - \mathcal{P}$. The function `Flashlight`$(\mathcal{P}, q, d)$ classifies q as *inside*, *outside* or *on* $\mathcal{P}$. `Flashlight` is a recursive algorithm that reduces the problem's dimension.

Each call to `Flashlight` selects a subset of the facets called the "front half". We say a facet is *in front* of q if it intersects the test ray and *behind* q if it intersects the opposite test ray. The *front half*, $\mathcal{P}_+$, is the facets that are *in front* of q. `Flashlight` reduces the problem's dimension by projecting $\partial\mathcal{P}_+$ to a perpendicular hyperplane for the test ray. See Figure 3.1 for the algorithm and Figure 3.2 for an example. (The words in brackets in Figure 3.1 should be ignored; they are used in the algorithm for the imprecise case.)

```
Algorithm for Flashlight(𝒫, q, d)
    let 𝒫+ be a set of facets (the front half), initially empty
    let H be a hyperplane perpendicular to the dth test ray
    for each facet f of 𝒫
        if f is [maybe] in front of q then
            if f is [maybe] behind q, return [maybe] on
            else append f to 𝒫+
    if d > 0
        return Flashlight(π_H ∂𝒫+, π_H q, d-1)
    else
        if |𝒫+| mod 2 = 1, return [clearly] inside
        else return [clearly] outside
```

Figure 3.1: `Flashlight` tests if point q is inside polyhedron $\mathcal{P}$ in $\mathbf{R}^{d+1}$. [Words in brackets are used in the imprecise case.]

THEOREM 3.5. *Let $\mathcal{P}$ be a polyhedron and let H be a hyperplane perpendicular to the test ray from a point q. The crossing parity of q relative to $\mathcal{P}$ in $\mathbf{R}^{d+1}$ is the same as the crossing parity of $\pi_H q$ relative to $\pi_H \partial\mathcal{P}_+$ in H.*

The proof of Theorem 3.5 uses the following construction. Pull apart the front half and its complement while stretching the boundary. We can do this without changing the inside/outside classification of the point. Eventually a perpendicular hyperplane separates the two parts. A test ray in this hyperplane has the same crossing parity as the original test ray. So intersecting this hyperplane with the stretched boundary gives us a similar classification problem one dimension lower. In an implementation, the stretching and intersecting is accomplished by projecting $\partial\mathcal{P}_+$ to the hyperplane.

THEOREM 3.6. (CORRECTNESS FOR POLYHEDRA) *Let $\mathcal{P}$ be a polyhedron and q be a point in $\mathbf{R}^{d+1} - \mathcal{P}$.* `Flashlight` *computes the crossing parity of q.*

PROOF: `Flashlight` projects $\partial\mathcal{P}_+$ to H and recurses on the new generalized polyhedron. Theorem 3.5 guarantees that this reduced problem has the same answer as the original. The recursion terminates when $d = 0$ and $\mathcal{P}$ is just an even number of points. A ray either crosses, or misses a point in $\mathbf{R}$—there are no singularities. Thus when $d = 0$, the crossing parity of q is the size parity of $\mathcal{P}_+$. ∎

Notice that the algorithm and its proof of correctness used very little geometric information about the location of a facet and its faces—only whether a facet can be translated relative to the test point. The proof is valid so long as the $\mathcal{P}_+$ contains all facets that are in front of q and none that are behind q. This flexibility makes `Flashlight` ideally suited for imprecise data and arithmetic.

4. The `Flashlight` algorithm for imprecise polyhedra

In the previous section we assumed that the polyhedron was precisely specified, and that the computer could do all computations exactly, as did the combinatorial algorithms mentioned in Section 1. Unfortunately, this is not, in general, the case.

DEFINITION 4.7. *An* imprecise (generalized) polyhedron *$\mathcal{P}$ is a (generalized) polyhedron whose faces are each known to be inside some open, convex set called the face's* box. *The exact location of a face is unspecified. The box for higher-dimensional faces contains the convex hull of the boxes of its sub-faces. The trace of $\mathcal{P}$,* trace($\mathcal{P}$), *is the union of the boxes of $\mathcal{P}$.*

Given an imprecise polyhedron, we can determine some facts with certainty, but not others. For instance we can determine that the test ray does *not* cross a facet when the test ray misses the facet's box entirely. But if the test ray goes through a ridge's box, it may or may not cross a facet. This incomplete information is enough, however, to perform the reduction step of `Flashlight`.

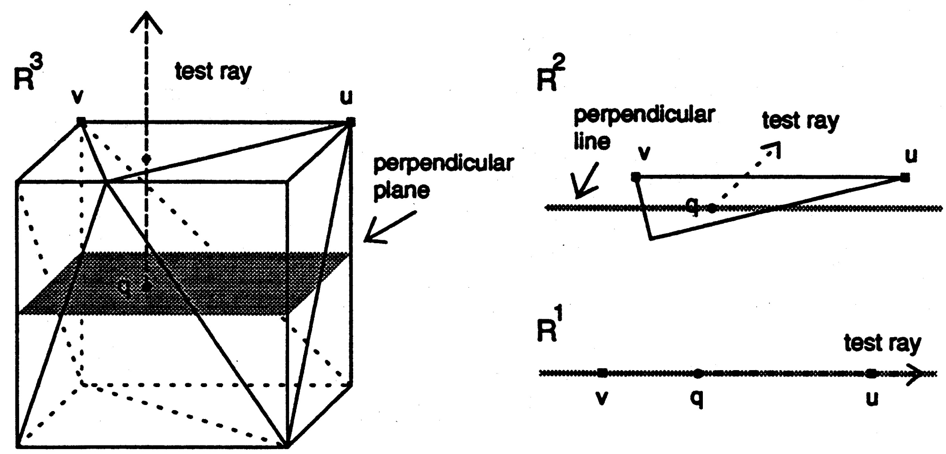

Figure 3.2: **Flashlight** reduces point inclusion to a subset of the vertices. In this case, the subset contains one vertex, u, so the point q is inside the cube.

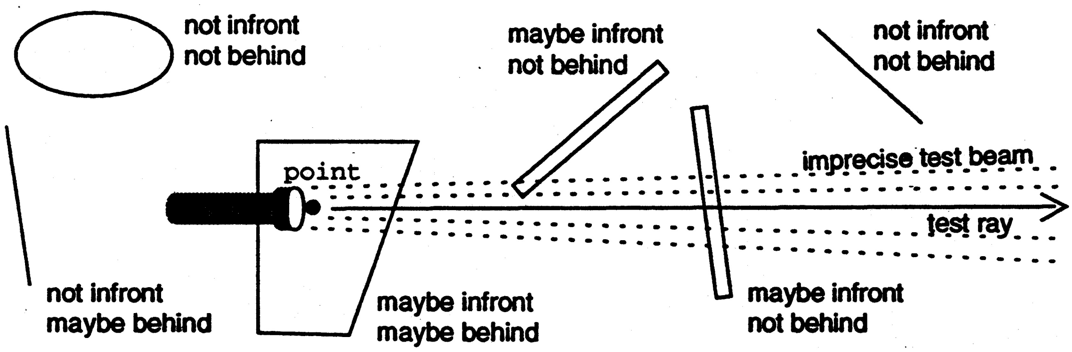

Figure 4.3: Examples of how the **Flashlight** classifies imprecise geometric objects.

We call the relationship of a test ray *not* intersecting a facet, *not infront*. We can test for *not infront* with certainty. The negation of *not infront* is *maybe infront*. Some instances of *maybe infront* will intersect the facet while others will miss the facet. For the opposite test ray, the corresponding relations are *not behind* and *maybe behind* (See Figure 4.3). **Flashlight** for imprecise polyhedra differs from the precise version only in the qualifiers "*maybe*" and "*clearly*" for the return values of the tests and **Flashlight** itself. See Figure 3.1.

The complexity of **Flashlight** is easily upper bounded by noting that **Flashlight** examines each face at most once. If facet crossing is tested in constant time, **Flashlight** is linear in the number of faces. For n vertices, the maximum number of faces for a d-polyhedron is $O(n^{\lfloor d/2 \rfloor})$ [9]. In practice, of course, the boundary of the front half is a small subset of the input, and this is a generous over-estimate.

5. Correctness proof for Flashlight

The proof of **Flashlight** under imprecision is very similar to the proof with precise data and arithmetic. We simply substitute boxes wherever a facet's simplex appears. We can still perform the translations of the front half and its complement. So if crossing parity is well-defined, **Flashlight** determines point inclusion for imprecise polyhedra. In this section, we sketch the proofs that crossing parity is well-defined and that **Flashlight** is correct. These proofs are considerably more difficult if we allow non-simplicial polyhedra [1]. To define the

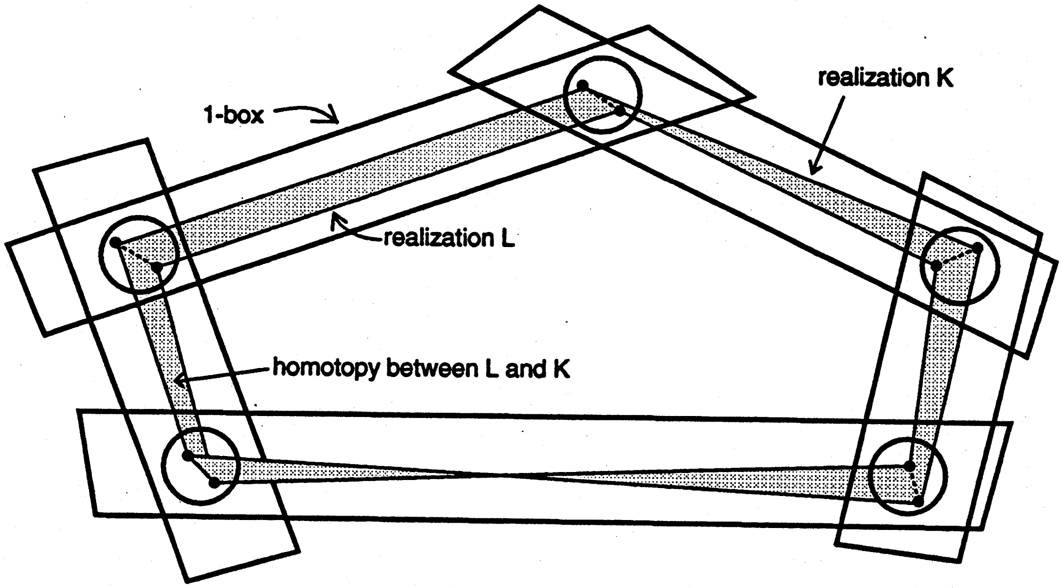

Figure 5.4: Multiple realizations of an imprecise polyhedron are homotopic. In this case, each realization is a pentagon, and the homotopy is space between them.

crossing parity of an arc and imprecise polyhedron, we introduce the concept of a *realization*.

DEFINITION 5.8. *A* realization *of an imprecise (generalized) polyhedron $\mathcal{P}$ is a (generalized) polyhedron K with the same subface relations as $\mathcal{P}$. The vertices of K are contained in the corresponding boxes of $\mathcal{P}$.*

Since the box of a face is convex, it includes the corresponding face of the realization. By Theorem 2.4, crossing parity is well defined for a realization. We need to show that all realizations yield the same crossing parity for a point. To do this, note that all realizations are homotopic. This is essentially an immediate consequence of the convexity of the boxes (see Figure 5.4 in lieu of proof).

Recall Theorem 2.4 which says that the crossing parity between two points and a polyhedron is independent of the path. We now show it is also independent of the realization.

THEOREM 5.9. *Let K and L be realizations of a d-dimensional imprecise, generalized polyhedron $\mathcal{P}$ in $\mathbf{R}^{d+1}$, and let c be an arc between points $q_1, q_2 \in \mathbf{R}^{d+1} - \mathrm{trace}(\mathcal{P})$ in general position with respect to K and L. Then the crossing parity of c and K is equal to the crossing parity of c and L.*

PROOF: Let H be the image of the homotopy from K to L and consider $c \cap H$. This is a union of disjoint intervals of c. The endpoints of the intervals are the points of $c \cap K$ and $c \cap L$. There are an even number of endpoints, so the the parities of $c \cap K$ and $c \cap L$ must be equal. ∎

DEFINITION 5.10. *Given an imprecise, generalized polyhedron $\mathcal{P}$ and a point $q \in \mathbf{R}^{d+1} - \mathrm{trace}(\mathcal{P})$, the* crossing parity *for q relative to $\mathcal{P}$ is the crossing parity of q relative to any realization K of $\mathcal{P}$.*

The following theorem is immediate from Theorems 2.4 and 5.9

THEOREM 5.11. *Given an imprecise, generalized polyhedra $\mathcal{P}$ and a point $q \in \mathbf{R}^{d+1} - \mathrm{trace}(\mathcal{P})$, the crossing parity for q is well defined.*

THEOREM 5.12. (CORRECTNESS OF `Flashlight`) *Let $\mathcal{P}$ be an imprecise polyhedron and q be a point in* $\mathbf{R}^{d+1} - \text{trace}(\mathcal{P})$. `Flashlight` *computes the crossing parity of* q.

PROOF: `Flashlight` correctly computes the crossing parity relative to a realization of $\mathcal{P}$ (Theorem 3.6) which equals the crossing parity relative to $\mathcal{P}$ (Theorem 5.11). ∎

References

[1] C. B. Barber and M. D. Hirsch. Point in polyhedron with imprecise data and arithmetic. Technical Report GCG46, The Geometry Center, 1992.

[2] B. T. Corkum and J. A. Wyllie. Program for insidepolytope. News group comp.graphics message, 1990. 1990Oct3.121706.11932@jarvis.csri.toronto.edu.

[3] S. Fortune. Computational geometry. In R. Martin, editor, *Directions in Geometric Computing*. Information Geometers, 1992. in preparation.

[4] L. Guibas, D. Salesin, and J. Stolfi. Epsilon geometry: Building robust algorithms from imprecise computations. In *5th Annual Symposium on Computational Geometry*, pages 208–217. ACM, 1989.

[5] P. Hanrahan. A survey of ray-surface intersection algorithms. In A. Glassner, editor, *An introduction to ray tracing*. Academic Press, 1989.

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

[7] W. P. Horn and D. L. Taylor. A theorem to determine the spatial containment of a point in planar polyhedron. *Computer Vision, Graphics, and Image Processing*, 45:106–116, 1989.

[8] Y. E. Kalay. Determining the spatial containment of a point in general polyhedra. *Computer Graphics and Image Processing*, 19:303–33, 1982.

[9] V. Klee. Convex polytopes and linear programming. In *Proc. IBM Sci. Comput. Symp.: Combinatorial Problems*, pages 123–158. IBM, 1966.

[10] J. Lane, B. Magedson, and M. Rarick. An efficient point in polyhedron algorithm. *Computer Vision, Graphics, and Image Processing*, 26:118–125, 1984.

[11] R. E. Moore. *Methods and Applications of Interval Analysis*. SIAM, 1979.

[12] J. R. Munkres. *Elements of Algebraic Topology*. Addison-Wesley, 1984.

[13] R.B. Tilove. Set membership classification: A unified approach to geometric intersection problems. In *IEEE Trans. on Computers C-29*, pages 874–883, 1980.