Stabbing Oriented Convex Polygons in Randomized $O(n^2)$ Time

Seth J. Teller
Michael E. Hohmeyer
University of California at Berkeley[‡]

Abstract

We present an algorithm that determines, in expected $O(n^2)$ time, whether a line exists that stabs each of a set of *oriented* convex polygons in R^3 with a total of n edges. If a stabbing line exists, the algorithm computes at least one such line. We show that the computation amounts to constructing a convex polytope in R^5 and inspecting its edges for intersections with a four-dimensional quadric surface, the Plücker quadric.

CR Categories and Subject Descriptors: [Computer Graphics]: I.3.5 Computational Geometry and Object Modeling – *geometric algorithms, languages, and systems.*

Additional Key Words and Phrases: Linear programming, stabbing lines, high-dimensional convex hulls, polygon traversals, Plücker coordinates, sightline visibility.

[‡]Computer Science Department, Berkeley, CA 94720

1 Introduction

Consider a collection of *oriented* convex polygons; that is, directed planar contours in R^3. Suppose one wishes to determine whether any line simultaneously intersects every polygon, while traversing the plane of each polygon in a specified direction (Figure 1). We show how to compute whether such a *stabbing line* exists for a given set of polygons, and if so, how to compute an example stabbing line. This problem is of practical importance in some visibility computations. For example, a polygonal scene in R^3 might be partitioned into convex cells, interconnected via *portals*, or translucent holes on the shared boundary between two cells. A stabbing line through a sequence of portals could serve as a witness sightline between two non-adjacent polyhedral cells [14]. The polygonal portals between each cell, in this case, would be oriented by the direction in which each portal is encountered along the sequence (for example, during a search of the subdivision adjacency graph).

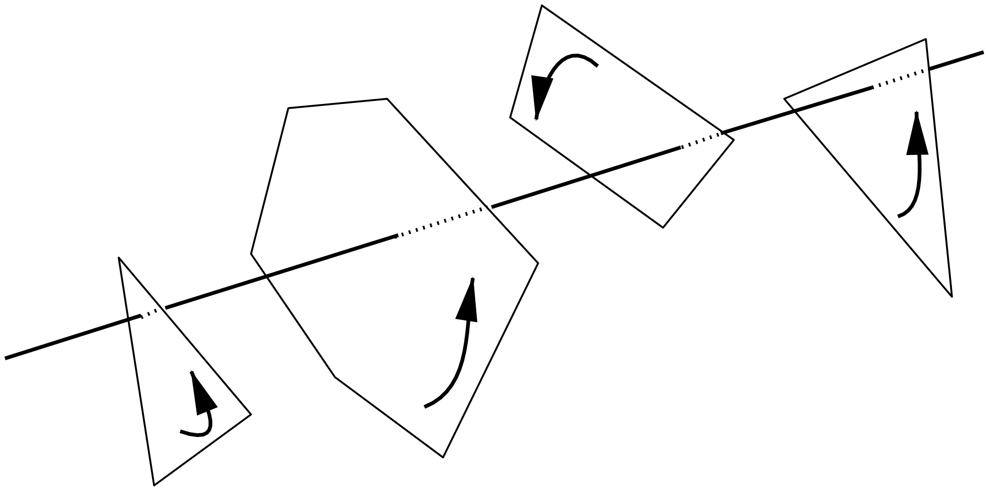

Figure 1: A stabbing line through a collection of polygons in R^3.

For a given set of polygons, let n be the total number of edges comprising the set. Various stabbing line algorithms for unoriented polygons have been formulated. Avis and Wenger presented an $O(n^4 \lg n)$ time algorithm to compute stabbing lines [2]. McKenna and O'Rourke improved this to $O(n^4\alpha(n))$ time [6], where $\alpha(n)$ is the functional inverse of Ackermann's function. If the polygons are triangles, and together comprise g distinct normals, an algorithm due to Pellegrini computes a stabbing line in $O(g^2n^2 \lg n)$ time if one exists [8]. When g is $O(n)$, this time bound is the same as that due to Avis and Wenger.

For the case of input polygons consisting only of *isothetic*, or axis-aligned, rectangles and boxes, Hohmeyer and Teller presented an $O(n \lg n)$ time stabbing line algorithm [5]. Amenta improved this with a randomized linear time algorithm [1]. Finally, Megiddo reduced the problem to linear programming, yielding a deterministic linear time algorithm [7].

Here, we consider the case in which the input polygons are in general, non-axial, convex, and oriented. We orient each polygon by selecting one of its two (antiparallel) plane normals. (Oriented polygons arise often in real applications, e.g. from topological constraints or, as described above, during combinatorial search algorithms.) We then search for (directed) stabbing lines that have a positive inner product with each polygon normal. Knowing the direction in which any stabbing line must traverse each of the polygons allows the formulation of a randomized $O(n^2)$ expected time algorithm. We have implemented this algorithm, using a modified version of d-dimensional Delaunay simplicialization code supplied by Allan Wilks and Allen McIntosh [15].

We use the Plücker coordinatization of lines [12]. Directed lines in R^3 are mapped into both points and hyperplanes in five-dimensional projective space. We show that finding a solution to the stabbing line problem is equivalent to finding a point on the intersection of a polytope and a quadric surface (the Plücker quadric) in R^5. The complexity of a polytope, described by its face graph, in R^d is $O(n^{\lfloor \frac{d}{2} \rfloor})$ [4]. Thus the worst-case complexity of a polytope in R^5 will be $O(n^2)$.

2 Plücker Coordinates

We use the Plücker coordinatization [12] of directed lines in three space. Any ordered pair of distinct points $p = (p_x, p_y, p_z)$ and $q = (q_x, q_y, q_z)$ defines a directed line ℓ in R^3. This line corresponds to a projective six-tuple $\Pi_\ell = (\pi_{\ell 0}, \pi_{\ell 1}, \pi_{\ell 2}, \pi_{\ell 3}, \pi_{\ell 4}, \pi_{\ell 5})$, each component of which is the determinant of a 2×2 minor of the matrix

$$\begin{pmatrix} p_x & p_y & p_z & 1 \\ q_x & q_y & q_z & 1 \end{pmatrix} \tag{1}$$

There are several conventions dictating the correspondence between the minors of (1) and the π_i. We define the $\pi_{\ell i}$ as:

$$\begin{aligned}
\pi_{l0} &= p_x q_y - q_x p_y \\
\pi_{l1} &= p_x q_z - q_x p_z \\
\pi_{l2} &= p_x \quad - q_x \\
\pi_{l3} &= p_y q_z - q_y p_z \\
\pi_{l4} &= p_z \quad - q_z \\
\pi_{l5} &= q_y \quad - p_y
\end{aligned}$$

(this somewhat asymmetric order was adopted in [9] to produce positive signs in some identities about Plücker coordinates).

If a and b are two directed lines, and Π_a, Π_b their corresponding Plücker mappings, a relation $side(a, b)$ can be defined as the permuted inner product $\Pi_a \odot \Pi_b$:

$$\Pi_a \odot \Pi_b = (\pi_{a0}\pi_{b4} + \pi_{a1}\pi_{b5} + \pi_{a2}\pi_{b3} + \pi_{a4}\pi_{b0} + \pi_{a5}\pi_{b1} + \pi_{a3}\pi_{b2})$$

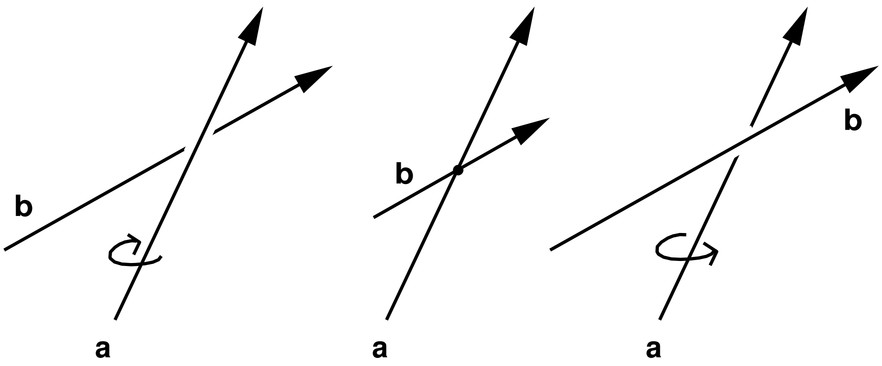

Figure 2: The right-hand rule in the context of the relation $side(a, b)$.

This sidedness relation has a geometric interpretation similar to the well-known "right-hand rule" (Figure 2): if the thumb of one's right hand is directed along a, then $side(a, b)$ is positive (negative) if b goes by a with (against) one's fingers. If lines a and b are coplanar (i.e., intersect or are parallel), $side(a, b)$ is zero.

Thus, the six-tuple Π_l can be treated either as a (homogeneous) point in P^5 or, after permutation, as the coefficients of a 5-dimensional hyperplane. The advantage of transforming lines to Plücker coordinates is that detecting incidence of lines in R^3 is equivalent to computing the inner product of a homogeneous point (the mapping of one line) with a hyperplane (the mapping of the other).

Plücker coordinates simplify computations on lines by mapping them to points and hyperplanes, which are familiar objects. However, although every directed line in R^3 maps to a point in Plücker coordinates, not every six-tuple, interpreted as Plücker coordinates, corresponds to a *real line*. Only those points Π satisfying the quadratic relation

$$\Pi \odot \Pi = 0 \tag{2}$$

correspond to real lines in R^3. All other points map to *imaginary lines*.

The six Plücker coordinates of a real line are not independent. First, since they describe a projective space, they are distinct only to within a scale factor. Second, they must satisfy Equation 2. Thus, the six Plücker coordinates describe a four-parameter space. This confirms basic intuition: one could describe all lines in R^3 in terms of, for example, their intercepts on two standard planes.

The set of points in P^5 satisfying Equation 2 is called the *Plücker quadric* [12]. One might visualize this set as a four-dimensional ruled surface embedded in R^5 that is analogous to a quadric hyperboloid of one sheet in R^3 (Figure 3).

Henceforth, we use the notation $\Pi : l \to \Pi_l$ to denote the map Π that takes a directed line l to the Plücker six-tuple Π_l, and the notation $\mathcal{L} : \Pi \to l_\Pi$ to denote the map that takes any point Π on the Plücker quadric and constructs the corresponding

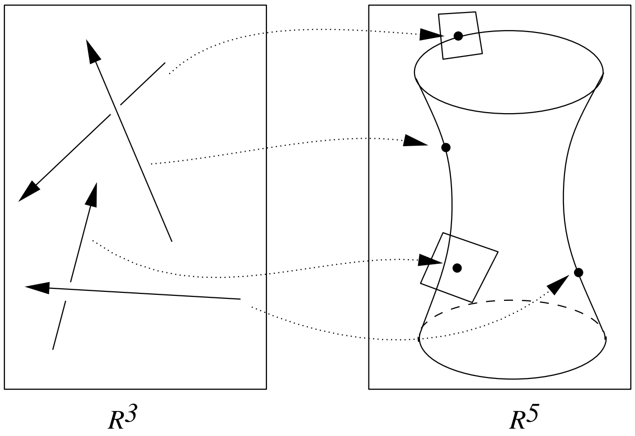

Figure 3: Real lines map to points on, or hyperplanes tangent to, the Plücker surface.

real directed line l_Π in R^3. Finally, given a six-tuple h representing a hyperplane in Plücker coordinates, we use h^+ to denote the closed halfspace bounded by h.

3 Computing a Stabbing Line

The input polygons to be stabbed have a total of n edges $E_k, 0 \le k < n$. Each edge E_k is a segment of a directed line e_k. Since the polygons are oriented, the e_k can be directed so that if a stabbing line S exists through all of the polygons, it must have the same sidedness relation with respect to each of the e_k. That is, S must satisfy (Figure 4):

$$side(S, e_k) \ge 0 \qquad \forall k.$$

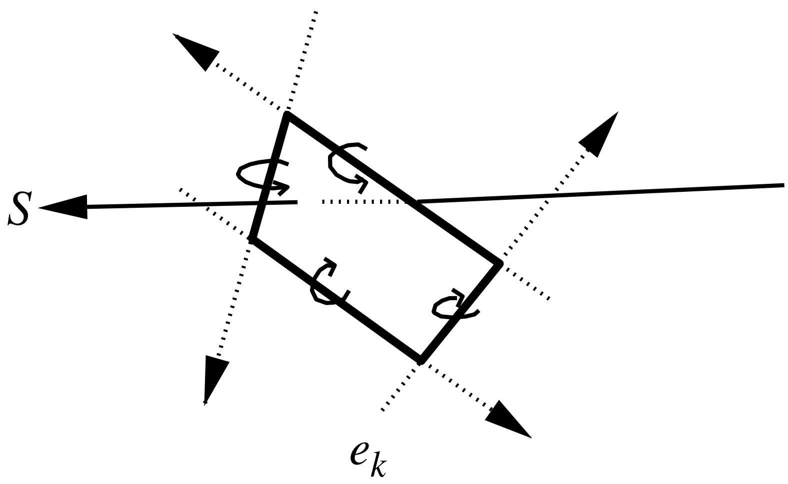

Figure 4: The stabbing line S must pass to the same side of all the e_k.

Define h_k as the oriented Plücker hyperplane corresponding to the directed line e_k:

$$h_k = \{\mathbf{x} \in \mathbf{P}^5 : \pi_{k4}x_0 + \pi_{k5}x_1 + \pi_{k3}x_2 + \pi_{k2}x_3 + \pi_{k0}x_4 + \pi_{k1}x_5 = 0\},$$

or

$$h_k = \{\mathbf{x} \in \mathbf{P}^5 : \mathbf{x} \odot \pi_k = 0\}. \tag{3}$$

For any stabbing line S, side$(S, e_k) \geq 0$. That is, $s \odot \pi_k \geq 0$, where $s = \Pi(S)$. Thus, s must be above all the hyperplanes h_k (Figure 5), and inside or on the boundary of the convex polytope $\bigcap_k h_k^+$. We say that such a point s is *feasible* with respect to the h_k.

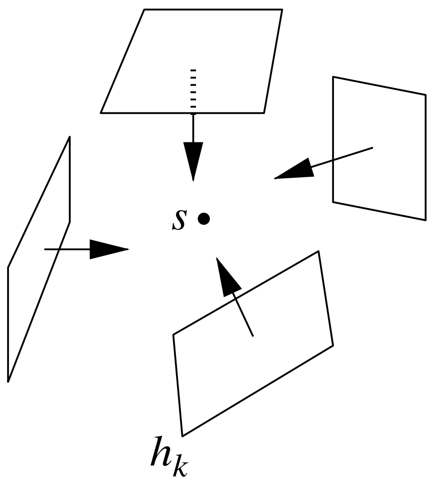

Figure 5: If S is a stabbing line, $s = \Pi(S)$ must be above all of the h_k in R^5.

The face graph of the polytope $\bigcap_k h_k^+$ has worst-case complexity quadratic in the number of halfspaces defining it [4], and can be computed by a randomized algorithm in optimal $O(n^2)$ expected time [3]. It is not sufficient merely to find a point inside this polytope, since most such points will not correspond to real lines. Rather, a stabbing line through the polygons exists if and only if there exists some point inside or on the boundary of the convex polytope, and on the Plücker quadric. Our algorithm computes such a point, if one exists.

Before proceeding with the algorithm, we review one important fact. If *any* stabbing line exists through the polygons, some stabbing line exists that is *tight*, or incident, on four edges from the original polygons [11]. The set of all such lines are the so-called *extremal* stabbing lines [8]. They may arise via incidence on two vertices from different polygons; from a single vertex and two skew edges from different polygons; or from four mutually skew edges from different polygons. It is these extremal stabbing lines which our algorithm identifies.

Consider the structure of the polytope bounding $\bigcap_k h_k^+$. Its zero-simplices, or vertices, arise as the intersection of five hyperplanes h_k. Its one-simplices, or edges, arise from the intersection of four of the h_k. Thus, any point on an edge of $\bigcap_k h_k^+$ and on the Plücker quadric corresponds (by the Plücker mapping) to a real line tight on four of the lines e_k (by projective transformation, we can always choose the plane at infinity so that it does not intersect $\bigcap_k h_k^+$).

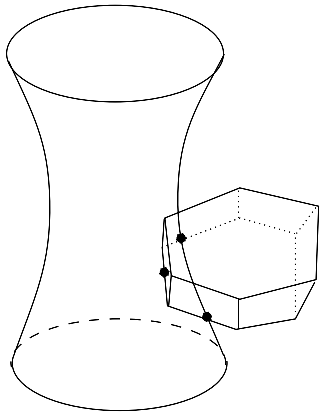

Figure 6: The algorithm intersects the edges of $\bigcap_k h_k^+$ with the Plücker quadric.

Thus, to discover a stabbing line, we need only find the intersection of an edge of the polytope $\bigcap_k h_k^+$ with the Plücker quadric (Figure 6). The combinatorial structure of $\bigcap_k h_k^+$ implies $O(n^2)$ sets of four lines chosen from the e_k. Any four lines l_i determine 0, 1, 2, or an infinite number of lines tight on the l_i. This is simply because the four lines imply an intersection of four hyperplanes in R^5, which is just a line in R^5. This line intersects the Plücker quadric in 0, 1, 2, or an infinite number of points. (The infinite intersection can arise due to the fact that the Plücker quadric is a ruled surface.) A procedure for computing the set of tight lines, and determining the type of line-surface intersection, is given in [13].

For each edge of $\bigcap_k h_k^+$, we examine the infinite line containing the edge for intersections with the Plücker quadric. Any such intersections represent lines that are incident on four of the e_k (the lines affine to the polygon edges in R^3). However, we must check that the intersection point in R^5 actually occurs inside $\bigcap_k h_k^+$. We do so by comparing this point to the faces (hyperplanes) bounding the convex hull edge. This can be done in constant time for any edge of $\bigcap_k h_k^+$, assuming that its face graph is suitably represented.

4 Implementation

Our implementation is based on three "primitives": 1) a d-dimensional linear programming algorithm; 2) a d-dimensional convex hull algorithm; and 3) an algorithm that computes the line(s) through four lines. The implementation of the stabbing line algorithm can be sketched as follows:

1. input the directed edges E_k

2. orient the edge endpoints to produce the directed lines e_k
3. transform the e_k to oriented Plücker halfspaces $h_k = \Pi(e_k)$
4. find f such that $f \odot h_k \geq 0$ for all k
   (linear programming: find (f,c) maximizing c subject to $f \odot h_k - c \geq 0$)
5. if no such f exists, return; there is no stabbing line
6. if $c = 0$ handle degenerate input
7. dualize the h_k about f to produce the point set $p_k = \frac{h_k}{f \odot h_k}$ in R^5
8. compute the convex hull $\mathrm{Conv}(p_k)$
9. compute the dual of $\mathrm{Conv}(p_k)$; i.e., the polytope $\bigcap_k h_k^+$
10. examine the edges of $\bigcap_k h_k^+$ for intersections with the Plücker quadric
11. if an intersection is found, check that it is in the interior of $\bigcap_k h_k^+$
12. if the intersection is valid, remap via $\mathcal{L}$ to construct a real stabbing line.

The first primitive, linear programming, is implemented as a randomized algorithm and runs in expected linear time [10]. The second primitive, convex hull computation in R^5, requires $O(n^2)$ time in principle [3]. We have implemented it, however, using a d-dimensional Delaunay simplicialization algorithm supplied by Wilks and McIntosh [15], which is somewhat slower. The third primitive, line incidence, runs in constant time.

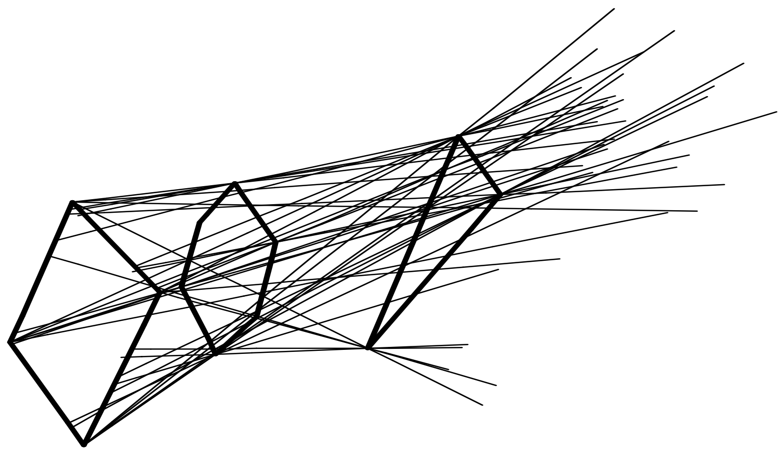

Figure 7: The thirty-six extremal stabbers of three oriented polygons ($n = 13$).

Figure 7 depicts the algorithm in operation. The input consists of three polygons: a square, a hexagon, and a triangle (thus $n = 13$). There are 270 *possible* extremal stabbing lines; one vertex and a pair of edges can be chosen in $4 \times 6 \times 3 \times 3 = 216$ ways (where the last factor of 3 is due to the fact that the vertex can be chosen from each of the three input polygons); two vertices can be chosen in $4 \times 6 + 4 \times 3 + 3 \times 6 = 54$ ways. For this input, however, the convex hull in R^5 has 130 edges (that is, only 130 of the 270 possible lines incident on four input edges satisfy all n Plücker constraints). These 130 convex hull edges yield 36 intersections with the Plücker quadric, and thus 36 real, extremal stabbing lines.

Conclusion

Using a duality relationship connecting directed lines in three-space, and point-hyperplane relationships in five-space, we have described an algorithm that finds all extremal stabbing lines through a set of oriented polygons with total complexity n, if any exist. The algorithm relies on a convex hull computation in R^5, which can be completed by a randomized algorithm in expected $O(n^2)$ time. We argued that only the edges of the generated polytope need be checked for a stabbing line solution. The resulting combinatorial structure has complexity $O(n^2)$ and is inspected deterministically, yielding an expected $O(n^2)$ time algorithm.

Acknowledgments

We are indebted to Allan Wilks and Allen McIntosh of AT&T Bell Labs for supplying code to compute d-dimensional simplicial Delaunay decompositions. We also thank Raimund Seidel and Nina Amenta for their helpful insight and comments.

References

[1] Nina Amenta. Finding a line traversal of axial objects in three dimensions. To appear in *Proc. 3^{rd} Annual ACM-SIAM Symposium on Discrete Algorithms*, 1992.

[2] D. Avis and R. Wenger. Algorithms for line traversals in space. In *Proc. 3^{rd} Annual Symposium on Computational Geometry*, pages 300–307, 1987.

[3] Kenneth L. Clarkson, Kurt Melhorn, and Raimund Seidel. Four results on randomized incremental constructions. *In preparation*, 1991.

[4] B. Grünbaum. *Convex Polytopes.* Wiley-Interscience, New York, 1967.

[5] Michael E. Hohmeyer and Seth J. Teller. Stabbing isothetic rectangles and boxes in $O(n \lg n)$ time. Technical Report UCB/CSD 91/634, Computer Science De-

partment, U.C. Berkeley, 1991. Also to appear in *Computational Geometry: Theory and Applications*, 1992.

[6] M. McKenna and J. O'Rourke. Arrangements of lines in 3-space: A data structure with applications. In *Proc. 4^{th} Annual Symposium on Computational Geometry*, pages 371–380, 1988.

[7] N. Megiddo. Stabbing isothetic boxes in deterministic linear time. *Personal communication to Nina Amenta*, 1991.

[8] Marco Pellegrini. Stabbing and ray-shooting in 3-dimensional space. Technical Report 540; Robotics Report No. 230, New York University Courant Institute of Mathematical Sciences, Computer Science Division, 1990.

[9] Marco Pellegrini and Peter Shor. Finding stabbing lines in 3-dimensional space. In *Proc. 2^{nd} ACM-SIAM Symposium on Discrete Algorithms*, pages 24–31, 1991.

[10] Raimund Seidel. Linear programming and convex hulls made easy. In *Proc. 6^{th} ACM Symposium on Computational Geometry*, pages 211–215, 1990.

[11] Raimund Seidel. Personal communication, February 1991.

[12] D.M.Y. Sommerville. *Analytical Geometry of Three Dimensions.* Cambridge University Press, 1959.

[13] Seth J. Teller and Michael E. Hohmeyer. Computing the lines piercing four lines. Technical Report UCB/CSD 91/665, Computer Science Department, U.C. Berkeley, December 1991.

[14] Seth J. Teller and Carlo H. Séquin. Visibility preprocessing for interactive walkthroughs. *Computer Graphics (Proc. SIGGRAPH '91)*, 25(4):61–69, 1991.

[15] Allan Wilks, Allen McIntosh, and Richard A. Becker. The data dual. *In preparation*, 1992.