Computing the Antipenumbra
of an Area Light Source

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

Abstract

We define the *antiumbra* and the *antipenumbra* of a convex area light source shining through a sequence of convex areal holes in three dimensions. The antiumbra is the volume from which all points on the light source can be seen. The antipenumbra is the volume from which some, but not all, of the light source can be seen. We show that the antipenumbra is, in general, a disconnected set bounded by portions of quadric surfaces, and describe an implemented $O(n^2)$ time algorithm that computes this boundary, where n is the total number of edges comprising the light source and holes.

The antipenumbra computation is motivated by a visibility scheme in which we wish to determine the volume visible to an observer looking through a sequence of transparent convex holes, or *portals*, connecting adjacent cells in a spatial subdivision. Knowledge of the antipenumbra should also prove useful for rendering shadowed objects. Finally, we have extended the algorithm to compute the planar and quadratic surfaces along which the rate of areal variation in the visible portion of the light source changes discontinuously due to occlusion. These surfaces are relevant in polygon meshing schemes for global illumination and shadow computations.

CR Categories and Subject Descriptors: [Computer Graphics]: I.3.5 Computational Geometry and Object Modeling – *geometric algorithms, languages, and systems*; I.3.7 Three-Dimensional Graphics and Realism – *color, shading, shadowing, and texture*.

Additional Key Words and Phrases: Radiosity, aspect graph, discontinuity meshing, stabbing lines, Plücker coordinates.

1 Introduction

1.1 Penumbrae and Antipenumbrae

Suppose an area light source shines past a collection of convex occluders. The occluders cast shadows, and in general attenuate or eliminate the light reaching various regions of space. There is a natural characterization of any point in space in this situation, depending on how much of the light source can be "seen" by the point. Figure 1 depicts a two-dimensional example. If the point sees none of the light source (that is, if all lines joining the point and any part of the light source intersect an occluder), the point is said to be in *umbra*. If the point sees some, but not all, of the light source, it is said to be in *penumbra*. Otherwise, the point may see all of the light source.

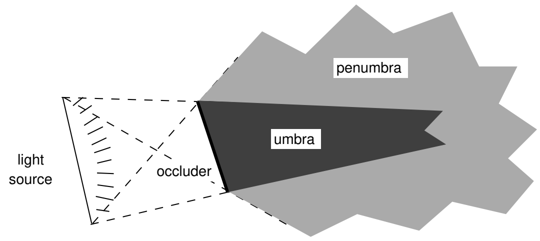


Figure 1: Umbra and penumbra of an occluder in 2D.

Imagine that the occluders are replaced by convex "holes," or transparent areas, in otherwise opaque planes. In a sense complementary to that above, every point in space can again be naturally characterized. We define the *antiumbra* cast by the light source as that volume from which the entire light source can be seen, and the *antipenumbra* as that volume from which some, but not all, of the light source can be seen (Figure 2). For a given light source and set of holes or occluders, the umbra is the spatial complement of the union of antiumbra and antipenumbra; similarly, the antiumbra is the spatial complement of the union of umbra and penumbra. Both the antiumbra and antipenumbra may be vacuous beyond the plane of the final hole, i.e., they may contain no points.

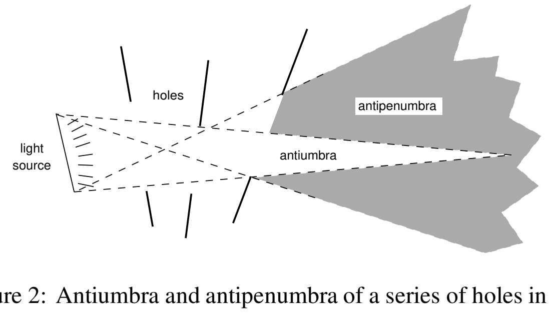


Figure 2: Antiumbra and antipenumbra of a series of holes in 2D.

For a *point* light source and a set of polygonal occluders in three

[‡]Computer Science Department, Berkeley, CA 94720 (seth@cs.berkeley.edu)

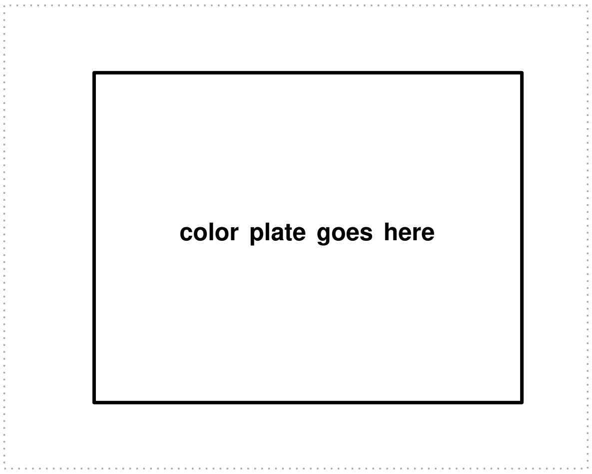


Figure 3: A spatial subdivision, the convex polyhedral cells of which are linked by convex polygonal transparent portals.

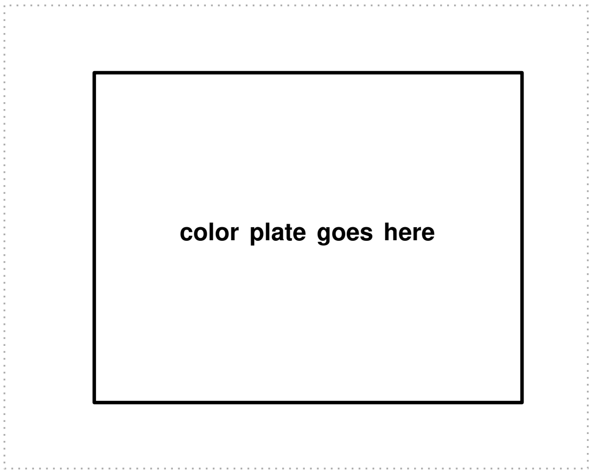


Figure 4: Successively narrowing antipenumbrae cast by the light source (here, the leftmost portal) through the cells of the subdivision.

dimensions, the penumbra is vacuous, and computing the umbra is straightforward [1, 7]. The problem becomes substantially more complex when the light source is lineal or areal. In general, as few as two occluders can give rise to shadows bounded by *reguli*, ruled quadric surfaces of negative Gaussian curvature [23], whose three generator lines arise from non-adjacent hole or light-source edges. Reguli were first used in the context of occlusion for the *aspect graph* computation, which catalogues all qualitatively distinct views of a polyhedral object under orthographic or perspective projection [14, 20]. At present, the best time bound for computing the aspect graph of a general polyhedral object with n vertices is $O(n^4 \lg n + m \lg m + c_t)$, where m is the number of qualitatively distinct views, at worst $O(n^6)$, and c_t the total number of changes between these views [10].

For a convex lineal or areal light source and a *single* convex occluder in 3D, the umbra and its union with the penumbra are convex. Such first-order shadows have been employed to yield convincing renderings of shadowed scenes [15, 16]. These penumbra algorithms, however, extend to multiple occluders only by effectively approximating the light source as a point or as a set of points. Several algorithms approximating multiple-occluder shadow boundaries have been described.

For example, in [16], the penumbra cast by multiple occluders is approximated by casting each occluder's penumbra individually, then performing polyhedral union and intersection operations on the result. An analogous approach is described in [4], where the light source is treated as a discrete set of point light sources, and the shadows of collections of occluders are cast and combined. An algorithm proposed in [19] replaces the area light source with a point at its center, and describes an error metric that bounds the spatial discrepancy between the computed and true penumbra. This error metric can then be used to control adaptive subdivision of the light source or occluders. Another recent algorithm approximates umbra volumes by constructing "penumbra trees" and "umbra trees"; these are BSP trees [8] whose polyhedral leaf cells bound polygon fragments in partial or complete shadow [5].

For polygonal light sources, holes, and occluders, the boundary between total illumination (antiumbra) and partial illumination (penumbra, antipenumbra) is piecewise-planar. The boundary between partial illumination (penumbra, antipenumbra) and no illumination (umbra), however, generally consists of portions of reguli. Generically, any connected portion of a regulus is non-convex, and any volume bounded by a portion of a regulus must be non-convex. Consequently, no algorithm that uses only polyhedral primitives or aggregations of convex objects can exactly represent the antipenumbra or umbra of an area light source for arbitrary polygonal occluders in three dimensions.

1.2 Motivation

Antipenumbra determination is motivated by visibility computations in a polygonal environment. Space is subdivided into polyhedral cells, which are mutually visible only through sequences of *portals*, or convex polygonal holes [9, 26, 27]. Figure 3 depicts five cells of one such polyhedral subdivision, and the portals linking adjacent cells. An observer constrained to a given *source* cell can see out of the cell only through a portal (say, that of the leftmost cell in Figures 3 and 4). If this portal is treated as an area light source, the antipenumbra cast by the portal beyond the plane of any further portal in a sequence bounds the volume visible to the constrained observer. For an observer determined to be inside the source cell, only objects inside this antipenumbral volume can be potentially visible.

A description of the antipenumbra may also prove useful for radiosity computations. For example, only polygons mutually visible through a portal sequence (i.e., in each other's antipenumbrae) can interact directly by exchanging luminous energy. Knowing a light source's antipenumbra would also be useful in the polygonal subdivision that shadowing and global illumination algorithms employ to model shadow boundaries [3, 4, 12, 13, 21].

Finally, the algorithm is of theoretical interest for two reasons. First, for the class of input described here, the algorithm computes "strong" (antiumbral) and "weak" (antipenumbral) polygon visibility [17] with respect to a polygon (area light source) in 3D. Second, again for this input class, the algorithm demonstrates an upper time and storage bound of $O(n^2)$ on the aspect graph computation.

1.3 Overview

It can easily be shown that, in three-space, the antiumbra is convex, polyhedral, has complexity $O(n)$, and can be computed in $O(n \lg n)$ time, where n is the total polygon edge complexity of a convex light source and some number of convex areal holes. Here, we show that the three-space antipenumbra is, in general, non-convex and disconnected, but has at most quadratic complexity in n. We present an implemented $O(n^2)$ time algorithm that computes the piecewise-quadratic boundary of the antipenumbra.

We also describe an extension of the algorithm that computes linear and quadratic *event surfaces* [10]. These are the loci at which the rate of areal variation in the visible portion of the light source, and therefore the derivative of illumination intensity, changes discontinuously due to occlusion. Such surfaces are useful, for example, in polygon meshing schemes for global illumination and shadow computations.

The algorithm uses Plücker coordinates, a five-dimensional line representation, to transform the edges of the light source and holes into hyperplanes bounding a five-dimensional polytope. The face structure of this polytope is then intersected with a four-dimensional quadric surface, the Plücker quadric. The resulting intersections or *traces* are remapped to 3D objects such as stabbing lines and portions of planar and quadric surfaces, some of which are shown to bound the antipenumbra. The remaining traces correspond to surfaces of illumination rate discontinuity lying within the antipenumbra.

2 Plücker Coordinates

We use the Plücker coordinatization [23] 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 3D. 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

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

We use the following convention dictating the correspondence between the minors of Equation 1 and the $\pi_{\ell i}$:

$$\begin{aligned}
\pi_{\ell 0} &= p_x q_y - q_x p_y \\
\pi_{\ell 1} &= p_x q_z - q_x p_z \\
\pi_{\ell 2} &= p_x \quad - q_x \\
\pi_{\ell 3} &= p_y q_z - q_y p_z \\
\pi_{\ell 4} &= p_z \quad - q_z \\
\pi_{\ell 5} &= q_y \quad - p_y
\end{aligned}$$

(this order was adopted in [18] to produce positive signs in some useful identities involving Plücker coordinates).

If a and b are two directed lines, and Π_a, Π_b their corresponding Plücker *duals*, a relation *side*(a, b) can be defined as the permuted inner product

$$\Pi_a \odot \Pi_b = \pi_{a0}\pi_{b4} + \pi_{a1}\pi_{b5} + \pi_{a2}\pi_{b3} + \pi_{a4}\pi_{b0} + \pi_{a5}\pi_{b1} + \pi_{a3}\pi_{b2}. \tag{2}$$

This sidedness relation can be interpreted geometrically with the right-hand rule (Figure 5): 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.

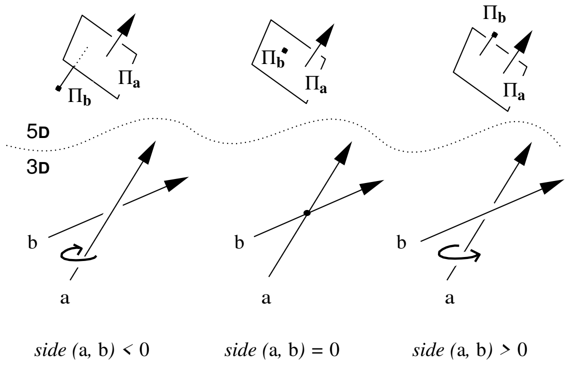


Figure 5: The right-hand rule applied to *side*(a, b).

The six-tuple Π_ℓ can be treated either as a homogeneous point in 5D, or (after suitable permutation via Equation 2) as the coefficients of a five-dimensional hyperplane. The advantage of transforming lines to Plücker coordinates is that detecting incidence of lines in 3D becomes equivalent to computing the inner product of a homogeneous point (the dual of one line) with a hyperplane (the dual 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 3D maps to a point in Plücker coordinates, not every point in Plücker coordinates corresponds to a *real line*. Only those points Π satisfying the quadratic relation

$$\Pi \odot \Pi = 0 \tag{3}$$

correspond to real lines in 3D. All other points correspond 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 3. Thus, Plücker coordinates describe a four-parameter space. This confirms basic intuition: one could describe all lines in three-space in terms of, for example, their intercepts on two standard planes.

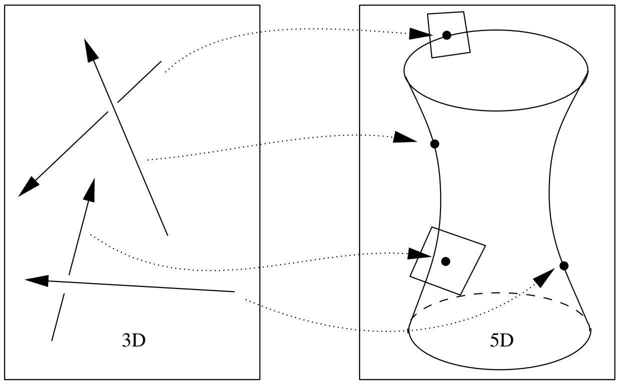


Figure 6: Directed lines map to points on, or hyperplanes tangent to, the Plücker surface.

The set of points in 5D satisfying Equation 3 is a quadric surface called the *Plücker surface* [23]. One might visualize this set as a four-dimensional ruled surface embedded in five dimensions, analogous to a quadric hyperboloid of one sheet embedded in three-space (Figure 6).

Henceforth, we use the notation $\Pi : \ell \to \Pi(\ell)$ to denote the map Π that takes a directed line ℓ to the Plücker point $\Pi(\ell)$, and the notation $\mathcal{L} : \Pi \to \mathcal{L}(\Pi)$ to denote the map that takes any point Π on the Plücker surface and constructs the corresponding real directed line $\mathcal{L}(\Pi)$. For any plane or hyperplane h, we denote the closed nonnegative halfspace of h as h^+, and say that a point in this halfspace is *on* or *above* h.

3 Extremal Stabbing Lines

We wish to describe the set of light rays originating at the light source and passing through each of a sequence of holes. Our algorithm exploits the fact that the holes are *oriented* by the sense in which they are traversed during a directed graph search through the portals of a spatial subdivision. For a particular hole sequence, each hole can be considered to admit light in only one direction.

In the following analysis, we call the light source and holes the *generator polygons*. In total, these polygons have n directed *generator edges* E_k, $k \in 1, \ldots, n$ (we assume at first that no two edges from different polygons are coplanar). Each edge E_k is a segment of a directed line e_k. Since the polygons are oriented, the e_k can be arranged so that if some directed line s *stabs* (intersects) each polygon, it must have the same sidedness relation with respect to each e_k. That is, any stabbing line s must satisfy (Figure 7):

$$\textit{side}(s, e_k) \geq 0, \qquad k \in 1, \ldots, n.$$

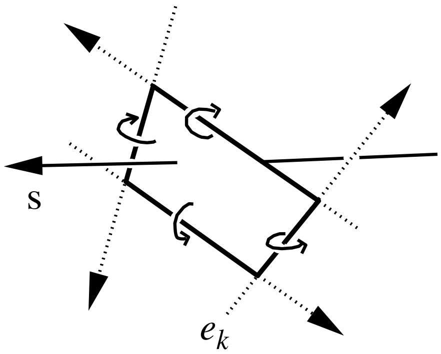


Figure 7: The stabbing line s must pass to the same side of each e_k.

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

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

For any stabbing line s, *side* $(s, e_k) \geq 0$. That is, $S \odot \Pi_k \geq 0$, where $S = \Pi(s)$, and $\Pi_k = \Pi(e_k)$. Thus, S must be on or above each hyperplane h_k (Figure 8), and inside or on the boundary of the convex polytope $\bigcap_k h_k^+$.

The face structure of the polytope $\bigcap_k h_k^+$ has worst-case complexity quadratic in the number of halfspaces defining it [2], and can be computed by a randomized algorithm in optimal $O(n^2)$ expected time [6]. Define the *extremal* stabbing lines as those lines incident on four generator edges [18]. Since four lines (i.e., constraints) are necessary to determine a line, if *any* stabbing lines exist, then at least one must be extremal. The structure of the polytope $\bigcap_k h_k^+$ yields all extremal stabbing lines [18, 25]. Each such line ℓ is incident, in 3D, upon four of the e_k. Consequently, the Plücker point Π_ℓ must

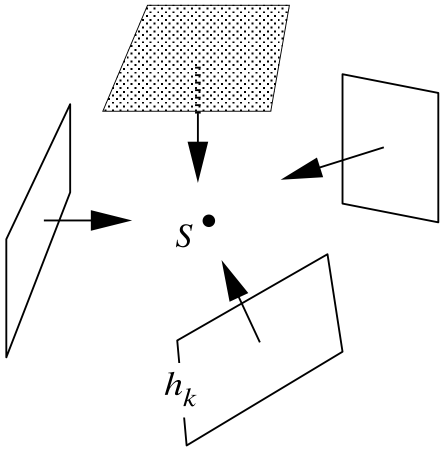


Figure 8: The 5D point $S = \Pi(s)$ must be on or above each h_k.

lie on four of the hyperplanes h_k in 5D, and must therefore lie on a *1D − face*, or edge, of the polytope $\bigcap_k h_k^+$. Thus, we can find all extremal stabbing lines of a given polygon sequence by examining the edges of the polytope for intersections with the Plücker surface. The extremal stabbing line corresponding to each intersection can be determined in constant time from the four relevant generator edges E_k [24].

Figure 9 depicts the output of an implementation of this algorithm. The input consists of nine polygons, with n = 39 edges total. The 5D convex polytope $\bigcap_k h_k^+$ has 755 edges, which together yield 82 intersections with the Plücker surface, and thus 82 extremal stabbing lines. All stabbing lines, considered as rays originating at the plane of the final hole, must lie within the antipenumbra of the "light source" (here, the leftmost polygon in the sequence). Some extremal stabbing lines lie in the interior of the antipenumbra because the edge graph of $\bigcap_k h_k^+$, when "projected" via $\mathcal{L}$ into three-space, overlaps itself.

2 x 3 color plate

Figure 9: The eighty-two extremal stabbing lines of nine oriented polygons in 3D. The total edge complexity n is thirty-nine.

4 Extremal Swaths (Event Surfaces)

There are three types of extremal stabbing lines: vertex-vertex, or VV lines; vertex-edge-edge, or VEE lines; and quadruple edge, or 4E lines. Imagine "sliding" an extremal stabbing line (of any type) away from its initial position, by relaxing exactly one of the four edge constraints determining the line (Figure 10). The surface, or *swath*, swept out by the sliding line must either be planar (if the line remains tight on a vertex) or a regulus, whose three generator lines embed three polygon edges. In the terminology of [10], swaths are VE or EEE *event surfaces* important in the construction of aspect graphs, since they are loci at which qualitative changes in occlusion occur.

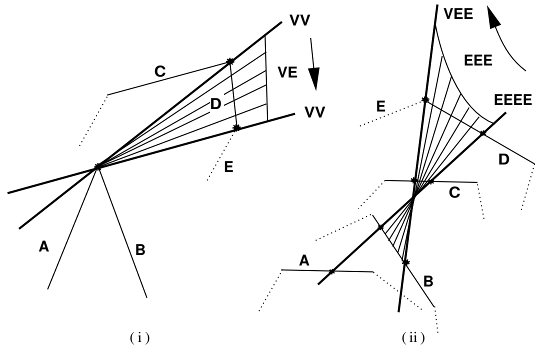


Figure 10: Sliding a stabbing line away from various extremal lines in 3D generates a VE plane (i) or a EEE quadratic surface (ii).

Figure 10-i depicts an extremal VV stabbing line tight on four edges A,B,C and D. Relaxing constraint C yields a VE (planar) swath tight on A, B, and D. Eventually, the sliding line encounters an obstacle (in this case, edge E), and terminates at a VV line tight on A,B,D, and E. Figure 10-ii depicts an extremal 4E stabbing line tight on the mutually skew edges A,B,C, and D; relaxing constraint A produces a EEE (regulus) swath tight on B, C, and D. The sliding line eventually encounters edge E and induces an extremal VEE line, terminating the swath.

Consider the same situations in the 5D space generated by the Plücker mapping (Figure 11). Extremal lines map to particular points in Plücker coordinates; namely, the intersections of the edges, or 1D-faces, of the polytope $\bigcap_k h_k^+$ with the Plücker surface. Since swaths are one-parameter line families, they correspond to *curves* in Plücker coordinates. These curves are the *traces* or intersections of the 2D-faces of $\bigcap_k h_k^+$ with the Plücker surface, and are therefore conics (in 5D, a polytope's 2D-faces are planar, and determined by the intersection of some three h_k). We call the 3D swaths corresponding to these 5D conic traces *extremal swaths*. All swaths have three generator lines, and consequently three generator edges, arising from the input polygons.

5 Internal and Penumbral Swaths

An object can be extremal (that is, lie on the boundary of the convex hull) in 5D, yet lie wholly inside the antipenumbra in three-space. Just as there are extremal stabbing lines that lie in the interior of the antipenumbra, so there are extremal swaths in the interior as well. We define a *penumbral* swath as an extremal swath that lies on the boundary of the antipenumbra (these are the red and green surfaces in Figure 4). All other extremal swaths are *internal*. Examining the 2D-faces of the polytope $\bigcap_k h_k^+$ yields all extremal swaths; however, we must distinguish between penumbral and internal swaths. This distinction can be made purely locally; that is, by examining only the swath's three generator edges and, in turn, their generator polygons. From only these (constant number of) objects, we show how to determine whether stabbing lines can exist on "both sides"

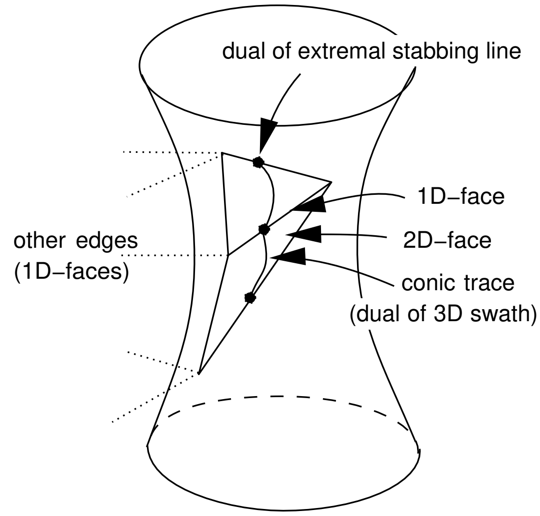


Figure 11: Traces (intersections) of extremal lines and swaths on the Plücker surface in 5D (higher-dimensional faces are not shown).

of the swath in question. If so, the swath cannot be penumbral, and is classified as internal.

5.1 Edge-Edge-Edge Swaths

Internal and penumbral EEE swaths can be distinguished as follows. Suppose three mutually skew generator edges A, B, and C give rise to an extremal EEE swath. Choose some line $\mathbf{L}$ incident on the generators respectively at points **a**, **b**, and **c** (Figure 12-i). At these points, erect three vectors $\mathbf{N}_a$, $\mathbf{N}_b$, and $\mathbf{N}_c$, perpendicular both to $\mathbf{L}$ and to the relevant generator edge, with their signed directions chosen so as to have a positive dot product with a vector pointing into the interior of the edge's generator polygon. (We refer to these vectors collectively as the $\mathbf{N}_i$.)

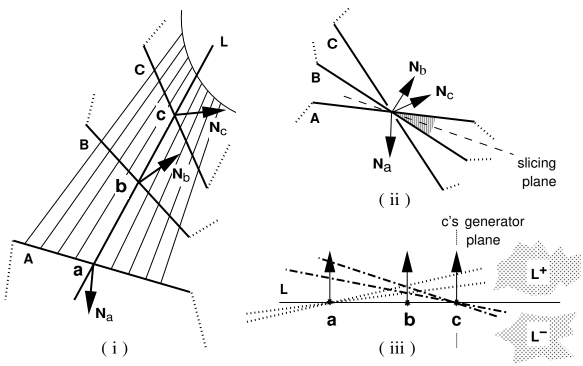


Figure 12: An internal EEE swath (i), viewed along $\mathbf{L}$ (ii) and transverse to $\mathbf{L}$ (iii). The $\mathbf{N}_i$ can be contained.

We say that three coplanar vectors can be *contained* if there exists a vector whose dot product with all three vectors is strictly positive. We claim that a swath is internal if and only if its corresponding $\mathbf{N}_i$ can be contained.

Suppose that the $\mathbf{N}_i$ can be contained (as in Figure 12). Consider any vector having a positive dot product with the $\mathbf{N}_i$ (such as one pointing into the gray region of Figure 12-ii). Next, "slice" the configuration with a plane containing both $\mathbf{L}$ and this vector, and view the swath in this plane (Figure 12-iii).

The trace (intersection) of the swath on this plane is simply the stabbing line $\mathbf{L}$, which partitions the slicing plane into two regions $\mathbf{L}^+$ and $\mathbf{L}^-$ beyond the plane of C's generator polygon (Figure 12-iii). We can move $\mathbf{L}$ infinitesimally by keeping it tight on, say, point **a**. The interiors of the other two holes allow $\mathbf{L}$ to pivot in only one direction, thus generating stabbing lines into $\mathbf{L}^+$ (dotted). Analogously, pivoting about point **c** generates stabbing lines into $\mathbf{L}^-$ (dashed). Since, by construction, the swath admits stabbing lines on both sides, it cannot be penumbral.

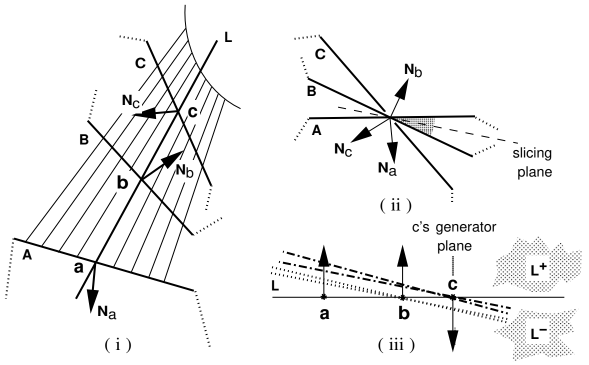


Figure 13: A penumbral EEE swath (i), viewed along $\mathbf{L}$ (ii) and transverse to $\mathbf{L}$ (iii). The N_i cannot be contained.

In contrast, the N_i of Figure 13-i cannot be contained. Thus, any vector (including one chosen from the gray region of Figure 13-ii) will have a negative dot product with at least one, and at most two, of the N_i. Suppose it has a negative dot product with N_c. Slice the configuration with a plane containing both $\mathbf{L}$ and one such vector (Figure 13-iii). Pivoting on point b generates stabbing lines into $\mathbf{L}^-$ (dotted), as does pivoting on point **c** (dashed). The configuration does not admit any stabbing lines into $\mathbf{L}^+$. Since this is true for any choice of slicing plane and (as we will show) for any choice of $\mathbf{L}$, we conclude that the swath is penumbral; i.e., it separates a region of zero illumination from a region of partial illumination.

5.2 Vertex-Edge Swaths

Suppose the swath in question has type VE. Label the two generator edges defining the swath vertex **v** as A and B, the remaining edge C (not in the plane of A and B), the two relevant generator polygons P and Q, and the plane through C and **v** as **S** (Figure 14). Orient **S** so that Q (C's generator polygon) is above it (i.e., in $\mathbf{S}^+$); this is always possible, since Q is convex. Plane **S** divides the space beyond the plane of Q into two regions $\mathbf{S}^+$ and $\mathbf{S}^-$. If stabbing lines can exist in only one of these regions, the swath is penumbral. This occurs if and only if **S** is a *separating plane* of P and Q; that is, if and only if polygon P is entirely below **S**.

Suppose **S** separates polygons P and Q. Imagine choosing some stabbing line from the swath, and moving it so that it comes free from the swath vertex **v**, but remains tight on edge C (and remains a valid, though non-extremal, stabbing line). If **S** separates P and Q, the moving line's intersection with P can only lie below **S** (Figure 14-i); thus, beyond the plane of Q, all such stabbing lines must intersect $\mathbf{S}^+$. Similarly, should the stabbing line move off of C and into Q's interior, while remaining tight on **v**, it can only intersect $\mathbf{S}^+$. The swath admits only stabbing lines in $\mathbf{S}^+$, and is therefore penumbral.

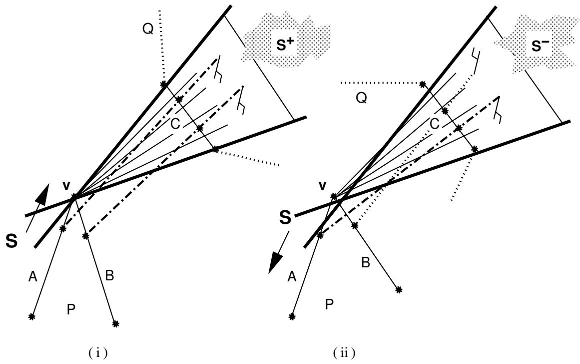


Figure 14: Penumbral (i) and internal (ii) VE swaths.

Suppose, in contrast, that the plane **S** does not separate P and Q, i.e., that one or both of the edges A and B lie in $\mathbf{S}^+$ (Figure 14-ii). Again move a swath line so that it comes free from the swath vertex **v**, but stays tight on edge C. If the line (dashed) moves along A above (say) plane S, it will intersect the region $\mathbf{S}^-$ beyond the plane of Q. Similarly, motion along edge B below **S** produces stabbing lines (dotted) in $\mathbf{S}^+$. The swath admits stabbing lines into both $\mathbf{S}^-$ and $\mathbf{S}^+$, and therefore cannot be penumbral.

Note that the three-vector construction for EEE swaths is applicable in this (degenerate) setting as well, since **S** is a separating plane if and only if the normals erected along A,B, and C cannot be contained.

5.3 The Containment Function

The containment function is a criterion for distinguishing between penumbral and internal swaths. However, it applies only along a single stabbing line, not over an entire swath. Fortunately, evaluating the containment function anywhere along a swath produces the same result. To see why this is so, consider any configuration of three coplanar vectors N_a, N_b, and N_c. Suppose that the configuration changes continuously from containable to non-containable (e.g., by rotation of vector N_c), as in Figure 15. At either moment of transition (marked with dotted lines in the figure), N_c and one of N_a or N_b must be antiparallel.

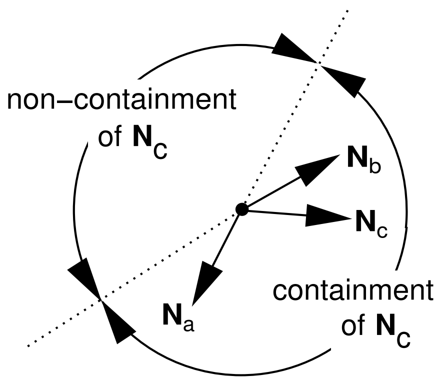
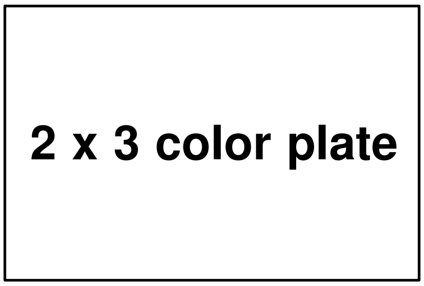
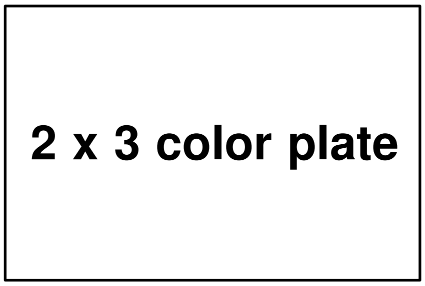


Figure 15: Directions of containment and non-containment for N_c, for fixed N_a and N_b. Transition directions are marked.

For this to occur in the EEE swath construction (Figure 13) two of the three generator edges must be coplanar. Similarly, in the VE swath construction (Figure 14), edge C and one of the edges A



(i) A light source and three holes (n = 15).



(ii) VE (red), EEE (green) penumbral swaths.

Figure 16: The antipenumbra cast by a triangular light source through three convex holes (n = 15).

or B must be coplanar. But the generator edges are *fixed*; only the sliding line varies. Thus, even though sliding an extremal line along a swath generates a continuously changing vector configuration, the *containment* of the vectors is a constant function. If either of the EEE or VE degenerate cases occurs in practice, the containment function is indeterminate. We detect this while examining the polytope face structure, and use special-case processing to generate the correct swath.

6 Computing the Antipenumbra

We have assembled all of the computational machinery necessary for determining the antipenumbra. We make the following assumptions: the input is a list of m oriented polygons, $P_1 \dots P_m$, given as linked lists of edges, the total number of edges being n. The first polygon P_1 is the light source, and all others are holes. The polygons are ordered in the sense that the negative halfspace determined by the plane of P_i contains all polygons P_j, $i < j \le m$ (thus, an observer looking along a stabbing line would see the vertices of each polygon arranged in counterclockwise order).

The algorithm dualizes each directed input edge to a hyperplane in Plücker coordinates, then computes the common intersection of the resulting halfspaces, a 5D convex polytope. If there is no such intersection, or if the polytope has no intersection with the Plücker surface, the antipenumbra is vacuous. Otherwise, the face structure of the polytope is searched for traces of penumbral swaths resulting from intersections of its 1D-faces (edges) and 2D-faces (generically, triangles) with the Plücker surface (cf. Figure 11). These traces are linked into loops, each corresponding to a connected component of the antipenumbral boundary. There may be multiple loops since the intersection of the polytope boundary with the non-planar Plücker surface may have several components.

Each loop consists of polytope edges and triangles in alternation. Each edge intersection with the Plücker surface (a point in 5D) is the dual of an extremal stabbing line in 3D; each triangle intersection with the Plücker surface (a conic in 5D) is the dual of an extremal swath in 3D. Incidence of an edge and triangle on the polytope implies adjacency of the corresponding line and swath in 3D; stepping across a shared edge from one triangle to another on the polytope is equivalent to stepping across a shared extremal stabbing line (a "seam") between two extremal swaths in three-space. Thus the algorithm can "walk" from 2D-face to 2D-face on the polytope's surface, crossing the 1D-face incident to both at each step. Each closed loop found in this manner in 5D is the dual (under the Plücker mapping) of the boundary of one connected component of the antipenumbra in three-space.

The algorithm can be described in pseudocode as:

input directed edges E_k from polygons $P_1 \dots P_m$

convert directed edges to directed lines e_k

transform e_k to Plücker halfspaces $h_k = \Pi(e_k)$

compute 5D convex polytope $\bigcap_k h_k^+$

identify 2D-face intersections of $\bigcap_k h_k^+$ with Plücker surface

classify resulting extremal swaths as penumbral or internal

for each connected component of the antipenumbra

    traverse penumbral 2D-faces of $\bigcap_k h_k^+$

        map each trace found to a 3D penumbral swath

        output swath loop as piecewise quadratic surface

7 Implementation

We have implemented the antipenumbra algorithm in C and FORTRAN-77 on a 20-MIP Silicon Graphics superworkstation. To compute the convex hull of n hyperplanes in 5D, we used a d-dimensional Delaunay simplicialization algorithm implemented by Allan Wilks at AT&T Bell Labs and Allen McIntosh at Bellcore, and a d-dimensional linear programming algorithm [22] implemented by Michael Hohmeyer at U.C. Berkeley.

Figure 16-i depicts a set of four input polygons with $n = 15$, and the leftmost polygon acting as a light source. The antipenumbra computation took about 2 CPU seconds. The polytope $\bigcap_k h_k^+$, formed from 15 halfspaces in 5D, has 80 facets, 186 2D-faces (triangles), and 200 1D-faces (edges). Of the 186 triangles, 78 induce dual traces on the Plücker surface, generating 78 extremal swaths in 3D. Of these 78 swaths, 62 are internal, and 16 are penumbral (Figure 16-ii); of these, 10 are planar VE swaths (red), and 6 are quadric EEE swaths (green). Of the 200 edges, 39 intersect the Plücker surface to yield extremal VV, VEE, or 4E stabbing lines (Figure 17-i). Note that the extremal stabbing lines form the "seams" of the antipenumbral boundary, as they demarcate junctions between adjacent VE and/or EEE swaths (Figure 17-ii).

2 x 3 color plate

(i) The 39 extremal stabbing lines from Figure 16.

2 x 3 color plate

(ii) Extremal stabbing lines as penumbral "seams."

Figure 17: The relationship between extremal stabbing lines and penumbral swaths.

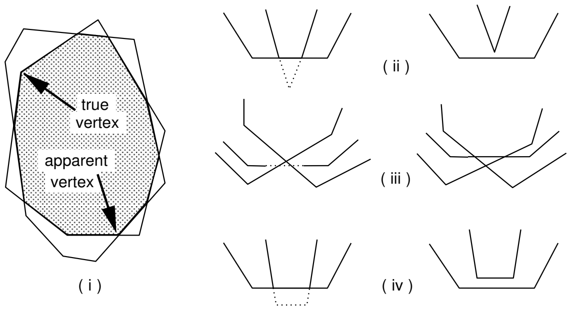


Figure 18: An observer's view of the light source (i). Crossing an extremal VE (ii), EEE (iii), or degenerate (iv) swath.

2 x 3 color plate

Figure 19: Curves of discontinuous illumination from the antipenumbra of Figure 16, intersected with a receiver plane.

2 x 3 color plate

(i) The antipenumbra cast by a light shining through two slits.

2 x 3 color plate

(ii) The third slit "clips" the antipenumbra, yielding two pieces.

Figure 20: An area light source and three holes can yield a disconnected antipenumbra.

8 Illumination Discontinuities

Internal VE and EEE swaths, which were merely categorized and discarded by the antipenumbra algorithm, are generally surfaces along which discontinuities occur in the first or second spatial derivatives of illumination intensity. Imagine an observer inside an antipenumbral volume, looking back through the generating hole sequence to the light source. Since each intervening hole edge "clips" a halfplane away from the light source, the observer must see a convex polygonal region of the light (Figure 18-i). The region edges arise directly from hole edges. The region vertices arise either from hole vertices, or from apparent intersections among non-adjacent hole edges as seen by the observer.

As the observer moves, the apparent area of the light source (and therefore the illumination intensity at the observer location) changes smoothly, except when a swath (i.e., event surface) is encountered. Generically, this happens in one of two fundamental ways for polyhedral objects [11]. Either a hole vertex appears (disappears) from the region boundary (Figure 18-ii), or the observer's line of sight simultaneously intersects three edges, causing the appearance (disappearance) of an apparent boundary vertex (Figure 18-iii). In either case a qualitative change occurs in the effect of infinitesimal observer motion on the apparent area of the light source.

Imagine the observer's motion as "sweeping" an occluding halfplane, determined by a hole edge and the observer position, over the light source. In general, each sweep surface occludes or reveals an area that changes quadratically with the observer position (parallel edges, as in Figure 18-iv, can cause linear variation in apparent area). Along extremal swaths, the set of contributing edges changes, adding or removing linear or quadratic terms from the function relating the amount of visible light area to observer location. Consequently, extremal swaths are in general loci of first- or second-derivative discontinuities in illumination intensity. (In the terminology of [13], planar traces of extremal swaths are curves of D^1 or D^2 discontinuity.)

Zeroth-derivative discontinuities (i.e., abrupt changes in illumination value) can occur only in the presence of a point light source, or if the observer crosses the plane of an area light source. Neither of these circumstances occurs in the antipenumbra computation, since (1) all light sources and holes are areal, and (2) no portal plane can be traversed twice during any valid portal sequence through the spatial subdivision.

The internal swaths induced by the polygon configuration of Figure 16, intersected with a plane beyond the final hole to produce segments of lines and conics, are shown in Figure 19. The segments comprise a planar curve arrangement of complexity $O(n^2)$, each cell of which contains points having the same *aspect*, or qualitative view, of the light source. Of course, penumbral swaths also demarcate qualitative changes in occlusion, since by stepping across one, an observer would see the light source appear or disappear from view.

Finally, we show how an area light source and as few as three holes can produce a disconnected antipenumbra. First, a light source and two holes, all thin rectangular slits, are arranged so as to admit a fattened regulus of antipenumbral light (Figure 20-i). A third slit then partitions the fattened regulus into two disconnected components (Figure 20-ii).

Conclusion

We presented an $O(n^2)$ time algorithm that computes the antipenumbra cast by a convex light source through a sequence of convex holes in three dimensions, with total edge complexity n. We described the antipenumbra as a disconnected volume bounded by regions of quadric and planar surfaces, and showed that each swath, or portion of the antipenumbral boundary in three-space, arises from a conic segment on the intersection of a five-dimensional convex hull with a four-dimensional quadric surface, the Plücker quadric. The algorithm is related to the aspect graph computation, in that it generates surfaces that separate regions with qualitatively distinct views of the light source or the intervening holes. We identified both penumbral and internal swaths as surfaces along which the first derivative of illumination intensity changes discontinuously due to occlusion.

We demonstrated an implementation of the antipenumbra computation on several polygon sequences. The algorithm was motivated by a visibility scheme involving static, volume-based culling in three dimensions. Knowledge of the antipenumbra may also prove useful, however, in shadowing algorithms and meshing schemes that model illumination discontinuities.

Acknowledgments

Jim Winget, my mentor at Silicon Graphics, was an extraordinary motivator and facilitator of this work. My advisor Carlo Séquin has been a constant source of inspiration, encouragement, and valuable observations. The late Professor René de Vogelaere taught me about reguli and classical geometry. Ziv Gigus, John Airey, Jim Ruppert, and Efi Fogel made thoughtful comments on an early draft of this paper. Raimund Seidel and Michael Hohmeyer also contributed helpful insight and comments. Paul Haeberli provided aesthetic advice, and helped prepare the color figures for submission and publication. Finally, Silicon Graphics afforded me access to their considerable physical resources.

References

[1] P. Atherton, K. Weiler, and D. Greenberg. Polygon shadow generation. *Computer Graphics (Proc. SIGGRAPH '78)*, 12:275–281, 1978.

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

[3] Daniel R. Baum, Stephen Mann, Kevin P. Smith, and James M. Winget. Making radiosity usable: Automatic preprocessing and meshing techniques for the generation of accurate radiosity solutions. *Computer Graphics (Proc. SIGGRAPH '91)*, 25(4):51–60, 1991.

[4] A.T. Campbell III and Donald S. Fussell. Adaptive mesh generation for global diffuse illumination. *Computer Graphics (Proc. SIGGRAPH '91)*, 24(4):155–164, 1990.

[5] Norman Chin and Steven Feiner. Fast object-precision shadow generation for area light sources using BSP trees. In *Proc. 1992 Symposium on Interactive 3D Graphics*, pages 21–30, 1992.

[6] Kenneth L. Clarkson and Peter W. Shor. Applications of random sampling in computational geometry II. *Discrete Computational Geometry*, pages 387–421, 1989.

[7] Frank C. Crow. Shadow algorithms for computer graphics. *Computer Graphics (Proc. SIGGRAPH '77)*, 11(2):242–248, 1977.

[8] H. Fuchs, Z. Kedem, and B. Naylor. On visible surface generation by a priori tree structures. *Computer Graphics (Proc. SIGGRAPH '80)*, 14(3):124–133, 1980.

[9] Thomas A. Funkhouser, Carlo H. Séquin, and Seth J. Teller. Management of large amounts of data in interactive building walkthroughs. In *Proc. 1992 Workshop on Interactive 3D Graphics*, pages 11–20, 1992.

[10] Ziv Gigus, John Canny, and Raimund Seidel. Efficiently computing and representing aspect graphs of polyhedral objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(6):542–551, 1991.

[11] Ziv Gigus and Jitendra Malik. Computing the aspect graph for line drawings of polyhedral objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(2):113–122, 1990.

[12] Pat Hanrahan and David Salzman. A rapid hierarchical radiosity algorithm. *Computer Graphics (Proc. SIGGRAPH '91)*, 25(4):197–206, 1991.

[13] Paul S. Heckbert. *Simulating Global Illumination Using Adaptive Meshing*. PhD thesis, Computer Sciences Department, University of California, Berkeley, June 1991.

[14] J.J. Koenderink and A.J. van Doorn. The internal representation of solid shape with respect to vision. *Biol. Cybern.*, 32:211–216, 1979.

[15] Tomoyuki Nishita and Eihachiro Nakamae. Half-tone representation of 3-D objects illuminated by area sources or polyhedron sources. In *Proc. IEEE COMPSAC, 1983*, pages 237–242, 1983.

[16] Tomoyuki Nishita and Eihachiro Nakamae. Continuous-tone representation of three-dimensional objects taking account of shadows and interreflection. *Computer Graphics (Proc. SIGGRAPH '85)*, 19(3):23–30, 1985.

[17] Joseph O'Rourke. *Art Gallery Theorems and Algorithms*. Oxford University Press, 1987.

[18] Marco Pellegrini. Stabbing and ray-shooting in 3-dimensional space. In *Proc. 6^{th} ACM Symposium on Computational Geometry*, pages 177–186, 1990.

[19] Ken Perlin and Xue-Dong Wang. An efficient approximation for penumbra shadow. Technical Report 346, New York University Courant Institute of Mathematical Sciences, Computer Science Division, 1988.

[20] W.H Plantinga and C.R. Dyer. Visibility, occlusion, and the aspect graph. *Int. J. Computer Vision*, 5(2):137–160, 1990.

[21] David Salesin, Dani Lischinski, and Tony DeRose. Reconstructing illumination functions with selected discontinuities. In *Proc. 3^{rd} Eurographics Workshop on Rendering*, 1992.

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

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

[24] 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, 1991.

[25] Seth J. Teller and Michael E. Hohmeyer. Stabbing oriented convex polygons in randomized $O(n^2)$ time. Technical Report UCB/CSD 91/669, Computer Science Department, U.C. Berkeley, 1992.

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

[27] Seth J. Teller and Carlo H. Séquin. Visibility computations in polyhedral three-dimensional environments. Technical Report UCB/CSD 92/680, Computer Science Department, U.C. Berkeley, 1992.