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 areal light source shining through a sequence of convex areal holes in three dimensions. The antiumbra is the volume beyond the plane of the final hole 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.

The antipenumbra computation is motivated by visibility computations, and might prove useful in rendering shadowed objects. We also present an implemented extension of the algorithm that computes planar and quadratic surfaces of discontinuous illumination useful for polygon meshing in global illumination 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: Shadow computations, visibility, area light sources, radiosity, discontinuity meshing.

[‡]Computer Science Department, Berkeley, CA 94720

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 attenuating or eliminating 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 that 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 some occluder), the point is said to be in the *umbra* of the occluder. If the point sees some, but not all, of the light source, it is said to be in the *penumbra*. Otherwise, the point may see all of the light source.

Suppose that the occluders are replaced by convex "holes", or transparent regions in otherwise opaque infinite planes. In a sense complementary to that above, every point in space beyond the plane of the final hole can 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). Both the antiumbra and antipenumbra may be vacuous, i.e., they may contain no points. For a given light source and set of holes, the cast 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.

When the light source is a point, computing the umbra and penumbra of a set of polygonal occluders is straightforward [6, 1]. The problem becomes substantially more complex when the light source is linear or areal. In general, as few as two occluders can give rise to shadows bounded by *regulae*, or ruled quadric surfaces of negative Gaussian curvature, whose three generator lines are affine to three non-adjacent edges in the input. These regulae were first noted in the literature in the context of the *aspect graph* computation, which catalogues all qualitatively distinct views of a polyhedral object from viewpoints on some surface bounding the object, under orthographic or perspective projection [23, 22]. The best time bounds for computing the aspect graph of a general object are currently $O(n^4 \lg n + m \lg m + c_t)$ for an object with n vertices, where m is the number of qualitatively distinct views, and c_t the total number of changes between these views [10].

Restricted or approximate algorithms for direct computation of the penumbra have been proposed. For a convex lineal or areal light source and a single convex occluder

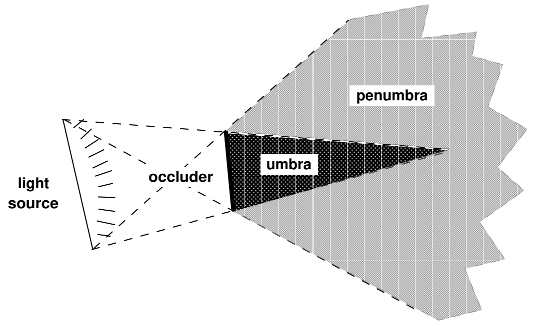

Figure 1: The umbra and penumbra of an occluder in R^2.

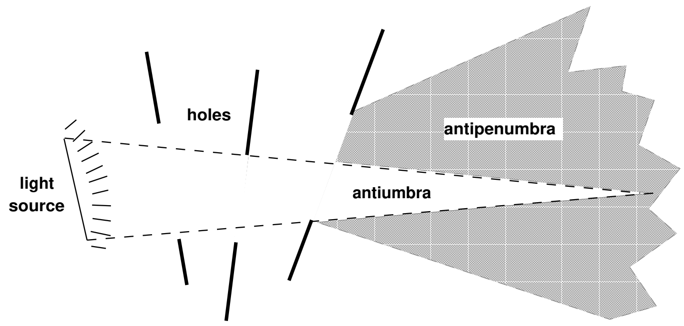

Figure 2: The antiumbra and antipenumbra of a series of holes in R^2.

in R^3, it is easy to show that both the umbra and penumbra are convex. Computing such first-order shadows has proven useful in a variety of global illumination algorithms [16, 17, 18]. These penumbrae algorithms extend to multiple occluders only by effectively approximating the light source as a point or as a set of points.

For example, in [17], the penumbra cast by multiple occluders is computed by casting each occluder's penumbra individually, then performing polyhedral union and intersection operations on the result. An analogous approach is proposed in [3], where the light source is treated as a set of point light sources, and the shadows of collections of occluders are cast and combined. An algorithm proposed in [21] 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 its occluders.

One recently proposed algorithm computes penumbra volumes, and polyhedral approximations to umbra volumes, by generating "penumbra trees" and "umbra trees"; these are BSP trees whose polyhedral leaf cells bound volumes in partial or complete shadow [4].

Regulae are quadrics, and have negative Gaussian curvature [25]. Thus any connected (superlineal) portion of a regulus is non-convex, and any volume bounded by a portion of a regulus must be non-convex. For polygonal light sources, holes, and occluders, the boundary between total illumination (antiumbra) and partial illumination (penumbra) is piecewise-linear. The boundary between partial illumination (penumbra) and no illumination (umbra), however, generally consists of portions of regulae. Consequently, no algorithm that uses only polyhedral primitives or aggregation of convex objects can exactly represent the antipenumbra of an area light source (or its complement, the umbra) for arbitrary polygonal occluders in R^3.

1.2 Motivation

The computation of antipenumbrae 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 [29]. If the first portal is thought of as a light source, the antipenumbra cast beyond the final portal must bound the volume visible to an observer whose eye is constrained to the first portal. Any objects beyond the plane of the final portal, but outside the antipenumbral volume, are determined invisible from the constrained observer's viewpoint [8].

Determination of the antipenumbra might also prove useful in a radiosity computation. For example, only polygons mutually visible through a portal sequence (i.e., in each other's antipenumbra) interact by exchanging luminous energy [15, 7, 9, 11]. Finally, knowledge of a light source's antipenumbra could be useful, for example, in the polygonal subdivision that the radiosity computation might employ to correctly represent shadow boundaries [3, 2, 13, 14].

1.3 Overview

It can easily be shown that, in three-space, the antiumbra is convex, has complexity $O(n)$, and can be computed in $O(n \lg n)$ time, where n is the total edge complexity of a convex light source and some number of convex, areal holes [26].

Here, we show that the three-space antipenumbra is, in general, non-convex and non-connected, and 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 present an extension of the algorithm that computes the so-called surfaces of D^1 and D^2 discontinuity [14] beyond the plane of the final hole; these are linear and quadratic surfaces on which illumination from the light source changes discontinuously due to occlusion.

The algorithm uses Plücker coordinates, a five-dimensional line representation, to transform the primal polygonal input into constraints on a dual five-dimensional polytope. The face-structure of this polytope is then intersected with a four-dimensional quadric surface, the Plücker quadric. The resulting dual traces are remapped to primal objects such as stabbing lines and portions of planar and quadric surfaces, which can be shown to bound the antipenumbra. The remaining dual traces are also surfaces of illumination discontinuity, but lie within the antipenumbra.

2 Plücker Coordinates

We use the Plücker coordinatization [25] 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. Following Pellegrini [20], we define the π_i as:

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

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 $\Pi_a \odot \Pi_b$:

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

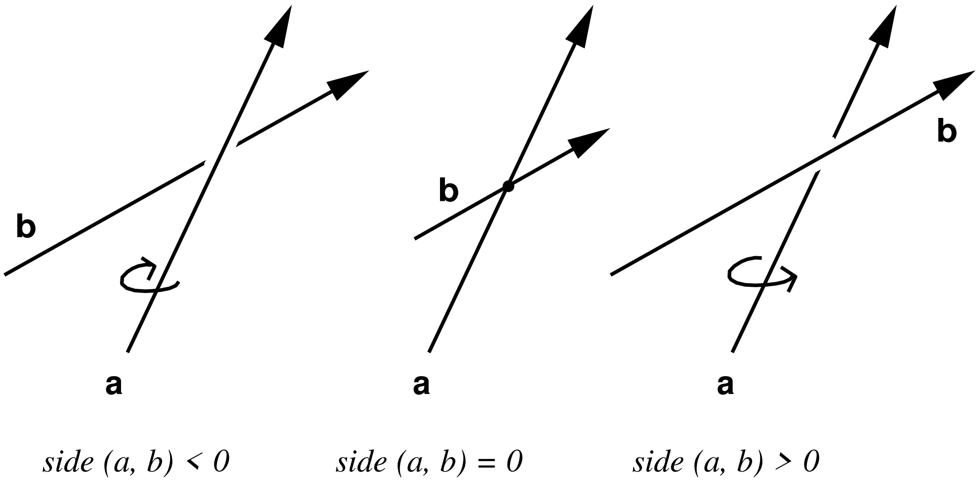

Figure 3: The right-hand rule in the context of the relation $side(a, b)$.

This sidedness relation has a geometric interpretation similar to the well-known "right-hand rule" (Figure 3): 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.

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 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 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 that satisfy 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 an implicit quadric surface called the *Plücker surface* [25]. 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 4).

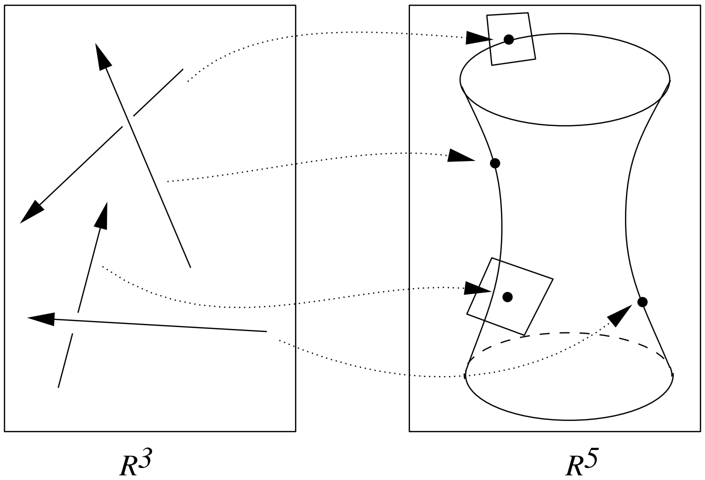

Figure 4: Directed lines map to points on, or hyperplanes tangent to, the Plücker surface.

Henceforth, we use the notation $\Pi : l \rightarrow \Pi_l$ to denote the map Π that takes a directed line l to the Plücker six-tuple Π_l, and the notation $\mathcal{L} : \Pi \rightarrow l_\Pi$ to denote the map that takes any point Π on the Plücker surface and constructs the corresponding real directed line l_Π in R^3. Given a six-tuple h representing a hyperplane in Plücker coordinates, we use h^+ to denote the closed positive halfspace bounded by h. Finally, we say that the space R^3 of oriented lines is the *primal* space, whereas the space of Plücker coordinates is the *dual* space.

3 Extremal Stabbing Lines

Call the light source and holes the "generator polygons". These polygons have n total "generator 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 some directed line S *stabs* (i.e., intersects the interior of) each polygon, it must have the same sidedness relation with respect to each of the e_k. That is, any stabbing line S must satisfy (Figure 5):

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

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$(S, e_k) \ge 0$. That is, $s \odot \pi_k \ge 0$, where $s = \Pi(S)$. Thus, s must be above all the hyperplanes h_k (Figure 6), 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.

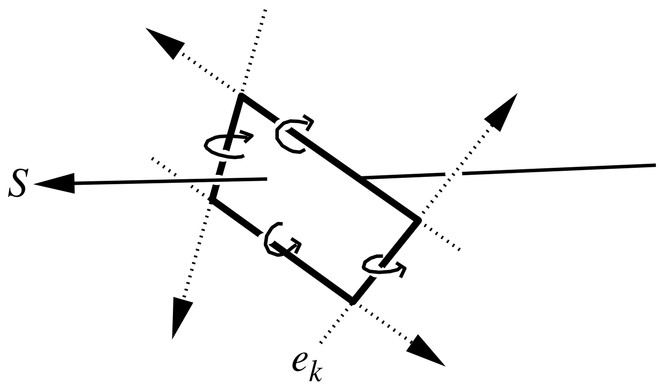

Figure 5: The stabbing line S must pass to the same side of all the e_k.

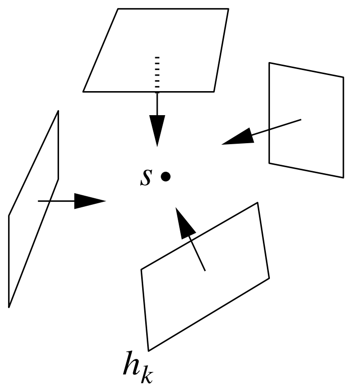

Figure 6: Dual point $s = \Pi(S)$ must be above all of the h_k in R^5.

The polytope $\bigcap_k h_k^+$ has worst-case complexity quadratic in the number of halfspaces defining it [12], and can be computed by a randomized algorithm in optimal $O(n^2)$ expected time [5]. Define the *extremal* stabbing lines as those lines incident on four generator edges [19]; if any stabbing lines exist then at least one must be extremal. We show in [28] that the structure of the polytope $\bigcap_k h_k^+$ yields all extremal stabbing lines. Each such line l is incident, in R^3, upon four of the e_k. Consequently, the Plücker point $\Pi(l)$ must be incident on four of the h_k in R^5, and must therefore lie on an edge of the polytope $\bigcap_k h_k^+$. Thus, we can find all extremal stabbing lines of a given polygon sequence merely by examining the edges of the polytope for intersections with the Plücker surface. Each such intersection is the dual of a primal stabbing line, which can be computed in constant time from the four generator edges [27]. Figure 7 depicts the output of an implementation of this algorithm. The input consists of three polygons: a square, a hexagon, and a triangle (thus $n = 13$). The convex hull in R^5 has 130 edges, which yield 36 intersections with the Plücker surface, and thus 36 extremal stabbing lines. All stabbing lines, considered as half-rays beginning at the plane of the final, must lie within the antipenumbra cast by the light source (here, the first polygon in the sequence). Note that some extremal stabbing lines lie in the interior of the antipenumbra. This is because the extremal stabbing lines correspond to part of the face structure of a five-dimensional polytope; when this structure is "projected" (via $\mathcal{L}$) into three space, the projected polytope structure overlaps itself, just as the edge graph of a regular 3-D polygon does when projected onto a plane.

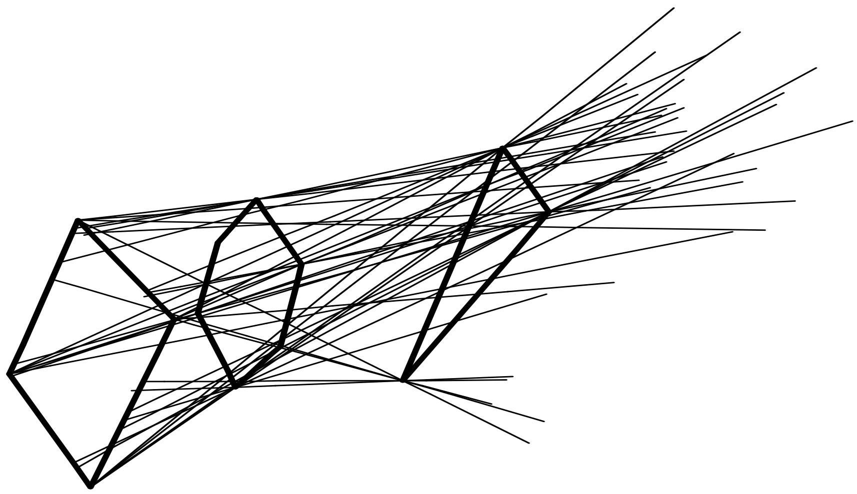

Figure 7: The thirty-six extremal stabbers of three oriented polygons ($n = 13$).

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 defining the line (Figure 8). The surface, or *swath*, swept out by the sliding line must be either planar (if the line remains tight on a vertex) or a regulus, whose generator lines are the three lines affine to the three constraint edges (in the terminology of [22], swaths are VE or EEE "event surfaces").

Figure 8-i depicts an extremal stabber tight on four edges A,B,C and D. Relaxing constraint C yields a VE swath (plane) tight on A, B, and D. Eventually, the sliding line encounters an obstacle (in this case, edge E), and ends at a 4E line (tight on A,B,D, and E). Figure 8-ii depicts a 4E stabber tight on the mutually skew edges A,B,C, and D; relaxing the constraint due to edge A produces a EEE swath (regulus) tight on B,

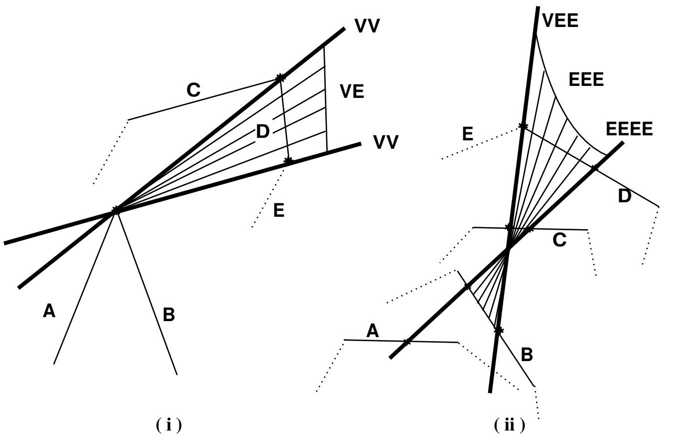

Figure 8: Sliding a stabbing line away from an extremal line.

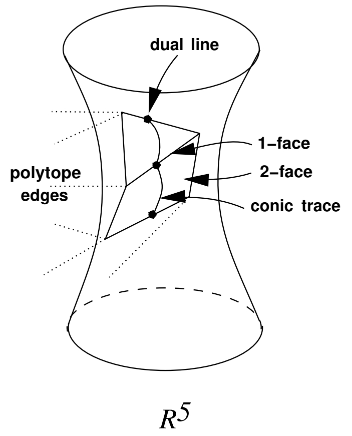

Figure 9: Traces of extremal lines and swaths on the Plücker surface.

C, and D. The sliding line eventually encounters edge E, inducing an extremal VEE line and ending the swath.

Consider the same situations in the dual space generated by the Plücker mapping (Figure 9). Extremal lines map to points in Plücker coordinates; recall that these points are intersections of the edges, or 1-faces, of the polytope $\bigcap_k h_k^+$ with the Plücker surface. Since swaths are one-parameter line families, they dualize to *curves* in Plücker coordinates. These curves are the intersections of the (planar) 2-faces of $\bigcap_k h_k^+$ with the Plücker surface (a convex polytope's 2-faces are those of its faces affine to the intersection of some three h_k), and are therefore conics. We call the primal swaths in R^3 corresponding to these dual conic traces *extremal swaths*, and the three primal edges that contribute to the swath its *generator edges*.

5 Internal and Penumbral Swaths

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. All other extremal swaths are *internal*. Examining the 2-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 determinine whether stabbing lines can potentially exist on "both sides" of the swath in question. If so, the swath cannot be penumbral, and is marked internal.

5.1 Edge-Edge-Edge Swaths

Internal and penumbral EEE swaths can be distinguished as follows. Suppose three mutually skew edges A, B, and C give rise to an extremal EEE swath. Choose some line $\mathbf{L}$ incident on the generators respectively at points $\mathbf{a}$, $\mathbf{b}$, and $\mathbf{c}$ (Figure 10-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 sign chosen so as to have a positive dot product with a vector pointing into the interior of the edge's generator polygon. (For brevity, we refer to these vectors collectively as the $\mathbf{N_i}$.)

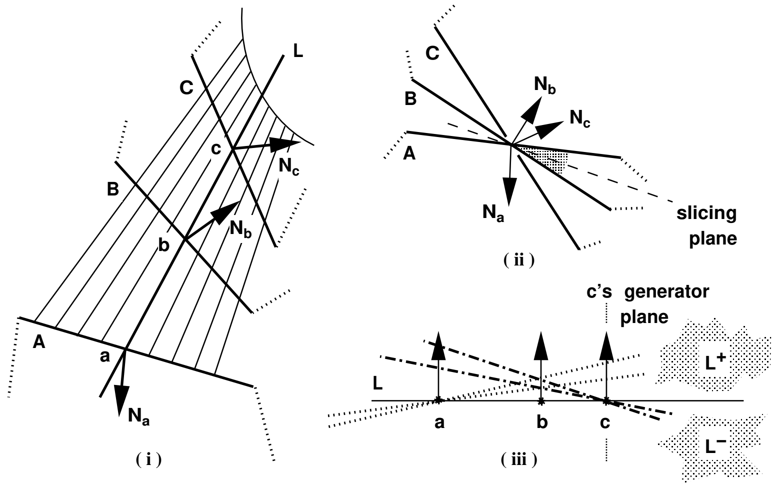

Figure 10: An internal EEE swath (i), viewed along $\mathbf{L}$ (ii) and transverse to $\mathbf{L}$ (iii).

We say that three coplanar vectors can be *contained* if there exists some other vector whose dot product with all three vectors is strictly positive. We claim that the swath is internal iff the $\mathbf{N_i}$ can be contained.

Suppose that the $\mathbf{N_i}$ can be contained (as in Figure 10). Consider any vector pointing into the gray region of Figure 10-ii (this vector has a positive dot product with the $\mathbf{N_i}$). We "slice" the configuration with a plane containing both $\mathbf{L}$ and this vector, and view the configuration in this plane (Figure 10-iii).

The trace of the swath on this plane is simply the stabbing line $\mathbf{L}$. As part of the original swath, $\mathbf{L}$ partitions the slicing plane into two regions $\mathbf{L^+}$ and $\mathbf{L^-}$ beyond the plane of C's generator polygon (Figure 10-iii). We can move $\mathbf{L}$ incrementally by keeping it tight on, say, point $\mathbf{a}$. The interiors of the other two holes allow $\mathbf{L}$ to pivot in only one direction (the dotted lines), thus generating stabbing lines into $\mathbf{L^+}$. Likewise, pivoting about point $\mathbf{c}$ generates stabbing lines into $\mathbf{L^-}$ (the dashed lines). Therefore, this swath can not be penumbral.

By contrast, the $\mathbf{N_i}$ of Figure 11-i can not be contained. Thus, any vector (including one chosen from the gray region of Figure 11-ii) will have a negative dot product with at least one, and at most two, of the $\mathbf{N_i}$. Suppose it has a negative dot product with $\mathbf{N_c}$. Again, slice the configuration with a plane containing both $\mathbf{L}$ and one such vector

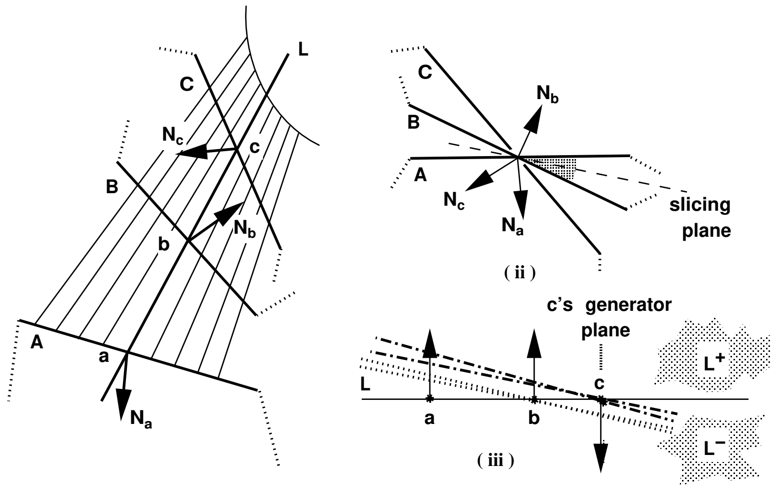

Figure 11: A penumbral EEE swath (i), seen along **L** (ii) and transverse to **L** (iii).

(Figure 11-iii). Pivoting on point **b** generates stabbing lines into $\mathbf{L}^-$, as does pivoting on point **c**. The configuration does not allow 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 the line **L**, we conclude that the swath is penumbral.

5.2 Vertex-Edge Swaths

Suppose the swath in question has type VE. Label the two generator edges defining the swath vertex as A and B, the remaining edge C (skew to both A and B), and the two relevant generator polygons P and Q (Figure 12). Orient the plane ABC so that Q (C's generator polygon) is above it; this is always possible, since Q is convex. Plane ABC divides the halfspace beyond the plane of Q into two regions; call them ABC^+ and ABC^-. If stabbing lines can exist in only one of those regions, the swath is penumbral. This occurs if and only if plane ABC is a *separating plane* of P and Q; that is, iff polygon P is entirely above plane ABC.

To see why this is so, imagine choosing some stabbing line from the swath, and moving it so that it comes free from the swath vertex, but remains tight on edge C (and remains a valid, though non-extremal, stabbing line). If plane ABC separates P and Q, the moving line's intersection with P can only lie below plane ABC (Figure 12-i); thus, beyond the plane of Q, all such stabbing lines must lie in ABC^+.

Suppose, in contrast, that plane ABC does not separate P and Q. The edges A and B must therefore lie in different halfspaces of the plane ABC (Figure 12-ii). Again move a swath line so that it comes free from the swath vertex, but stays tight on edge C. If the line moves along A above (say) plane ABC, it will intersect the region ABC^- beyond the plane of Q. Likewise, motion along edge B produces stabbing lines in ABC^+. Thus plane ABC cannot be penumbral.

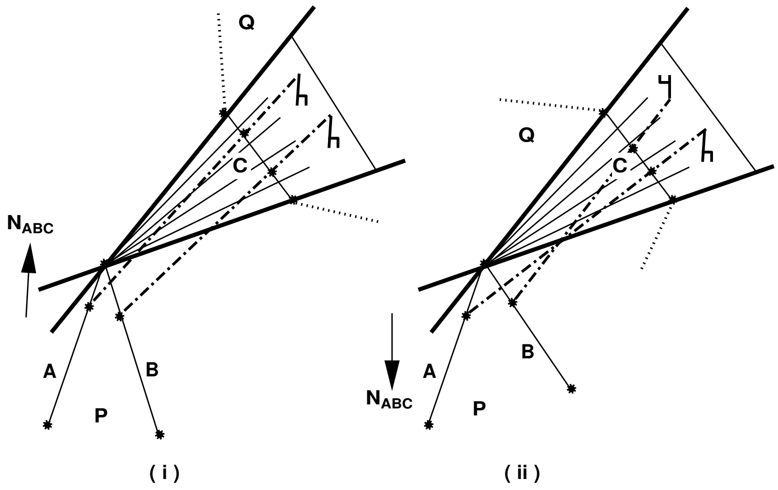

Figure 12: Penumbral (i) and internal (ii) VE swaths.

Note that the three-vector construction for EEE swaths applies to this (degenerate) setting as well, since plane ABC is a separating plane if and only if the normals erected along A,B, and C can not be contained.

6 The Containment Function

The containment function is a criterion for distinguishing between penumbral and internal swaths. However, it applies only to a single stabbing line, not to an entire swath. Fortunately, evaluating the containment function at any line along a swath produces the same result. To see why this is so, consider a configuration of three coplanar vectors $\mathbf{N}_A$, $\mathbf{N}_B$, and $\mathbf{N}_C$, changing continuously from containable to non-containable (e.g., by rotation of vector $\mathbf{N}_C$), as in Figure 13. At the moment of transition (labeled "3" in the middle of the figure), $\mathbf{N}_C$ and one of $\mathbf{N}_A$ or $\mathbf{N}_B$ must be parallel or antiparallel.

For this to occur in the EEE swath construction (Figure 11), two of the generator edges A, B, and C must be coplanar. Similarly, in the VE construction (Figure 12), edge C and one of edges A and B must be coplanar. Neither of these cases occur under our non-degeneracy assumption (when they do occur in practice we detect them in constant time and use special-case processing to generate the correct swath). Thus, even though sliding the stabbing line along the swath generates a continuously changing vector configuration, containment of the configuration is a constant function.

7 The Algorithm

We have assembled all of the computational machinery necessary for generating a description of the antipenumbra. We make the following assumptions: the input is a list

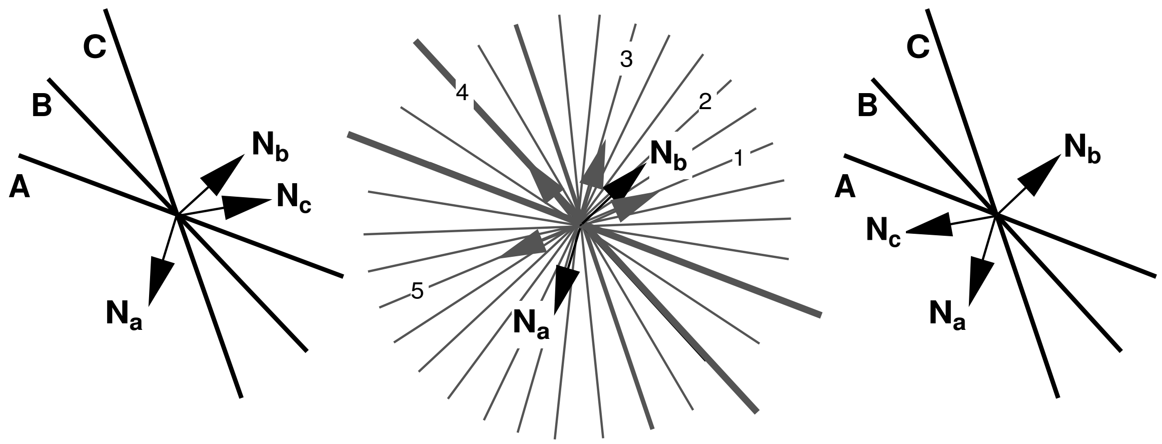

Figure 13: Change from containment to non-containment (by rotation of $\mathbf{N}_c$).

of m oriented polygons, $P_0 \ldots P_{m-1}$, given as linked lists of edges, the total number of edges being n. The zeroth polygon is the light source, and all others are holes. The polygons are ordered in the sense that one halfspace bounded by the plane of P_k contains all polygons $P_j, k < j < m$.

The algorithm dualizes each directed input edge to a hyperplane in Plücker coordinates, then computes the common intersection of the hyperplanes, a convex polytope (if there is no such intersection, or if the polytope has no intersection with the Plücker surface, the penumbra is vacuous). The face structure of the polytope is then searched for closed loops resulting from intersections of its 1-faces and 2-faces with the Plücker surface (there may be more than one loop, because the intersection of the polytope boundary with the (non-convex) Plücker surface may have many components).

Each loop consists of polytope 1-faces (edges) and 2-faces in alternation. Each 1-face intersection with the Plücker surface is the dual of a 4E extremal stabbing line in R^3; each 2-face (conic) intersection with the Plücker surface is the dual of a swath in R^3. Incidence of some 1-face and 2-face on the polytope implies adjacency of the corresponding line and swath in R^3; stepping across a 1-face from one 2-face to another on the polytope is equivalent to stepping across a shared extremal stabbing line between two extremal swaths in three-space. Thus the algorithm can "walk" from penumbral 2-face to penumbral 2-face on the polytope's surface, crossing the 1-face incident to both at each step. Each closed loop found in this manner in R^5 is the dual (under the Plücker mapping) of the boundary of one connected component of the antipenumbra in R^3.

The algorithm can be described as:

1. input the directed edges E_k from the polygons $P_0 \ldots P_{m-1}$
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. compute the convex polytope $\bigcap_k h_k^+$

5. mark all 2-faces of $\bigcap_k h_k^+$ as penumbral or internal
6. for each connected component of the antipenumbra
7. walk polytope structure from penumbral 2-face to 2-face
8. map each dual 2-face found to a primal swath in R^3
9. output primal swath loop as piecewise quadratic surface
10. end while

8 Implementation

We have implemented this algorithm in C and Fortran-77 on a 20 MIP Silicon Graphics 320 GTXB superworkstation. To compute the convex hull of n hyperplanes in R^5, we used a modified version of d-dimensional Delaunay simplicialization code supplied by Allan Wilks and Allen McIntosh at AT&T Bell Labs [30], and a linear-time d-dimensional linear programming algorithm [24] implemented by Michael Hohmeyer at U.C. Berkeley.

Figure 14-a depicts a set of four input polygons with $n = 18$, and the zeroth (leftmost) polygon acting as a light source. The antipenumbra computation took about 2 CPU seconds. The polytope $\bigcap_k h_k^+$, formed from 18 halfspaces in R^5, has 140 facets, 350 1-faces (edges), and 312 2-faces. Of the 350 edges, 67 intersect the Plücker surface to yield primal 4E stabbing lines (Figure 14-b). Of the 312 2-faces, 67 induce dual traces on the Plücker surface, thus corresponding to 67 primal swaths in R^3 (Figure 14-c). Of these 67 swaths, 48 are internal, and 19 are penumbral; of these 19, 10 are VE swaths (red), and 9 are EEE swaths (green). Note that the extremal stabbing lines form the "breakpoints" of the antipenumbra boundary, as they demarcate junctions between adjacent VE or EEE swaths (Figure 14-d).

Figure 15 depicts the internal swaths induced by the same configuration. The swaths have been intersected with a "receiver plane" beyond the plane of the final hole; the resulting line and conic segments are the curves of illumination discontinuity in a mesh-based radiosity computation [14]. An observer stepping across such a curve would see a *qualitative* change in an occlusion characteristic of the light source or an intervening hole, in that an edge or vertex would become occluded or unoccluded.

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 16-a). A third slit is then added that partitions the fattened regulus into two disconnected pieces (Figure 16-b).

color plate goes here

(a) A light source and three holes ($n = 18$).

color plate goes here

(b) The 67 extremal stabbing lines from (a).

color plate goes here

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

color plate goes here

(d) Extremal stabbing lines as "breakpoints."

Figure 14: Various views of the antipenumbra of 4 input polygons ($n = 18$).

color plate goes here

color plate goes here

(a)Discontinuity surfaces from Fig. 14, intersected with a receiver plane.

(a) The extremal stabbing lines form internal "breakpoints" as well.

Figure 15: Discontinuity surfaces from Figure 14, intersected with a receiver plane.

color plate goes here

color plate goes here

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

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

Figure 16: An area light source and three holes can yield a disconnected antipenumbra.

Conclusion

We presented an $O(n^2)$ time algorithm that computes the antipenumbra cast by a convex light source through a series of convex holes, with total edge complexity n. We represented the antipenumbra as a disconnected volume bounded by portions of quadric and planar surfaces, and showed that each swath, or portion of the primal boundary, arises as the primal of a dual conic curve on the trace of a five-dimensional convex hull with the Plücker quadric. The algorithm is related to the aspect graph approach, in that it generates surfaces that separate qualitatively distinct regions (e.g., those that can't see the light source from those that can).

We demonstrated an implementation of the antipenumbra computation on some realistic polygon sequences. The computation was motivated by a visibility scheme in three dimensions; however, we note that knowledge of the antipenumbra might prove useful in shadowing algorithms and radiosity meshing schemes.

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, enthusiasm, and intellectual challenges. The late Professor Rene de Vogelaere, whom I shall miss immensely, taught me about regulae and classical geometry. Ziv Gigus generously gave me the benefit of his careful and thoughtful comments. Raimund Seidel and Michael Hohmeyer also contributed helpful insight and comments. Allan Wilks and Allen McIntosh allowed me to use an implementation of their n-dimensional Delaunay simplicialization code. Silicon Graphics afforded me free run of their considerable physical resources. Lastly, Paul Haeberli provided aesthetic advice, and helped prepare the color plates for submission.

References

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

[2] 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.

[3] A.T. Campbell III and Donald S. Fussell. Adaptive mesh generation for global diffuse illumination. *Computer Graphics (Proc. SIGGRAPH '90)*, 24(4):155–164, 1990.

[4] 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.

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

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

[7] Ernst R.G. Eckert and Robert M. Drake, Jr. *Heat and Mass Transfer*. McGraw-Hill, 1959.

[8] 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.

[9] Benjamin Gebhart. *Heat Transfer*. McGraw-Hill, 1961.

[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] Cindy M. Goral, Kenneth E. Torrance, Donald P. Greenberg, and Bennett Battaile. Modeling the interaction of light between diffuse surfaces. *Computer Graphics (Proc. SIGGRAPH '84)*, 18(3):213–222, 1984.

[12] Branko Grünbaum. *Convex Polytopes*. Wiley-Interscience, New York, 1967.

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

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

[15] Hoyt C. Hottel. *Radiant Heat Transmission, 3rd ed.* Heat Transmission, edited by W.H. McAdams. McGraw-Hill, 1954.

[16] 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.

[17] 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.

[18] Tomoyuki Nishita, I. Okamura, and Eihachiro Nakamae. Shading models for point and linear light sources. *ACM Transactions on Graphics*, 4(2):124–146, 1985.

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

[20] 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.

[21] 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.

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

[23] W.H. Plantinga, C.R. Dyer, and W.B Seales. Visibility, occlusion, and the aspect graph. Technical Report 736, University of Wisconsin at Madison, December 1987.

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

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

[26] Seth J. Teller. Computing the antiumbra cast by an area light source. Technical Report (in preparation), Computer Science Department, U.C. Berkeley, 1991.

[27] 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.

[28] 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.

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

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