CONSTRUCTION AND DISPLAY ALGORITHMS FOR THE ASP

by

Harry Plantinga

and

Charles R. Dyer

Computer Sciences Technical Report #735

December 1987

Construction and Display Algorithms for the Asp

Harry Plantinga - Charles Dyer

Department of Computer Sciences
University of Wisconsin - Madison

Abstract

The *aspect representation* or *asp* is a continuous, viewer-centered representation for polyhedra introduced by Plantinga and Dyer [1987b]. In this paper we present in detail algorithms for working with the asp under orthographic projection. We derive equations for aspect surfaces, show how to represent asps, and present a detailed algorithm for their construction. We then show how to display an image of the represented object from any viewpoint with hidden surfaces removed by finding a cross-section of the asp for a given viewpoint. We present separate algorithms for the convex and non-convex cases.

This work was supported in part by the NSF under grant DCR-8520870.

1. Introduction

The *aspect representation* or *asp* is a continuous, viewer-centered representation for polyhedral objects, introduced by Plantinga and Dyer [1987b]. It is viewer-centered in the sense that it represents the appearance of an object, and it is continuous in the sense that it represents appearance from all viewpoints, or as a continuous function of viewpoint. The asp for a polyehedron is defined as the volume of aspect space that the object fills, where aspect space is viewpoint space cross the image plane. The asp has been use for various applications such as object recognition and constructing the aspect graph [Plantinga and Dyer, 1986; 1987a; 1987c; 1987d]. In this paper we present detailed algorithms for constructing the asp for a polyhedron and for displaying a cross-section of the asp from a given viewpoint. The algorithms for the convex case have been implemented, and the algorithms for the non-convex case are in the process of being implemented.

We assume familiarity with the idea of the aspect representation and with the terminology of the asp; for an introduction to the asp and a discussion of its properties, see [Plantinga and Dyer, 1987b]. This paper is intended as a follow-on to that one, not as a paper that stands on its own.

In this paper the asp is constructed under orthographic projection. The image and the world are assumed to have a fixed coordinate system, with the object or scene of interest centered on the origin in the world. The object or objects represented are assumed to be in front of the viewer from all viewpoints.

We first consider the convex case. In Section 2 we examine the surfaces that bound the asp for a convex polyhedron and show how to construct and represent the faces of an asp. We show how to use that to construct the asp for the polyhedron. In Section 3 we show how to find a cross-section of the asp for a particular viewpoint. The cross-section is an image from that viewpoint with hidden lines removed, i.e. a "line-drawing image" of the polyhedron.

We then consider the non-convex case. In Section 4 we show how to construct the asp for a non-convex polyhedron. In Section 4.1 we derive the surfaces bounding the visibility of a non-convex polyhedron and show how to represent the surfaces and faces on the surfaces. In Section 4.2 we show how to find the intersection of asps, a procedure necessary in constructing the asp for a non-convex polyhedron. In Section

4.3 we show how to use the intersection algorithm to construct the asp for a polyhedron. In Section 5 we describe how to find the cross-section of the asp at a fixed viewpoint for a non-convex polyhedron.

2. Constructing the Asp for a Convex Polyhedron

In this section we present an algorithm for constructing the asp for a convex polyhedron. The algorithm is presented in a way that makes the generalization to the non-convex case as simple as possible. Thus it may be possible to simplify parts of the algorithm, but not in a way that generalizes to the non-convex case.

The asp for a convex polyhedron is the union of the cells of aspect space in which each face of the polyhedron is visible. In this discussion we will call the faces of the asp *vertices*, *edges*, *ridges*, *facets*, and *cells*, according to whether they are 0-, 1-, 2-, 3-, or 4-dimensional. The asp for a polygon in R^3 is a 4-dimensional cell of aspect space; the asp for an edge of the object is a facet, and the asp for a vertex of the object is a ridge. The asp for a polyhedron is the union of the asps for its faces.

Since the polyhedron is convex, each of its faces is visible from all viewpoints in front of the plane containing that face. These asp cells correspond 1-1 with the faces of the polyhedron, and the asp cell for a face is just the image space extent of the face for every viewpoint in front of the plane containing the face. An asp cell is bounded by asp facets, one for each edge bounding the face. Since an edge is on two faces, there are two facets in the asp for every edge of the polyhedron. The facets are 3-faces, and they intersect in 2-faces or asp ridges. There is an asp ridge connecting every pair of asp facets, so there is a ridge for each vertex of every face. Also, there is a ridge for each vertex-face incidence; the total number of ridges is less than three times the number of vertices by Euler's formula for a polyhedron.

The asp edges and vertices do not correspond to any single part of the polyhedron; rather, they correspond to *visual events* bounding the visibility of the face of the polyhedron. The visual event corresponding to a vertex of the asp is the event of two vertices of the object appearing at the same point in the image. The asp vertex corresponds to the viewpoint and image point at which the event occurs. The visual event corresponding to an edge of the asp is a line or curve of viewpoints where an edge and a vertex of the object appear to intersect in the image. An edge of the asp is a curve of points in aspect space, corresponding to the viewpoints and image points where the edge and the vertex appear to intersect. These visual events bound the visibility of object edges, vertices, faces, and so on. Therefore they bound the regions of aspect space occupied by asp facets, ridges, etc.

2.1. Finding the Faces Bounding a Cell in an Asp

The asp for a face of a convex polyhedron is a cell of aspect space. It is bounded by facets and ridges corresponding to the edges and vertices of the face. The ridges are bounded by the visual events bounding the visibility of the face, i.e. the asp vertices and asp edges corresponding to vertex-vertex and vertex-edge pairs of the face. Figure 1 depicts the visual events in the plane containing a face of a polyhedron. The pentagon in Figure 1 should be interpreted as a face of a polyhedron, and the visual events are in the plane containing that face.

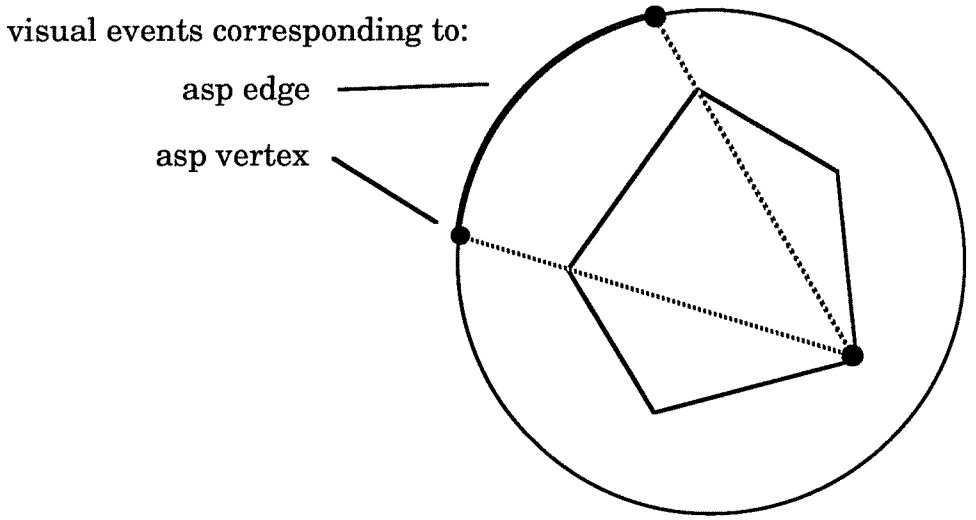

Figure 1. Visual events corresponding to asp vertices and edges for the face of a convex polyhedron. The dotted lines represent viewing directions of visual events, and the vertices and edge on the circle represent corresponding asp faces.

Given a ridge in the asp that corresponds to a vertex of a face, the asp vertices and edges bounding that ridge correspond to the visual events bounding the visibility of that vertex on that face. The face disappears from view whenever the viewpoint crosses from above the plane of the face to below the plane. Thus the asp edges and vertices bounding the ridge are given by the visual events of the sort in Figure 1 for every other edge and vertex of the face. The vertex is visible from other viewpoints below the plane containing the face as well, but we are constructing the asp for a face, and the vertex *as a part of the face* is only visible from viewpoints above the plane of the face.

However, since the object is convex and the face is visible from all viewpoints above the plane of the face (or all viewing directions pointing toward the front of the face), the asp edges and vertices that form the boundary for a ridge will always be a chain of edges and vertices forming a great circle in a plane parallel to the face. That chain represents the fact that the face is visible from all viewpoints above the plane containing it. Thus, it is simpler to set the boundary of visibility of a ridge to the

great circle directly, without computing the individual edges that comprise the great circle. The great circle can be represented as a single asp edge without any bounding vertices, but it reduces the number of special cases in the algorithms below to represent the great circle as a pair of asp edges between two arbitrary points on the circle.

2.2. Representing Asp Vertices

In this paper we use orthographic projection and take viewpoint space to be the 2-D spherical space of viewing directions as shown in Figure 2. (In Figure 2, a rotation with positive θ and negative ϕ is shown.) Aspect space has an image plane for each point of viewpoint space. Note that when $\phi = \pm 90°$ the orientation of the image plane is ambiguous—$(\theta,\phi) = (0°,90°)$ and $(\theta,\phi) = (30°,90°)$ represent the same viewpoint, but the implied orientation of the image plane is different. We define the orientation of the image plane when $\phi = \pm 90°$ to be the orientation that would be reached as ϕ approaches 90° and $\theta = 0°$.

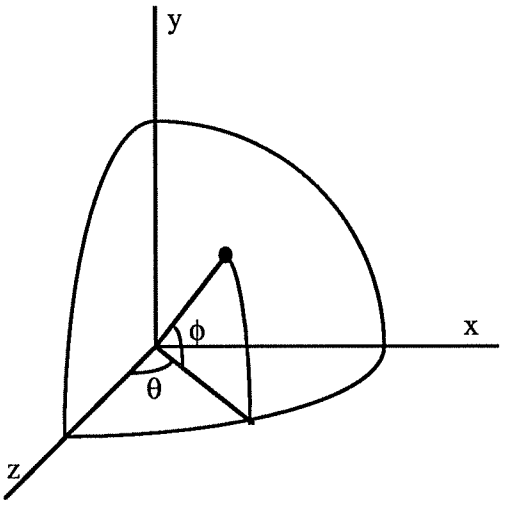

Figure 2. Viewpoint space.

We first show how to find the vertex of aspect space corresponding to the visual event in which two vertices of the object appear at the same point in an image. For a point (x_0,y_0,z_0) of object space, the corresponding 2-surface of aspect space is given by

$$[x_0,y_0,z_0]\begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}\begin{bmatrix} \text{Project} \\ \text{onto} \\ \text{z=0} \\ \text{plane} \end{bmatrix} =$$

$$[x_0 \cos\theta - z_0 \sin\theta,\ x_0 \sin\theta \sin\phi + y_0 \cos\phi + z_0 \cos\theta \sin\phi,\ 0]$$

Therefore the equations of the asp for a point are

$$u = x_0 \cos\theta - z_0 \sin\theta \tag{1}$$

$$v = x_0 \sin\theta \sin\phi + y_0 \cos\phi + z_0 \cos\theta \sin\phi \tag{2}$$

Two vertices of a polyhedron can appear at the same point in an image from two viewing directions that are antipodal or opposite. Only one of those points can be visible from a particular viewing direction since it is "in front of" the other point. We can calculate the viewing direction (and, using Eqs. (1) and (2) above, the point of aspect space) from which this happens in the following way:

Take two copies of Eqs. (1) and (2), one pair each for the points (x_1,y_1,z_1) and (x_2,y_2,z_2). Since the edges appear to intersect, they appear at a single point in the image, (u,v). Thus we can set the equations equal. The results are

$$x_1 \cos\theta - z_1 \sin\theta = x_2 \cos\theta - z_2 \sin\theta \tag{3}$$

$$x_1 \sin\theta \sin\phi + y_1 \cos\phi + z_1 \cos\theta \sin\phi = x_2 \sin\theta \sin\phi + y_2 \cos\phi + z_2 \cos\theta \sin\phi \tag{4}$$

Solving (3) for θ we get

$$\tan\theta = \frac{x_2 - x_1}{z_2 - z_1} \tag{5}$$

or equivalently

$$\sin\theta = \frac{x_2 - x_1}{\sqrt{(x_2 - x_1)^2 + (z_2 - z_1)^2}} \tag{6}$$

$$\cos\theta = \frac{z_2 - z_1}{\sqrt{(x_2 - x_1)^2 + (z_2 - z_1)^2}} \tag{7}$$

Solving Eq. (4) for ϕ and substituting in (6) and (7) yields

$$\tan\phi = \frac{-(y_2 - y_1)\cos\theta}{z_2 - z_1} \tag{8}$$

Therefore Eqs. (5) and (8) give one of the viewing directions in which the object points appear at the same image point, and the other is the antipodal or diametrically opposite point. For two points $\mathbf{P}_1$ and $\mathbf{P}_2$ on a polyhedron, the viewing direction where $\mathbf{P}_2$ appears to be in front of $\mathbf{P}_1$ is the one such that the dot product of a unit vector in the direction (θ,ϕ) and the vector from $\mathbf{P}_2$ to $\mathbf{P}_1$ is greater than zero. From the other viewing direction, $\mathbf{P}_1$ appears to be in front of $\mathbf{P}_2$.

The point of the image plane where this occurs is found by solving Eqs. (5) and (8) for (θ,ϕ) and plugging that into Eqs. (1) and (2). For a convex polyhedron, vertices can thus be represented by their four coordinates in aspect space.

2.3. Representing Asp Edges

We next consider the asp edge corresponding to the visual event generated by a vertex (x_2,y_2,z_2) and an object edge between vertices (x_1,y_1,z_1) and $(x_1+a_1,y_1+b_1,z_1+c_1)$. We can construct the asp edge for a vertex-edge visual event in a similar manner to that for an asp vertex. Instead of two copies of Eqs. (1) and (2), however, we have Eqs. (1) and (2) for the vertex and Eqs. (9) and (10) (below) for the edge :

$$u = (x_1 + s\,a_1)\cos\theta - (z_1 + s\,c_1)\sin\theta \tag{9}$$

$$v = (x_1 + s\,a_1)\sin\theta\sin\phi + (y_1 + s\,b_1)\cos\phi + (z_1 + s\,c_1)\cos\theta\sin\phi \tag{10}$$

where $0 \le s \le 1$. Setting (1) = (9) gives

$$x_2\cos\theta - z_2\sin\theta = (x_1 + s\,a_1)\cos\theta - (z_1 + s\,c_1)\sin\theta \tag{11}$$

Solving this for θ yields

$$\tan\theta = \frac{x_2 - x_1 - s\,a_1}{z_2 - z_1 - s\,c_1} \tag{12}$$

Similarly, we get the following equation for ϕ:

$$\tan\phi = \frac{-(y_2 - y_1 - s\,b_1)\cos\theta}{z_2 - z_1 - s\,c_1} \tag{13}$$

Comparing Eqs. (12) and (13) with (5) and (8) make it is apparent that the equations for an asp edge can be obtained by substituting

$(x_2 - x_1 - s\ a_1)$ for x_1,
$(y_2 - y_1 - s\ b_1)$ for y_1, and
$(z_2 - z_1 - s\ c_1)$ for z_1

in the equations for an asp vertex.

Eqs. (12) and (13) define the viewpoints associated with the asp edge as a function of the parameter s, for $0 \le s \le 1$. The corresponding points of the image plane are given by Eqs. (9) and (10). Notice that the constants used in the four equations are the coordinates of the three object vertices that are involved in the visual event—the vertex and the two endpoints of the object edge. Therefore the asp edge is represented by the equation of the curve that it lies on (the coordinates of the three points that generated the visual event, or pointers to the object vertex and edge) and pointers to the asp vertices bounding the edge. Note that the bounding asp vertices are also given implicitly by Eqs. (9), (10), (12), and (13) above when $s=0$ and $s=1$.

3. Constructing an Image from the Asp

Given a viewpoint (θ,ϕ), the (θ,ϕ)–cross-section of the asp is the view of the polyhedron from that viewpoint. Thus drawing the (θ,ϕ)–cross-section of the asp is equivalent to constructing an image of the polyhedron from that viewpoint with hidden lines removed. In this section we will show how to find that cross-section of the asp.

Each cell of the asp corresponds to a face of the polyhedron, and the appearance of the face of the polyhedron from a viewpoint (θ,ϕ) is the cross-section of the cell from (θ,ϕ). The cross-section of the cell for a fixed viewpoint is a polygon since faces of polyhedra always appear as polygons in an image. We find this polygon by finding its edges, which are represented as facets of the asp.

The facets of the asp are 3-faces, and they are bounded by 2-faces (ridges) that correspond to the endpoints of the edges generating the facets. If the edge is visible it has two endpoints, so exactly two of the ridges bounding the facet will be visible and will project to the endpoints of the edge. Thus we can tell whether the facet is visible by determining whether it has two visible ridges. If it does, then the intersection points of those ridges and the (θ,ϕ) plane are the endpoints of the visible line segment.

A ridge of the asp lies on a 2-surface of aspect space. If the point of that surface at the viewpoint (θ,ϕ) is in the ridge (i.e. in the region of the surface bounded by the edges that bound the ridge), then the corresponding vertex of the polyhedron is

visible; otherwise it is not visible. Thus we can determine whether the point should be displayed by determining whether the point of the surface at that viewpoint is in the ridge on that surface. Equivalently, we can project the bounding edges and ridges down to viewpoint space and determine whether the viewpoint is in the spherical polygonal region that they bound. We present in detail below the algorithm for determining whether the point is in the spherical polygon (and therefore whether the object vertex is visible).

3.1. Determining Whether an Asp Ridge is Visible

In this section we show in detail how to determine whether an asp ridge is visible. A ridge is bounded by a cycle of asp edges and vertices; the edges are given by Eqs. (9), (10), (12), and (13) above. Notice that Eqs. (12) and (13) do not refer to the image plane. Thus we can ignore Eqs. (9) and (10) for the edges and consider Eqs. (12) and (13) by themselves as the projections of the edges on viewpoint space. Determining whether a ridge is visible then amounts to determining whether a point is in a region bounded by edges of the form given by Eqs. (12) and (13). Since the curves bounding the region are arcs of great circles, the region is a spherical polygon.

The standard algorithm for determining whether a point is in a polygon is to draw a ray from the point infinitely far in some direction and count the number of times it intersects the boundary of the polygon. If it intersects an odd number of times, the point is in the polygon. This algorithm must be modified for the case of a spherical polygon because a ray on a sphere does not go infinitely far in some direction: it goes around the sphere and back to the point. However, we do know that if a ridge is visible from some viewpoint, then a great circle from that viewpoint around viewpoint space and back to the same viewpoint will first cross the boundary *going out* of the spherical polygon if it crosses the boundary at all. Conversely, if the viewpoint is not in the spherical polygon, then the great circle will first intersect the boundary *going in* to the spherical polygon if it crosses the boundary at all.

This suggests an algorithm for determining whether a point is in a spherical polygon: pick a great circle containing the viewpoint that intersects the boundary of the spherical polygon, find the intersection with the boundary that is closest to the viewpoint, and determine whether that intersection goes into or out of the polygon.

A great circle is the intersection of a sphere with a plane through the center of the sphere, so we can find a great circle that intersects the boundary by taking the intersection of the sphere with the plane defined by the center of the sphere and two other points: the viewpoint in question and a point on one of the edges of the

spherical polygon. This plane intersects viewpoint space in the required great circle. Representing viewpoint space as a unit sphere centered on the origin, the points defining the plane are then (0,0,0), (θ_1,ϕ_1) (the test viewpoint), and (θ_2,ϕ_2) given by Eqs. (12) and (13) for one of the edges of the polygon, with some s chosen so that $0 < s < 1$. We can use a spherical-to-Cartesian coordinate transformation to get the Cartesian coordinates for the points; the Cartesian coordinates for the viewpoint (θ,ϕ) on the unit sphere is $(\sin\theta\cos\phi, -\sin\phi, \cos\theta\cos\phi)$. A unit normal to the plane is then given by the cross product of vectors to the two points on the sphere, i.e.

$$\mathbf{n}_1 = (\sin\theta_1\cos\phi_1, -\sin\phi_1, \cos\theta_1\cos\phi_1) \times (\sin\theta_2\cos\phi_2, -\sin\phi_2, \cos\theta_2\cos\phi_2) \tag{14}$$

We want to find any intersections of this great circle with each asp edge. The asp edges are generated by an edge and vertex of the polyhedron; let the endpoints of the edge and the vertex be (x_1, y_1, z_1), (x_2, y_2, z_2), and (x_3, y_3, z_3), respectively. Those three points determine another plane, and a normal to the plane is given by

$$\mathbf{n}_2 = (x_1-x_3, y_1-y_3, z_1-z_3) \times (x_2-x_3, y_2-y_3, z_2-z_3) \tag{15}$$

The planes defined by Eqs. (14) and (15) intersect in a line, and the line intersects the sphere of viewpoints in two points. These two points are the two possible intersection points of the asp edge and the great circle. The directions of the points are given by

$$\mathbf{n}_1 \times \mathbf{n}_2 \tag{16}$$

and

$$\mathbf{n}_2 \times \mathbf{n}_1 \tag{17}$$

We can convert these vectors in Cartesian coordinates to viewpoints in the form (θ,ϕ) by a polar-to-spherical transformation where the radius $\mathbf{r} = 1$. For a vector $\mathbf{v}$, the transformation is:

$$\theta = \tan^{-1}(v_x / v_z) \tag{18}$$

$$\phi = \sin^{-1}(-v_y) \tag{19}$$

Thus we can calculate the two possible intersection points of an edge with the great circle. In order to determine whether the edge does intersect the great circle, it remains to solve (12) and (13) for s given the values for θ and ϕ calculated from Eqs. (18) and (19). It is sufficient to use the value of θ from (18) and solve (12) for s, yielding

$$s = \frac{(z_2 - z_1)\, v_x - (x_2 - x_1)\, v_z}{c_1\, v_x - a_1\, v_z} \tag{20}$$

If $0 \le s \le 1$, then the edge and the great circle intersect.

As we find all of the intersection points of edges with the great circle, we can calculate their distances from the test viewpoint and keep track of the closest intersection point. We can use the Cartesian distance or the angle between the two points on the unit sphere.

Once we have found the closest intersection point, we can determine whether the viewpoint is in the spherical polygon by computing the sense of the intersection. That is, we compute whether the great circle is going out of the polygon or in to the polygon. To do so, we assume that the edges of the spherical polygon are listed in a particular order, say clockwise (looking from the center of the sphere). Then the triple cross product of the vectors from the test viewpoint to the intersection point, from the intersection point clockwise along the boundary of the spherical polygon, and from the center of the sphere to the intersection point, determines whether the point is inside the spherical polygon: if the triple cross product is negative then the great circle is going out of the polygon so the test viewpoint must have been inside the polygon. If the triple cross product is positive then the viewpoint must have been outside the polygon. If the test viewpoint is $\mathbf{v} = (v_x, v_y, v_z)$, the intersection point is $\mathbf{i} = (i_x, i_y, i_z)$, and the edge of the spherical polygon that the intersection point is on has direction $\mathbf{d} = (d_x, d_y, d_z)$, then the triple cross product is given by

$$(\mathbf{i} - \mathbf{v}) \times \mathbf{d} \bullet \mathbf{i} = \begin{vmatrix} i_x - v_x & i_y - v_y & i_z - v_z \\ d_x & d_y & d_z \\ i_x & i_y & i_z \end{vmatrix} \tag{21}$$

At this point we have determined whether the test viewpoint is inside the spherical polygon. For each ridge of the asp, we make that test. Then, for each facet, if both of its bounding ridges are visible the whole facet is visible, so we display the corresponding edge. The result of this process is the display of the complete set of visible edges of the polyhedron.

4. Constructing the Asp for Non-convex Polyhedra

The asp for a non-convex polyhedron is the union of the cells of aspect space in which each face of the polyhedron is visible. However, because faces can be partially occluded by other faces, constructing the asp for a non-convex polyhedron is more complicated than for a convex polyhedron. The asp for a face **f** of a polyhedron as

obstructed by the other faces is the asp for **f** minus (in the sense of set subtraction) the asps for the faces or parts of faces in front of **f** [Plantinga and Dyer, 1987].

The subtraction of one asp from another is equivalent to the intersection of the asp with the complement of the other, i.e. $A - B = A \cap \neg B$. Taking the intersection of two asps will introduce new kinds of faces to the asp. For example, in the convex case a ridge of the asp corresponds to a vertex of a face the polyhedron. Another kind of ridge that occurs in the non-convex case corresponds to the *apparent* intersection of two edges of the polyhedron. That is, two edges that in fact do not intersect, may appear to intersect from some viewpoints if one is in front of the other (i.e., a "T-junction"). This visual event is visible from a 2-D range of viewpoints and has 0-D extent in image space, so it is a ridge of the asp. It arises from taking the intersection of two asp facets.

Thus constructing the asp in the non-convex case will require calculating and representing other kinds of asp ridges, edges, and vertices. These new faces arise when finding the intersection of asp surfaces. Facets lie on 3-surfaces, ridges on 2-surfaces, and edges on 1-surfaces. In the next section we derive standard forms for these surfaces and show how to find the intersection of pairs of asp surfaces. In Section 4.2 we show how to find intersections of asps. In Section 4.3 we describe in more detail how to construct the asp in the non-convex case.

4.1. Aspect Surfaces

The aspect representation for a face of a convex polyhedron is a cell of aspect space—a 4-dimensional volume bounded by 3-surfaces. The points of aspect space correspond to points of the image plane occupied by the polygon when viewed from the given viewpoint. The cell is bounded by 3-surfaces, which correspond to the edges bounding the polygon. The 3-surfaces are bounded by 2-surfaces, corresponding to the vertices bounding the polygon. The 2-surfaces are the intersection of two of the three-surfaces. The asp edges and vertices correspond to visual events bounding the visibility of the polygon.

The aspect representation for a polygon that is a face of a non-convex polyhedron is also a cell of aspect space, bounded by 3-surfaces or facets. The facets again correspond to the edges bounding the polygon. Ridges bound the visibility of facets, and in the non-convex case there are ridges corresponding to the vertices bounding the edges of the polygon. However, there is another kind of ridge, corresponding to the case where the visibility of a polygon edge is bounded by another, occluding edge. In that case there is a ridge corresponding the *apparent*

intersection of the two edges. This ridge results from the subtraction of the asp for the occluding polygon from the asp for the given polygon. The surface on which the ridge lies results from the intersection of the surfaces on which the two facets lie. It is the general sort of 2-surface. We derive equations for the general 2-surface in Section 4.1.2.

The visual events that occur in the convex case (the horizon effect) also bound the visibility of a ridge in the non-convex case. However, another kind of visual event that bounds the visibility of a ridge in the non-convex case is the apparent intersection of three unconnected object edges in a single image point. We derive the general equations for such a boundary in Section 4.1.3.

Asp vertices correspond to visual events that are visible from a single viewpoint and that occupy a single point in the image plane. In the convex case, the only sort of such an event is the apparent intersection of two vertices from the same face of a polyhedron. In the non-convex case, the most general visual event is the apparent intersection of four object edges in a single point from some viewpoint. We derive the equations for such a point in Section 4.1.4.

In order to find the intersection of cells of aspect space, we must be able to find the intersection of pairs of asp surfaces of various kinds. Note that *all* of the asp surfaces—3-surfaces, 2-surfaces, 1-surfaces, and vertices—result from the intersection of a set of asp 3-surfaces—one, two, three, and four, respectively. Thus, finding the intersection of an asp 3-surface and an asp 2-surface is equivalent to finding the intersection of the three "parent" asp 3-surfaces involved. Also, since two asp 1-surfaces that lie on the same 2-surface have two common 3-surface "parents," finding the intersection of two 1-surfaces is equivalent to finding the intersection of the four unique 3-surfaces involved in the problem. Thus finding the intersection of two asp surfaces is equivalent to finding the intersection of the unique "parents" of those surfaces.

In order to determine whether one asp edge is going "into" or "out of" another asp edge (assuming directed edges), we will want to be able to compute a tangent to an asp edge at a point and a normal to that tangent. We show how to do this in Section 4.1.5.

4.1.1. 3-Surfaces of Aspect Space

A 3-surface of aspect space corresponds to the visibility of a line in object space. The equations for such a surface for a line passing through the points $\mathbf{p}_1 = (x_1,y_1,z_1)$ and $\mathbf{p}_1 + \mathbf{a}_1 = (x_1+a_1,y_1+b_1,z_1+c_1)$ were derived earlier as:

$$u = (x_1 + s\, a_1)\cos\theta - (z_1 + s\, c_1)\sin\theta \tag{22}$$

$$v = (x_1 + s\, a_1)\sin\theta\sin\phi + (y_1 + s\, b_1)\cos\phi + (z_1 + s\, c_1)\cos\theta\sin\phi \tag{23}$$

In addition, we need another 3-surface, represented with different constants and a different parameter, so that we may take the intersections of these surfaces to derive the 2-surface equations. This 3-surface represents the visibility of a line through the points $\mathbf{p}_2 = (x_2,y_2,z_2)$ and $\mathbf{p}_2 + \mathbf{a}_2 = (x_2+a_2,y_2+b_2,z_2+c_2)$:

$$u = (x_2 + s_2\, a_2)\cos\theta - (z_2 + s_2\, c_2)\sin\theta \tag{24}$$

$$v = (x_2 + s_2\, a_2)\sin\theta\sin\phi + (y_2 + s_2\, b_2)\cos\phi + (z_2 + s_2\, c_2)\cos\theta\sin\phi \tag{25}$$

4.1.2. 2-Surfaces of Aspect Space

2-surfaces of aspect space arise in the general case as the intersection of two 3-surfaces. This may occur in two ways. As in the convex case, a 2-surface may correspond to a vertex of the object. In that case the 2-surface on which it lies is the intersection of the 3-surfaces corresponding to the object edges that meet at that vertex. The other way that an asp 2-surface can arise is as the intersection of two general 3-surfaces. In that case the resulting 2-surface corresponds to the visual event of the *apparent* intersection of two object edges, i.e. the intersection of two edges in an image, although the edges do not intersect in space. We derive the equations for the latter, more general kind of 2-surface. If the two object edges intersect in a point, the equations derived will simplify to the equations for the simpler case.

Suppose we have two 3-surfaces given by Eqs. (22)-(23) and (24)-(25) above. The intersection of the two 3-surfaces is a 2-surface, which can be found by setting Eqs. (22)=(24) and (23)=(25). Solving Eqs. (22)-(25) for θ and ϕ yields

$$\tan\theta = \frac{(x_2 + s_2\, a_2) - (x_1 + s\, a_1)}{(z_2 + s_2\, c_2) - (z_1 + s\, c_1)} \tag{26}$$

$$\tan \phi = \frac{-[(y_2 + s_2\ b_2) - (y_1 + s\ b_1)] \cos \theta}{(z_2 + s_2\ c_2) - (z_1 + s\ c_1)} \tag{27}$$

Since this is a 2-surface it depends only on two parameters. In Eqs. (26) and (27) (together with Eqs. (22)-(25)) the surface is given as a function of s and s_2. More often, however, we will want u and v as a function of θ and ϕ. Thus, we will solve (26) and (27) for θ and ϕ.

We can simplify the equations by eliminating sinusoidal functions through a change of variables. Instead of θ and ϕ, we can express the equations in terms of a viewpoint $\mathbf{v} = (v_x, v_y, v_z) = (\sin\theta \cos\phi, -\sin\phi, \cos\theta \cos\phi)$, so that the vector from $\mathbf{v}$ to the origin is the viewing direction (θ,ϕ). In other words, we substitute for (θ,ϕ) the unit vector $\mathbf{v}$ in the direction (θ,ϕ). Note that $|\mathbf{v}| = 1$. For Eq. (26) we get

$$\frac{v_x}{v_z} = \frac{(x_2 + s_2\ a_2) - (x_1 + s\ a_1)}{(z_2 + s_2\ c_2) - (z_1 + s\ c_1)} \tag{28}$$

and for (27) we get

$$\frac{v_y}{v_z} = \frac{(y_2 + s_2\ b_2) - (y_1 + s\ b_1)}{(z_2 + s_2\ c_2) - (z_1 + s\ c_1)} \tag{29}$$

Note that we could have arrived at these equations immediately by noting that the viewing direction $\mathbf{v}$ must be parallel to a vector from one of the lines to the other at the points where they appear to intersect, i.e. a vector from $\mathbf{p}_1 + s\ \mathbf{a}_1$ to $\mathbf{p}_2 + s_2\ \mathbf{a}_2$ (see Figure 3).

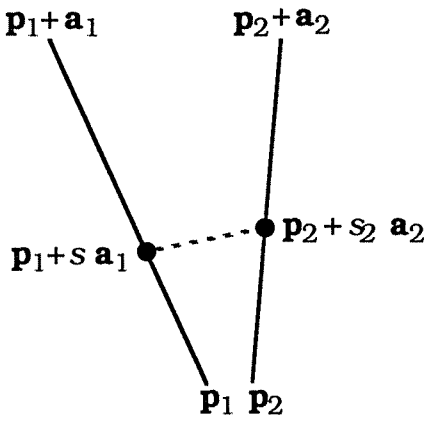

Figure 3. The viewing direction along which two lines appear to intersect

Solving (28) and (29) for s_2 yields

$$s_2 = \frac{v_x (z_1 - z_2 + s\ c_1) - v_z (x_1 - x_2 + s\ a_1)}{v_x c_2 - v_z a_2} \tag{30}$$

$$s_2 = \frac{v_y (z_1 - z_2 + s\, c_1) - v_z (y_1 - y_2 + s\, b_1)}{v_y c_2 - v_z b_2} \tag{31}$$

Setting (30) and (31) equal and solving for s yields

$$s = \frac{v_x (y_{21} c_2 - z_{21} b_2) + v_y (z_{21} a_2 - x_{21} c_2) + v_z (x_{21} b_2 - y_{21} a_2)}{v_x (b_1 c_2 - b_2 c_1) + v_y (c_1 a_2 - c_2 a_1) + v_z (a_1 b_2 - a_2 b_1)}$$

where $x_{ij} = x_i - x_j$, or

$$s = \frac{\mathbf{v} \cdot [(\mathbf{p}_2 - \mathbf{p}_1) \times \mathbf{a}_2]}{\mathbf{v} \cdot (\mathbf{a}_1 \times \mathbf{a}_2)} \tag{32}$$

Eq. (32) can be substituted into Eq. (22)-(23) to find u and v.

The surface can be represented by twelve constants: a_1, a_2, b_1, b_2, c_1, c_2, x_1, x_2, y_1, y_2, z_1, z_2. These are related to the coordinates of the four endpoints of the two lines involved in the visual event. Note that the surface can also be represented by a different set of twelve constants: the six coefficients of the v_x, v_y, and v_z terms in the numerator and denominator together with the other six constants of Eqs. (22)-(23).

Eqs. (22), (23), and (32) represent a 2-surface in aspect space that results from the intersection of two 3-surfaces corresponding to asp facets. This kind of surface did not arise in the convex case. Finally, notice that when the four endpoints of the line segments all lie in the same plane, Eq. (32) simplifies so that the surface is of the kind given by Eqs. (12) and (13).

4.1.3. 1-Surfaces of Aspect Space

The intersection of three 3-surfaces of aspect space is a 1-surface or curve. It corresponds to the visual event of the apparent intersection of three object edges in an image. Three object edges can appear to intersect from a 1-dimensional curve of viewpoints, and their apparent intersection is a single point, so the aspect representation for this visual event is a 1-surface or curve in aspect space.

The intersection of a 3-surface and a 2-surface is also a 1-surface, but this case is equivalent to the former case since the 2-surface is the intersection of two 3-surfaces. Another equivalent intersection problem is the intersection of two 2-surfaces, both of which lie on the same 3-surface, since in that case both 2-surfaces have a 3-surface "parent" in common.

We could find the 1-surface by finding the intersection of three pairs of equations of the form of Eqs. (22)-(23) for three 3-surfaces. However, there is a simpler method. Consider three edges in space, $\mathbf{p}_1$ to $\mathbf{p}_1 + \mathbf{a}_1$, $\mathbf{p}_2$ to $\mathbf{p}_2 + \mathbf{a}_2$, and $\mathbf{p}_3$ to $\mathbf{p}_3 + \mathbf{a}_3$. Pick a point $\mathbf{p}_1 + s\,\mathbf{a}_1$ on one line (see Figure 4) and find a line through that point that intersects both of the other lines.

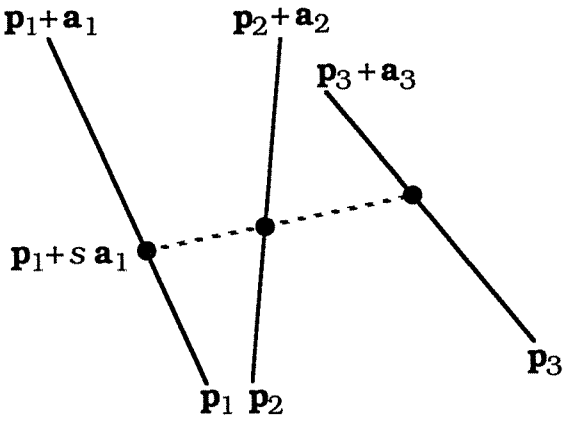

Figure 4. A viewing direction along which three object edges appear to intersect in a single point in an image

Consider the planes defined by the point $\mathbf{p}_1 + s\,\mathbf{a}_1$ each of the other two edges. The viewing direction is parallel to the intersection of these two planes. A normal to the plane defined by the points $\mathbf{p}_1 + s\,\mathbf{a}_1$, $\mathbf{p}_2$, and $\mathbf{p}_2 + \mathbf{a}_2$ is given by

$$(\mathbf{p}_1 + s\,\mathbf{a}_1 - \mathbf{p}_2) \times \mathbf{a}_2$$

and a normal to the plane defined by the points $\mathbf{p}_1 + s\,\mathbf{a}_1$, $\mathbf{p}_3$, and $\mathbf{p}_3 + \mathbf{a}_3$ is given by

$$(\mathbf{p}_1 + s\,\mathbf{a}_1 - \mathbf{p}_3) \times \mathbf{a}_3$$

Therefore a vector parallel to the viewing direction in question (though not a unit vector) is given by

$$\mathbf{v}' = ((\mathbf{p}_1 + s\,\mathbf{a}_1 - \mathbf{p}_2) \times \mathbf{a}_2) \times ((\mathbf{p}_1 + s\,\mathbf{a}_1 - \mathbf{p}_3) \times \mathbf{a}_3) \tag{33}$$

Thus

$$\mathbf{v}' = \langle s\,bc_{12} + bz_{12},\ s\,ca_{12} + cx_{12},\ s\,ab_{12} + ay_{12}\rangle \times \langle s\,bc_{13} + bz_{13},\ s\,ca_{13} + cx_{13},\ s\,ab_{13} + ay_{13}\rangle$$

where $bc_{12} = b_1\,c_2 - b_2\,c_1$, $bz_{12} = b_2\,z_{12} - c_2\,y_{12}$, and so on. Therefore, if $\mathbf{v}' = \langle v_x', v_y', v_z'\rangle$ then we have

$$v_x' = s^2 (ca_{12}\, ab_{13} - ab_{12}\, ca_{13}) + s (ca_{12}\, ay_{13} - ab_{12}\, cx_{13} + cx_{12}\, ab_{13} - ay_{12}\, ca_{13}) + (cx_{12}\, ay_{13} - ay_{12}\, cx_{13}) \tag{34}$$

$$v_y' = s^2 (ab_{12}\, bc_{13} - bc_{12}\, ab_{13}) + s (ab_{12}\, bz_{13} - bc_{12}\, ay_{13} + ay_{12}\, bz_{13} - bz_{12}\, ay_{13}) + (ay_{12}\, bz_{13} - bz_{12}\, ay_{13}) \tag{35}$$

$$v_z' = s^2 (bc_{12}\, ca_{12} - ca_{12}\, bc_{13}) + s (bc_{12}\, cx_{13} - ca_{12}\, bz_{13} + bz_{12}\, ca_{13} - cx_{12}\, bz_{13}) + (bz_{12}\, cx_{13} - cx_{12}\, bz_{13}) \tag{36}$$

Since $\tan\theta = v_x/v_z = v_x'/v_z'$, we can substitute

$$\begin{aligned} e_1 &= ca_{12}\, ab_{13} - ab_{12}\, ca_{13} \\ e_2 &= ca_{12}\, ay_{13} - ab_{12}\, cx_{13} + cx_{12}\, ab_{13} - ay_{12}\, ca_{13} \\ e_3 &= cx_{12}\, ay_{13} - ay_{12}\, cx_{13} \\ e_4 &= bc_{12}\, ca_{12} - ca_{12}\, bc_{13} \\ e_5 &= bc_{12}\, cx_{13} - ca_{12}\, bz_{13} + bz_{12}\, ca_{13} - cx_{12}\, bz_{13} \\ e_6 &= bz_{12}\, cx_{13} - cx_{12}\, bz_{13} \end{aligned}$$

into Eqs. (34) and (36), yielding

$$\tan\theta = \frac{e_1 s^2 + e_2 s + e3}{e_4 s^2 + e_5 s + e_6} \tag{37}$$

We can solve (37) for s in terms of θ, yielding

$$s^2 (e_4 \tan\theta - e_1) + s (e_5 \tan\theta - e_2) + e_6 \tan\theta - e_3 = 0 \tag{38}$$

or

$$s = -\frac{(e_5 \tan\theta - e_2) \pm \sqrt{(e_5 \tan\theta - e_2)^2 - 4(e_4 \tan\theta - e_1)(e_6 \tan\theta - e_3)}}{2 (e_4 \tan\theta - e_1)} \tag{39}$$

Then, ϕ in terms of θ and s is

$$\tan\phi = \frac{-v_y' \cos\theta}{v_x'} \tag{40}$$

Thus, given some θ we get s from Eq. (39), v_x' and v_y' from Eqs. (34) and (35), and ϕ from Eq. (40). We can then get u and v from Eqs. (22) and (23). Therefore,

Eqs. (22), (23), (34), (35), (39), and (40) represent an aspect 1-surface by giving u, v, and ϕ in terms of a quadratic function of θ.

4.1.4. Vertex of Aspect Space

A vertex of aspect space results from the intersection of four asp 3-surfaces, two general 2-surfaces, two 1-surfaces on a 2-surface, or any other combination of surfaces with four unique 3-surface "parents." The visual event that gives rise to an asp vertex is the apparent intersection of four object edges in a single point. This occurs from only one viewpoint, so the corresponding asp surface is a vertex.

We could find the vertex of aspect space corresponding to the apparent intersection of four object edges by finding the intersection of four pairs of equations of the form of Eqs. (22)-(23) for four 3-surfaces. However, there is a simpler method. Consider four lines in space, through $\mathbf{p}_1$ & $\mathbf{p}_1 + \mathbf{a}_1$, $\mathbf{p}_2$ & $\mathbf{p}_2 + \mathbf{a}_2$, $\mathbf{p}_3$ & $\mathbf{p}_3 + \mathbf{a}_3$, and $\mathbf{p}_4$ & $\mathbf{p}_4 + \mathbf{a}_4$. Pick a point $\mathbf{p}_1 + s\,\mathbf{a}_1$ on one line (see Figure 5).

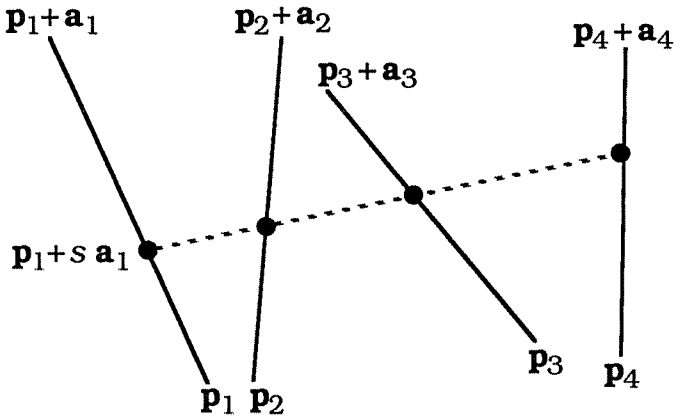

Figure 5. A viewing direction along which four object edges appear to intersect in a single point in an image

A viewing direction through the point $\mathbf{p}_1 + s\,\mathbf{a}_1$ from which the four edges appear to intersect in a point must be parallel to the three planes defined by $\mathbf{p}_1 + s\,\mathbf{a}_1$ and each of the other lines. Therefore if there is such a line of sight passing through $\mathbf{p}_1 + s\,\mathbf{a}_1$, the triple cross product of the normals to the three planes must be zero. The normals to the planes are given by the cross product of two vectors in each plane:

$$(\mathbf{p}_1 + s\,\mathbf{p}_1 - \mathbf{p}_2) \times \mathbf{a}_2$$
$$(\mathbf{p}_1 + s\,\mathbf{p}_1 - \mathbf{p}_3) \times \mathbf{a}_3$$
$$(\mathbf{p}_1 + s\,\mathbf{p}_1 - \mathbf{p}_4) \times \mathbf{a}_4$$

If the three planes intersect in a single line, then the normals have a triple cross product of zero, which in determinant notation is

$$\begin{vmatrix} s\,bc_{12}+bz_{12} & s\,ca_{12}+cx_{12} & s\,ab_{12}+ay_{12} \\ s\,bc_{13}+bz_{13} & s\,ca_{13}+cx_{13} & s\,ab_{13}+ay_{13} \\ s\,bc_{14}+bz_{14} & s\,ca_{14}+cx_{14} & s\,ab_{14}+ay_{14} \end{vmatrix} = 0 \tag{41}$$

In order to find the point of aspect space we must write out (41) in terms of the powers of s. We can then solve for s using the cubic equation.

For the s^3 term only the bc_{ij}, ca_{ij}, and ab_{ij} terms of (41) are involved, so letting a be the coefficient of the s^3 term we get

$$a = \begin{vmatrix} bc_{12} & ca_{12} & ab_{12} \\ bc_{13} & ca_{13} & ab_{13} \\ bc_{14} & ca_{14} & ab_{14} \end{vmatrix} \tag{42}$$

For the s^2 term we must have one row of terms not involving s in each coefficient matrix, so we get

$$b = \begin{vmatrix} bz_{12} & cx_{12} & ay_{12} \\ bc_{13} & ca_{13} & ab_{13} \\ bc_{14} & ca_{14} & ab_{14} \end{vmatrix} + \begin{vmatrix} bc_{12} & ca_{12} & ab_{12} \\ bz_{13} & cx_{13} & ay_{13} \\ bc_{14} & ca_{14} & ab_{14} \end{vmatrix} + \begin{vmatrix} bc_{12} & ca_{12} & ab_{12} \\ bc_{13} & ca_{13} & ab_{13} \\ bz_{14} & cx_{14} & ay_{14} \end{vmatrix} \tag{43}$$

For the s term we have two rows of terms not involving s and one row involving s in each coefficient matrix:

$$c = \begin{vmatrix} bc_{12} & ca_{12} & ab_{12} \\ bz_{13} & cx_{13} & ay_{13} \\ bz_{14} & cx_{14} & ay_{14} \end{vmatrix} + \begin{vmatrix} bz_{12} & cx_{12} & ay_{12} \\ bc_{13} & ca_{13} & ab_{13} \\ bz_{14} & cx_{14} & ay_{14} \end{vmatrix} + \begin{vmatrix} bz_{12} & cx_{12} & ay_{12} \\ bz_{13} & cx_{13} & ay_{13} \\ bc_{14} & ca_{14} & ab_{14} \end{vmatrix} \tag{44}$$

Finally, the term that is constant in s involves only the bz_{ij}, cx_{ij}, and ay_{ij} constants:

$$d = \begin{vmatrix} bz_{12} & cx_{12} & ay_{12} \\ bz_{13} & cx_{13} & ay_{13} \\ bz_{14} & cx_{14} & ay_{14} \end{vmatrix} \tag{45}$$

Then we can solve the cubic equation

$$a\,s^3 + b\,s^2 + c\,s + d = 0 \tag{46}$$

for s. Once we have found s, we can find v_x', v_y', and v_z' using Eqs. (34)-(36), and we can calculate θ and ϕ from them. We can then find u and v using Eqs. (22) and (23).

4.1.5. Finding a Tangent to an Asp Edge

Given an asp edge in the form of Eqs. (22), (23), (39), and (40), we can find the slope at some point (θ,ϕ) on the edge by finding $d\phi/d\theta$. First we must find $ds/d\theta$ from Eq. (39). If we let $a = e_4 \tan\theta - e_1$, $b = e_5 \tan\theta - e_2$, and $c = e_6 \tan\theta - e_3$, this yields

$$\frac{ds}{d\theta} = \frac{(e_4 b - e_5 a)\sqrt{b^2-4ac} \pm a(e_5 b - 2 e_6 a - 2 e_4 c) \pm (-e_4)(b^2-4ac)}{4 a^2 \sqrt{b^2-4ac}\cos^2\theta} \tag{47}$$

Then we can find $d\phi/d\theta$ by taking the derivative of the right-hand side of Eq. (40), yielding

$$m = \frac{d\phi}{d\theta} = \frac{[v_y'(2 e_1 s + e_2) - v_x'(2 e_4 s + e_5)]\cos\theta \frac{ds}{d\theta} + v_x' v_y' \sin\theta}{v_x'^2 + v_y'^2 \cos^2\theta} \tag{48}$$

Given some point (θ,ϕ) on a curve, we can plug (θ,ϕ) into Eqs. (47) and (48) to get the slope of the curve. A tangent vector is then $\langle 1,m \rangle$.

An alternative, approximate method of finding a tangent to a curve at a point is computationally more efficient. If the point in question is at the value s_0 of the parameter s along the curve, the method is to find the point on the curve at the value $s_0 + \varepsilon$, where ε is small. Then the difference between the points is a vector in the direction of the tangent.

4.2. Finding the Intersection of Asps

In order to construct the asp for a face as occluded by other faces in front of it, we must subtract the asp for the other faces from the asp for the given face. This subtraction can be accomplished as a complement and an intersection operation on the asp cells. We will show how to find the intersection of cells of aspect space, i.e. asps for polygons. The asp for a polyhedron is the union of the asps for its faces.

4.2.1. Finding the Intersection of Two Asp Cells

We use a brute-force algorithm to find the intersection of two asp cells. We find the intersection of every pair of facets of the asps, and each time we find an intersection, we "glue together" the two asp facets at the intersection ridges. We then "cut away" the outer faces incident upon the ridges at which more than two facets meet. This process is illustrated for polygons in R^2 in Figures 6-8. In Figure 6

we show overlapping polygons. The intersection points have been added in Figure 7, and the outer faces "cut away" in Figure 8.

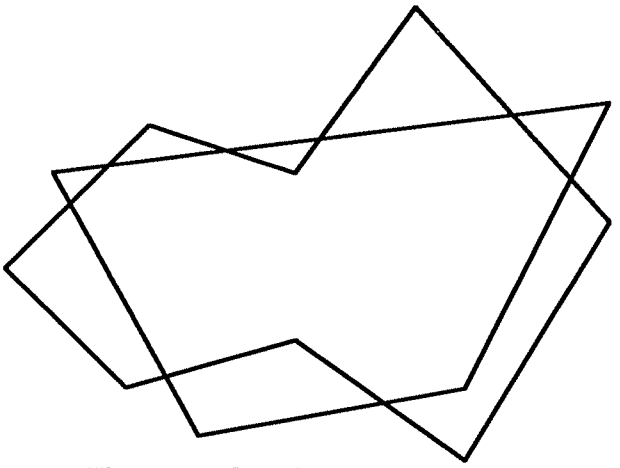

Figure 6. Overlapping polygons

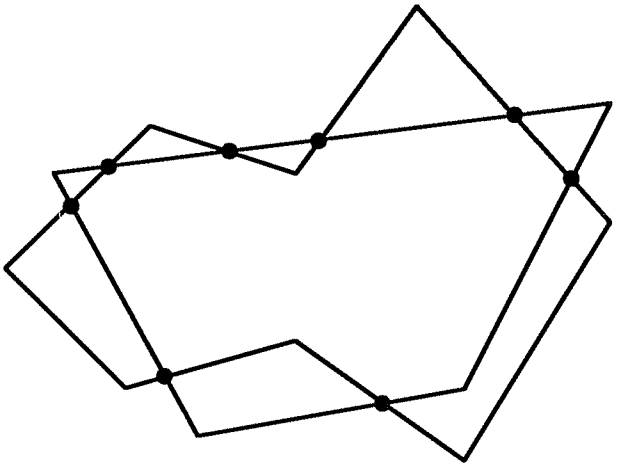

Fig. 7. Intersection points added to overlapping polygons.

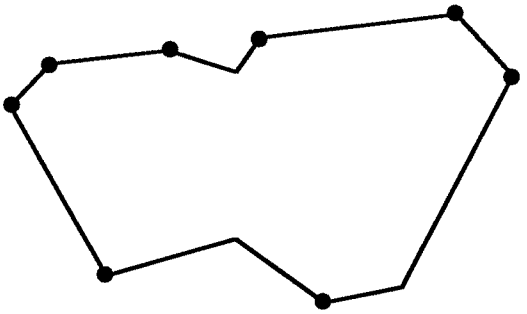

Fig. 8. The intersection results when the outer faces are cut away.

In the case of polygons in the plane, finding the intersection point of a pair of edges involves finding the intersection point of the lines containing the edges and determining whether that point lies in both segments. If so, we then split the

segments at the point and "cut away" the new half-segments that are in one of the polygons but not the other.

We illustrate the process for higher dimension by presenting a naive algorithm for the intersection of two k-polytopes in R^k. We state the algorithm recursively with four procedures:

d-polytope intersection in R^d:

Find the intersection of all pairs of facets using the d–1-polytope in R^d procedure below.

Cut away outer facets at ridges with four incident facets rather than the normal two.

d–1-polytope intersection in R^d:

Find the intersection of the hyperplanes (i.e. d–1 spaces) containing the polytopes.

If the hyperplanes are the same then solve the intersection problem in the hyperplane using the d-polytope in R^d procedure above recursively.

If the intersection is empty, return the empty set.

If the intersection is a d–2-plane, find the intersection of the d–1-polytopes with the d–2-plane (using the d-polytope d–1-plane intersection procedure below) and solve the d–2-polytope in R^{d-2} intersection problem using the d-polytope in R^d procedure above.

Split the two d–1-polytopes and insert the intersections into the d–2-polytopes that resulted from the intersection into the data structures for the d–1-polytopes.

d-polytope, hyperplane intersection in R^d:

Find the intersection of each facet with the hyperplane using the d–1-polytope, hyperplane intersection in R^d procedure.

The intersection will consist of d–2-polytopes in the hyperplane. Join them at common boundaries to form d–1-polytopes in the hyperplane.

Return the list of d–1-polytopes.

d–1-polytope, hyperplane intersection in R^d:

Find the intersection of the hyperplane **h** containing the d–1-polytope **p** and the given hyperplane.

If the hyperplanes are the same, return the d–1-polytope.

If the intersection is empty, return the empty set.

If the intersection is a d–2-plane, return the intersection of **p** with the d–2-plane in **h** using the d-polytope, hyperplane in R^d procedure.

4.2.2. The Algorithm

For finding the intersection of two asp cells, we will unroll the recursion and give procedures for the problems of each dimension, starting with the conceptually easier lower-dimensional problems.

Intersection of an edge and a curve on a 2-surface

Suppose we have an edge and a 1-surface or curve of the form of Eqs. (37)-(40) on a 2-surface of the form of Eqs. (22), (23), and (32). In order to find the points of intersection, we first find the intersection of the given curve and the curve on which the edge lies. The points of intersection are given by Eqs. (22), (23), and (46). There will be up to three intersection points of the two curves; it remains to determine whether the intersection points lie on the edge. We determine this by plugging the θ coordinate of the points into Eq. (39) yielding s, and determining whether $0 \le s \le 1$. If so, the point is on the edge. The procedure returns a list of zero or more points of intersection.

Intersection of two edges on a 2-surface

In order to find the intersection of two edges on a 2-surface, we find the intersection of one of the edges and the curve containing the other edge using the procedure given above. We test any returned intersection points to see if they are on the other edge. We determine this by plugging the θ coordinate of the points into Eq. (39) yielding s, and determining whether $0 \le s \le 1$. If so, the point is on the edge.

We then insert the intersection points into the data structures for the two edges, splitting each edge. We "cut away" the outer edges, so that only the two inner edges remain connected to each intersection point. Of the four edges potentially incident at each intersection point, we pick one of the lower ones, i.e. the predecessor of the other one on the same curve along the boundary of a face in the clockwise ordering. We delete it if it is to the left (or outside) of the other curve and retain it if it is to the right, in which case we delete the other edge along the same curve. We repeat the process for the other two edges.

We can determine whether an edge is to the right of a curve at a vertex in the following way: we compute the slope m_1 of the edge at the vertex. Then $<-m_1,1>$ is an outward normal to the edge at the intersection point in the (θ,ϕ)-plane. We also compute m_2, a tangent to the curve at the intersection point, in clockwise direction. If $<-m_1,1> \cdot <1,m_2> > 0$, i.e. $m_2 > m_1$ then the edge is to the right of (i.e. inside) the curve. The procedure inserts the intersection points into the data structure, cuts away the outer edges, and returns a list of intersection points.

Intersection of a ridge and a curve on a 2-surface

Suppose we have a curve **c** and a ridge on the same 2-surface. In order to find the intersection of **c** and the ridge, we find the intersection of each edge bounding the ridge with **c** using the curve-edge procedure above. For each intersection point we calculate the value of the parameter s of the point on **c**. The result is a sequence of points on **c**. We sort the points according to their position along the curve, and return a list of edges on **c** bounded by the intersection points. The first and second points are the boundaries of the first edge, the third and fourth points the boundaries of the next edge, and so on. We keep pointers to the edges that generated the vertices.

Intersection of two ridges on a 2-surface

Given two ridges $\mathbf{r}_1$ and $\mathbf{r}_2$ on a 2-surface, we find their intersection by first finding the intersection of every pair of edges, one from each ridge, using the procedure above. Since that procedure cuts away outer edges, the result is the intersection. However, the data structure will have many unconnected edges, etc. in it, so it must be "cleaned" by copying the surviving parts to a new data structure. The result is a list of ridges of intersection of the two ridges.

Intersection of a ridge and a 2-surface on a 3-surface

In order to find the intersection of a ridge and a 2-surface on a 3-surface, we first find the intersection of the 2-surface containing the ridge and the given 2-surface. If the 2-surfaces are the same, the problem is an intersection problem of two ridges on a 2-surface. We solve it using the procedure given above. If the 2-surfaces intersect in a 1-surface or curve, we find the intersection of the ridge and the curve on the 2-surface using the prodecure given above. This procedure returns a list of edges of intersection of the ridge and 2-surface.

Intersection of two ridges on a 3-surface

Given two ridges $\mathbf{r}_1$ and $\mathbf{r}_2$ on a 3-surface, we first find the intersection of the 2-surfaces containing the ridges. If they are the same, the problem is one of finding the intersection of ridges on a 2-surface, which is done using the procedure above. If they are different, their intersection is a curve, and we find the intersection of each ridge with the curve on a 2-surface. Then find the intersection of the lists of edges by finding the intersection of each edge of one list with all the edges of the other list, and concatenating the results. Next, insert the endpoints of the edges into the ridges that generated them, splitting those ridges as necessary.

The next step is to cut away the outer ridges at each edge of intersection. That can be done by finding the endpoints of the edge and each ridge that intersected the edge at that point. In each ridge, cut away the two outer edges incident at that vertex. As a result, the outer edges at that ridge will have been cut away.

Intersection of a facet and a 2-surface on a 3-surface

This problem is analogous to the problem of finding the intersection of a polyhedron and a plane in 3-space. The result is a list of ridges (polygons) on the 2-surface. First find the intersection of each ridge bounding the facet with the given 2-surface using the procedure above. The result of each intersection is a list of edges. Since the procedure must return a list of ridges on the 2-surface, it remains to connect the edges into ridges.

Note that the edges have the same connectivity as a subset of the ridge-edge structure of the original facet. That is, two edges of intersection are connected at a

vertex if the two ridges that they came from are connected at an edge. Thus, for every endpoint of an edge of the intersection, we can determine the other endpoint that it should be joined with by using the fact that they resulted from the intersection with the same edge of the facet. For every new vertex resulting from the intersection of the plane with an edge of the facet, we keep a pointer to the edge giving rise to it. Joining vertices is then a matter of checking each vertex, and if it has not already been joined with another one, tracing the link to the edge that gave rise to it and tracing the link from there to the other vertex generated by that edge.

Intersection of two facets on a 3-surface

In order to find the intersection of two facets on a 3-surface, first find the intersection of every pair of ridges of the facets using the procedure above. The procedure for finding the intersection of the ridges inserts the intersection edges into the two facet data structures and deletes the outer edges at vertices with four incident edges on a 2-surface. The procedure then "cleans" the data structure by copying the surviving parts. The result is a list of facets.

Intersection of two facets in aspect space

In order to find the intersection of two facets in aspect space, first find the intersection of the two 3-surfaces containing the facets. If the 3-surfaces are the same, then solve the problem with the facets-on-a-3-surface procedure, above. If the 3-surfaces are different, then their intersection is a 2-surface. The procedure above can be used to find the intersection of each facet with the 2-surface on the 3-surface. Two lists of ridges on the 2-surface result. Find the intersection of these lists by finding the intersection of each ridge of one list with every ridge of the other list, using the procedure above. The result is a list of ridges of intersection of the two facets.

The next step is to insert the ridges into the two data structures and cut away the outer facets at the ridges of intersection. The ridges are inserted by following the pointers to the facets that generated the ridges and inserting the ridges into those facets. The outer facets at each ridge of intersection are then cut away by finding the edges bounding the ridge. The outer ridges are cut away at those edges in the same manner as in the procedure for finding the intersection of two ridges on a 3-surface. The result of cutting away those ridges is that the outer facets at the ridges of intersection will be disconnected.

Intersection of two cells in aspect space

In order to find the intersection of two cells of aspect space, we first find the intersection of every pair of facets of the cells using the procedure above. The procedure for finding the intersection of the facets inserts the intersection ridges into the two facet data structures and deletes the outer structure. Then the data structure must be "cleaned" by copying the surviving parts. The result is a list of cells of aspect space, the intersection of the two cells.

4.3. Constructing the Asp

Constructing the asp for a polyhedron involves constructing the asp for each face as occluded by the asps for the faces in front of it and taking the union. The asp for a face **f** as occluded by the faces in front of it is the asp for **f** minus the asps for the faces in front of it. Thus, to find the asp for **f** as a part of the polyhedron, first find the asp for **f** by itself. This is the same as in the convex case, i.e. it has a cell for the whole face, facets for the edges of the face, and ridges for the vertices of the face. The visibility of every vertex (as a part of **f**) is bounded by a great circle of viewpoints corresponding to the horizon for **f**.

The faces in front of **f** are found by computing the intersection of the plane containing **f** with the planes containing the other faces. The (directed) line of intersection of two planes is parallel to the cross product of outward normals to the planes. Then if every vertex is above the line in the plane (i.e. to the left in the direction of the intersection line), the face is in front of **f**. If every vertex is below the line in the plane, the face is behind **f**. If some vertices are above the line and some below, find the intersection points of edges with the line and cut the polygon at the intersection points. One or several polygons above **f** may result.

Once we have found the faces above **f**, we find the asps for those faces and complement them. Then we take the intersection of **f** with each of those faces. (Complementing the faces may just involve setting a flag, so that in the "cutting away the outer faces" step of the intersection algorithm we cut away the proper faces.)

5. Constructing an Image from the Asp for a Non-convex Polyhedron

Given a viewpoint (θ,ϕ), the (θ,ϕ)–cross-section of the asp is the view of the polyhedron from that viewpoint. Thus drawing the (θ,ϕ)–cross-section of the asp is equivalent to constructing an image of the polyhedron from that viewpoint with hidden lines removed. In this section we show how to find that cross-section of the asp for a non-convex polyhedron.

In the non-convex case, each cell of the asp corresponds to a region of the image, which may be a face or part of a face of a polyhedron The appearance of the part of the face of the polyhedron from a viewpoint (θ,ϕ) is the cross-section of the cell from (θ,ϕ). The cross-section of the cell for a fixed viewpoint is a polygon since regions in an image corresponding to parts of faces of a polyhedron are always polygons. Each facet bounding the cell corresponds to an edge of the image polygon from some viewpoint, but not all of the facets correspond to visible edges of the polygon from (θ,ϕ). Thus, for the viewpoint (θ,ϕ), we must find the facets that are visible, find the edges in the image that they correspond to, and draw the edges.

A facet is bounded by a number of ridges, corresponding to vertices in the image. Since the cross-section of a facet at a particular viewpoint is an edge in the image and all edges in the image have two visible endpoints, if the facet is visible then two of the ridges are defined at the viewpoint (that is, the projection of the viewpoint onto the surface containing the ridge is in the ridge). If the facet is not visible, none of the ridges will be visible. Thus, to determine whether a facet is visible from (θ,ϕ), it is sufficient to check all of the bounding ridges. If two of them are visible, then the points of intersection of those ridges with the (θ,ϕ)-plane are the endpoints in the image of an edge bounding a polygon.

It remains to determine whether a ridge is visible at a given viewpoint. Equivalently, we can project the bounding asp edges and vertices into viewpoint space and determine whether the viewpoint is in the region of viewpoint space that they bound. We present in detail below the algorithm for determining whether the point is in the region and therefore whether the object vertex is visible.

5.1. Determining Whether an Asp Ridge is Visible

An asp ridge is bounded by a cycle of asp edges and vertices; the edges are of the form of Eqs. (22), (23), and (40) above. Notice that Eq. (40) does not have any terms involving image plane coordinates. Thus we can ignore Eqs. (22) and (23) for the edges and consider Eq. (40) by itself as the projection of an edge onto viewpoint space. Determining whether a ridge is visible then amounts to determining whether

a point is in a region bounded by edges of the form given by Eq. (40). The boundaries of the region are not arcs of great circles, so the region is not spherical polygonal as in the convex case. However, we can determine whether a point is in the region using the same algorithm, since that algorithm did not make use of the spherical-polygonal nature of the region.

The algorithm we used in the convex case for determining whether a point is in a polygon was to pick a great circle in viewpoint space containing the viewpoint and intersecting the boundary of the projection of the ridge. We then found the closest intersection point of this great circle with the boundary of the ridge and determined whether that intersection went into or out of the polygon. The only difference in the non-convex case is that the projection of asp edges into viewpoint space are no longer linear but quadratic.

We find a great circle intersecting the test viewpoint and an edge of the region in viewpoint space in the same manner as in the convex case. In this case, the viewpoint on an edge of the region is on an edge of the form given by Eqs. (26), (27), (30), and (32), with some s chosen so that $0 < s < 1$. As in the convex case, a normal to the great circle given by Eq. (14) above. We would like to find an equation for the great circle of viewpoints for θ in terms of ϕ. To do this we note that the dot product of the normal $\mathbf{n} = (n_x,n_y,n_z)$ and a viewpoint (θ,ϕ) on the great circle is zero. This yields

$$\tan \phi = \frac{n_x \sin \theta + n_z \cos \theta}{n_y} \tag{49}$$

In order to find the viewpoint of intersection of this great circle with an asp edge, we solve (49) and (40) for θ. This yields

$$\tan \theta = -\frac{v_y' n_y + v_x' n_z}{v_x' n_x} \tag{50}$$

We must then determine whether the points of intersection (θ,ϕ) thus calculated actually lie on the edge of the asp. We do this by calculating s from Eq. (39) and determining whether $0 \le s \le 1$. If so, (θ,ϕ) is an intersection point of the great circle and the edge.

As we find all of the intersection points of edges with the great circle, we can calculate their distance from the test viewpoint and keep track of the closest intersection point. We can use the Cartesian distance or the angle between the two points on the unit sphere.

Once we find the closest intersection point, in order to determine whether the viewpoint is in the spherical polygon it remains to determine the sense of the intersection. That is, we must determine whether the great circle is going in to or out of the projection of the ridge in viewpoint space. In order to determine that, we take the dot product of a tangent to the curve in the clockwise direction and the normal to the plane of the great circle. If the dot product is greater than zero, the great circle is going *in* to the region of viewpoint space, so the point is not in the region. If the dot product is less than zero, the great circle is going *out* of the region, and the ridge is visible from that viewpoint.

To this point we have determined whether the test viewpoint is inside the region of viewpoint space. For each ridge of the asp, we make that test. Then for each facet, if it has two visible bounding ridges, we draw an edge in the image between the image points corresponding to those two ridges at the given viewpoint. The result of this process is the display of all of the visible edges of the polyhedron.

6. Conclusion

In this paper we have presented in detail algorithms for constructing the asp and for displaying a cross-section of the asp from a given viewpoint. Separate algorithms are presented for the convex and the general cases. The algorithms for the convex case have been implemented, and the algorithms for the non-convex case are in the process of being implemented.

REFERENCES

[1986] Plantinga, W. Harry and Charles R. Dyer, "An Algorithm for Constructing the Aspect Graph," Proc. 27th Annual Symp. on Foundations of Comput. Sci., 1986, pp. 123-131.

[1987a] Plantinga, W. Harry and Charles R. Dyer, "The asp: a continuous viewer-centered representation for 3D object recognition," TR 682, Department of Computer Sciences, University of Wisconsin, Madison WI 1987.

[1987b] Plantinga, W. Harry and Charles R. Dyer, "The Aspect Representation," TR 683, Department of Computer Sciences, University of Wisconsin, Madison WI, 1987.

[1987c] Plantinga, W. Harry and Charles R. Dyer, "The asp: a continuous viewer-centered represntation for 3D object recognition," Proc. IEEE First Int. Conf. on Computer Vision, 1987, pp. 626-630.

[1987d] Plantinga, W. Harry and Charles R. Dyer, "Visibility, occlusion, and the aspect graph," TR 736, Department of Computer Sciences, University of Wisconsin, Madison WI, 1987.