Shaded Rendering and Shadow Computation for Polyhedral Animation

W. Brent Seales
Charles R. Dyer

Department of Computer Science
University of Wisconsin
Madison, Wisconsin 53706 USA

Abstract

In this paper we discuss an approach for solving the problem of rendering a shaded sequence of views of a polyhedral scene with shadows and with hidden surfaces removed. We concentrate specifically on the hidden surface problem, the computation of shadow regions, and video-rate display. The method relies on a precomputation phase which constructs the aspect representation, a representation which encodes visibility information for each face in the model. The method exploits the frame-to-frame coherence present in such a sequence by deriving from the aspect representation only the changes in appearance with respect to viewpoint. The primary off-line cost of the shaded display is the cost associated with constructing the aspect representation. The primary on-line cost is the scan-conversion of the visible surfaces and shadow polygons.

Keywords: Animation, Aspect Representation, Hidden-surface computation, Interactive Viewing, Viewpath Coherence.

1. Introduction

Interactively viewing a static scene requires the computation and display of a sequence of images generated from a user-controlled, continuously-changing viewpoint. The goal of a system for interactive viewing is to render a 3D scene as it would appear from a particular vantage point, and to update the appearance of the scene dynamically as the user interactively modifies the vantage point within the environment. Applications include computer-aided design and visualization [Farr85], flight simulation [Yan85] and architectural walk-through [Broo86].

There are two essential requirements for this type of animated display: realism in each image and video-rate display. In meeting these requirements there is a fundamental trade-off between off-line and on-line solutions. The precomputation of information can be as extreme as complete image rendering off-line. The on-line animation phase is then reduced to a playback of the precomputed images [Denb86]. While the images rendered can be very realistic, the main drawback of the total precomputation approach is the size and inflexibility of the resulting animation description. At the other end of the spectrum is the synthesis of the frames of the animation sequence on-line. Animation in this case depends on fast dynamic frame rendering [Fuch83]. Both situations, however, require some amount of precomputation in order to achieve video-rate display and realistic image synthesis.

In this paper, we discuss the problem of displaying a polyhedral scene interactively from a moving viewpoint. We refer to this as *interactive viewing*; we consider in detail the restricted viewer motion of a great circle on the unit sphere under orthographic projection. We present an algorithm for animating the display of a polyhedral scene with hidden-surfaces removed, including the use of shading models and shadow computation.

Our algorithm takes advantage of *viewpath* coherence, a form of frame-to-frame coherence, by computing the appearance of the scene in the first frame, and then computing the viewpoints in viewpoint space at which the scene changes substantively, that is, at which the topological structure of the image changes. The viewpoints at which the appearance changes substantively are computed through the construction of the *aspect representation* for the scene, abbreviated by the word *asp*. The asp is a representation that makes explicit exactly which vertices, edges, and faces are visible from all viewpoints. The algorithm has two phases: a preprocessing phase, in which the initial appearance of the polyhedron and all events in viewpoint space are computed, and the visible edge graph is constructed; and an on-line, interactive phase, in which a sequence of frames along a user-controlled viewpath is displayed in real time.

The asp quantifies explicitly how all possible kinds of *visual events* occur as a continuous function of viewpoint. Informally, a visual event can be thought of as any change in the scene as a result of occlusion. Thus the disappearance of

The support of the National Science Foundation under Grant No. IRI-8802436 is gratefully acknowledged.

a face as it turns away from the viewing direction is one such event. The partial occlusion of an edge by some other edge (the beginning or ending of a T-junction) is another kind of event. More formally, all topological changes in a polyhedral scene are the result of the apparent intersection in the image of three edges [Plan88].

The orthographic viewing model can be characterized as the set of viewing directions defined by the points on the surface of the unit sphere S^2 (see Fig. 1). Any viewer movement can be described as a 1D path of viewpoints on the surface of S^2. In general, the apparent intersection of three edges occurs at a single point along a 1D view path on S^2, and corresponds to a topological change in the appearance of the scene. The location of all such events can be computed and ordered sequentially along the 1D viewpath. By precomputing and ordering the viewpoints where topological changes occur, the coherency between viewpoints is exploited. This coherence between frames is a result of the fact that for most small changes in viewpoint along a smooth viewpath, only linear changes in the appearance of the scene take place. Linear changes are those changes in the viewpoint which do not change the structure of the projected line drawing in the image. Changes in the structure are characterized in the asp as visual events corresponding to the apparent intersection of edges in the image. In order to take advantage of the coherence between viewpoints and hence between frames, these topological changes are explicitly computed and stored in order to represent exactly the changes in appearance in viewpoint space. At runtime a viewpath through this viewpoint space determines which events are relevant to updating each successive frame in an animation sequence.

Hubschman and Zucker introduced the idea of using frame-to-frame coherence to decrease the time required for hidden-line removal [Hubs81]. Their method was restricted to convex polyhedra and required $\Theta(n^3)$ storage, however. Our algorithm places no restrictions on the model polyhedra, allowing arbitrary nonconvexities, requires less storage space, and applies to the hidden-surface case.

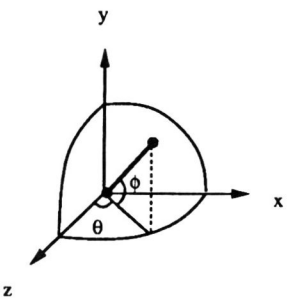

Figure 1. The viewpoint (θ, ϕ) on the unit sphere.

Shelley and Greenberg used frame-to-frame coherence for the generation of an animation sequence corresponding to a smooth viewpath through a 3D environment [Shel82]. A smooth, interactively-defined viewpath was represented as a B-spline, and was exploited to reduce the expense of the sorting and culling operations for visible line/surface computation. Although the viewpath was specified interactively, the computation of the appearance of the scene along the viewpath was done off-line. In contrast, our approach generates the appearance of the scene on-line as the viewpath is interactively defined.

The Binary Space Partition Tree (BSP-tree) has been used to display polyhedral scenes in near real-time by precomputing a structure which gives relative depth-ordering for faces in the model [Fuch83]. The display of a single frame from some viewpoint with hidden-surfaces removed involves traversing the BSP-tree to generate a list of polygons in back-to-front order. The BSP-tree does not take advantage of the frame-to-frame coherence in animation sequences, however. Each frame of a viewpath is generated by a separate and complete traversal of the BSP-tree structure. Also, the BSP-tree approach only applies to hidden-surface removal, not hidden-line removal. Finally, the BSP-tree approach requires drawing all of the polygons in the BSP-tree, whereas our algorithm only draws the visible polygons.

We concentrate in this paper on the efficient application of the asp to the hidden-surface problem of interactive viewing. Section 2 defines the asp and the visual event for polyhedral objects. Section 3 presents how the asp is used with shading models, and how the asp encodes shadow regions. The use of the asp to display an animation sequence for a 1D rotation sequence with hidden-surfaces removed is presented in Section 4. Section 5 concludes with results from a prototype implementation and a discussion of extensions such as perspective projection and arbitrary viewer motion.

2. The Aspect Representation

The aspect representation, or *asp* for short [Plan90, Plan88], quantifies the appearance of each face in a polyhedral scene, taking into account the occlusion relationships between faces. The appearance of a single face can be represented as a 4D volume in the space of the image plane $\times$ viewpoint space. This multi-dimensional space is called *aspect space*, and its properties are dependent upon both the projection process and the model geometry. The 4D volume corresponding to a single planar face is constructed by computing its boundaries which are hypersurfaces, curves, and points. A complete discussion of 4D aspect space can be found in Plantinga's thesis [Plan88]. In this paper viewpoint is restricted to a single degree of freedom. Using the viewing model shown in Fig. 1, we restrict viewpoint by assuming $\phi = 0$. Thus aspect space is reduced to 3D, and so we describe the aspect representation with respect to this restriction.

The equations which give the position in the image plane of a vertex in $\mathbb{R}^3$ under orthographic projection define a 1D curve in aspect space. An edge connecting two vertices can be parametrized with a single parameter in $\mathbb{R}^3$, so that the

equations which give the position of an edge in the image plane define a 2D surface in aspect space. The 2D surface for an edge is bounded by the 1D curves for the vertices of that edge in aspect space. The appearance of a face is no more that the appearance of each of the edges of the face, connected by common vertices. Note that points on a face lie inside a 3D volume in aspect space which is bounded by the 2D surfaces and 1D curves corresponding to the edges and vertices of the face. This 3D volume and its boundaries is called the *asp* for a face. The asp for the scene is the collection of asps for the faces in the scene.

A fundamental property of aspect space is that occlusion in object space is equivalent to set subtraction in aspect space. Consider two faces and their corresponding asps. A point which lies within the asps for both faces is a single image point generated from both faces. Since only one face can be visible, the point is removed from the asp for the face which is occluded. Thus, the exact set of visible points of a face from all viewing directions can be computed by performing set subtraction in aspect space. The algorithm for constructing the asp for a face **F** is to subtract from the asp for **F** the asps for the faces in front of **F**. Subtraction is set subtraction, and faces in front of **F** are those faces which occlude **F** from some viewpoint. The subtraction of one asp from another corresponds to the removal of the region of intersection of two 3D volumes.

The well-formed nature of the 3D asp for a face allows the set subtraction to be performed in a way similar to polyhedral intersection. Specifically, the construction of the asp for a scene is done by defining the intersection of the boundaries of two 3D asps. The 2D surfaces which bound an asp correspond to the visibility of the edges of a face. The intersection of two 2D surfaces in aspect space is a 1D curve which lies on each of the 2D surfaces. Since each 2D surface in aspect space corresponds to the visibility of an edge in the model, this 1D curve represents the apparent intersection of the two edges in the image. This apparent intersection is a visual event denoted as an EE-event. The equation for the corresponding curve is derived from the geometry of the apparent intersection (see Fig. 2(a)).

Consider two edges $\mathbf{E}_1$ and $\mathbf{E}_2$ in $\mathbb{R}^3$. Assume that $\mathbf{E}_1$ connects the points $\mathbf{p}_1 = (x_1, y_1, z_1)$ and $\mathbf{p}_1 + \mathbf{a}_1$ where $\mathbf{a}_1 = (a_1, b_1, c_1)$. Likewise, $\mathbf{E}_2$ connects the points $\mathbf{p}_2 = (x_2, y_2, z_2)$ and $\mathbf{p}_2 + \mathbf{a}_2$ where $\mathbf{a}_2 = (a_2, b_2, c_2)$. The edge $\mathbf{E}_1$ can be parametrized in $\mathbb{R}^3$ by $\mathbf{p}_1 + s\,\mathbf{a}_1$, where $0 \le s \le 1$. The viewing direction $\mathbf{v}$ can be described in spherical coordinates as the vector $\mathbf{v} = (\sin\theta, 0, \cos\theta)$, a function of the viewpoint parameter θ. Now, given a viewpoint $\mathbf{v}$, $\mathbf{E}_1$ and $\mathbf{E}_2$ appear to intersect in the image plane at

$$u = (x_1 + s\,a_1)\cos\theta - (c_1 + s\,z_1)\sin\theta \qquad (1)$$

$$v = y_1 + s\,b_1$$

where

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

Eq. 1 gives the coordinates in the image plane (u, v) of the apparent intersection of $\mathbf{E}_1$ and $\mathbf{E}_2$ as a function of only the viewpoint parameter θ. Fig. 2 shows the edges $\mathbf{E}_1$ and $\mathbf{E}_2$ in $\mathbb{R}^3$ and the apparent intersection of $\mathbf{E}_1$ and $\mathbf{E}_2$ in the image plane.

The 1D curve of intersection described by Eq. 1 extends over some range of viewpoints $[\theta_i, \theta_j]$. Fig. 2(a) shows the apparent intersection of the edges $\mathbf{E}_1$ and $\mathbf{E}_2$ projected into the image plane. As the viewpoint changes, the point where the edges appear to intersect changes according to Eq. 1. Fig. 2(b) shows the 2D surfaces in aspect space which correspond to the edges $\mathbf{E}_1$ and $\mathbf{E}_2$. The 1D curve of intersection is the trace of the apparent intersection of $\mathbf{E}_1$ and $\mathbf{E}_2$ as described by Eq. 1.

In constructing the 3D asp for a face, points in aspect space can arise as the intersection of a 2D surface and a 1D curve described by Eq. 1, or the intersection of two 1D curves. An intersection point of this type corresponds geometrically to the apparent intersection of three edges. This apparent intersection is a visual event denoted as an EEE-event. The geometry, shown in Fig. 3(a), yields an explicit

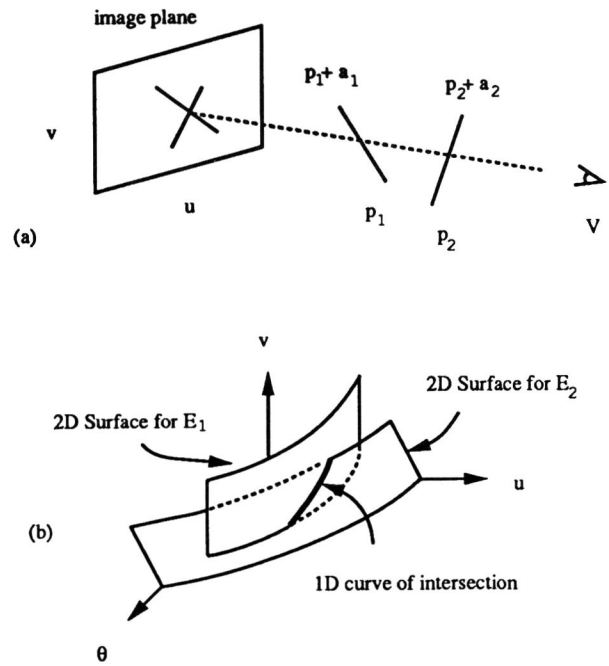

Figure 2. 1D curve generated by the intersection of two 2D surfaces in aspect space. **(a)** The apparent intersection of two edges in the image plane. **(b)** Intersection of the two 2D surfaces and the resulting 1D curve in aspect space.

equation for such points. Consider three edges E_1, E_2 and E_3 which lie in $\mathbb{R}^3$ as illustrated in Fig. 3(a). Let S_i represent the 2D surface in aspect space corresponding to E_i. The intersection of S_1 and S_2 in aspect space is a 1D curve C_1 on S_1. Similarly, the intersection of S_1 and S_3 in aspect space is a 1D curve C_2 on S_1. It is the intersection of the two curves C_1 and C_2 on S_1 in aspect space which corresponds to the viewing direction where all three edges appear to intersect in the image plane. This viewing direction is equivalent to the direction of some vector through a point $\mathbf{p}_1 + s\mathbf{a}_1$ on E_1, $0 \le s \le 1$, which intersects both E_2 and E_3. A vector parallel to this direction is given by

$$\mathbf{v} = ((\mathbf{p}_1 + s\mathbf{a}_1 - \mathbf{p}_2) \times \mathbf{a}_2) \times ((\mathbf{p}_1 + s\mathbf{a}_1 - \mathbf{p}_3) \times \mathbf{a}_3) \quad (2)$$

Since the y-component of the viewing direction is constrained to be zero, Eq. 2 has the form

$$\alpha s^2 + \beta s + \gamma = 0 \quad (3)$$

where α, β and γ are linear functions of the endpoints of E_1, E_2 and E_3. Eq. 3 can be solved for the parameter s, and then Eq. 2 can be solved for the desired vector v. The value of the viewpoint parameter θ for vector v is the value of θ where the two 1D curves C_1 and C_2 intersect in aspect space on S_1. This is shown in Fig. 3(b).

The goal of the construction of the asp is the construction of the exact boundaries of the 3D volume in aspect space. These boundaries are a mixture of simple vertices and vertices resulting from EEE-events, as well as simple curves and curves corresponding to EE-events. The computation and organization of these boundaries is the key component in the generation of a fast sequence of different views of the face.

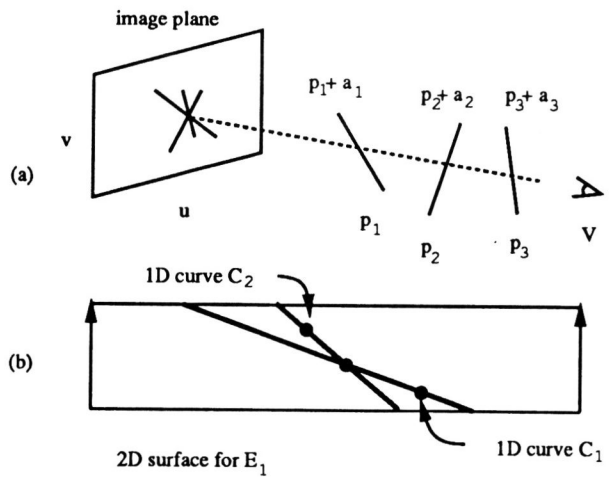

Figure 3. (a) The apparent intersection of three scene edges in the image plane (b) The intersection of two 1D curves on a 2D surface in aspect space

Fig. 4 shows part of the asp computed for a large square face as occluded by a smaller square face. The asp for the small face is subtracted from the asp for the larger, generating 1D curves corresponding to T-junctions. The asp for the large face has a hole which is generated by the subtraction of the asp for the small face. The cross-sections in Fig. 4(c) show the appearance of the large face for several viewpoints. The asp is bounded by the curves which are computed by the asp intersection process.

3. Shaded Display and Shadow Computation

Using the asp for shaded display of the model requires two important features: appropriate ordering information in the asp, and the 3D coordinates of each 2D polygon point derived from the asp. The ordering is necessary for scan-conversion. The 3D coordinates for each 2D point are necessary for shadow computation. An interpolated shading model can be applied using precomputed normals on each face, so we concentrate here on the shadow computation.

3.1. The Asp Cross-Section

The appearance of a face is extracted from the asp by computing a cross-section of the asp volume. By cross-section we mean the intersection of the asp volume with a plane corresponding to a fixed viewpoint. This appearance includes occlusion, so the extent of the cross-section in the image plane is a closed (but not necessarily connected) region corresponding to the appearance of the face with the hidden parts removed. Thus the interior can be shaded without affecting any other visible element in the scene. In Fig. 4(c), the cross-section of the asp for the large face is shown for each topologically different part of the volume. The appearance for the large face can be obtained from the asp directly as an ordered set of vertices, since the curves defining those vertices are themselves ordered. It is the organization of these curves over the viewpoint dimension of aspect space which allows for efficient animation. This organization will be discussed further in Section 4.

3.2. Recovering 3D Points from the Asp

Computing shadow regions depends on associating each point in the 2D appearance of a face with the 3D point which generates it. Note that the appearance obtained from the asp contains only image coordinates without the depth information. Under the orthographic viewing model, however, the added constraint of the equation of the plane which generates the appearance is sufficient to fully determine the 3D coordinates for 2D points in the appearance.

The linear transformation **T** which recovers the 3D coordinates corresponding to a 2D image point is derived from the description of the asp for a single point $\mathbf{P} = (x, y, z) \subset \mathbb{R}^3$. The asp for this fixed point is a curve in aspect space describing the image coordinates of **P** as a function of the viewpoint parameter θ on the view sphere (see Fig. 1). Given only the image point (u, v) derived from a particular asp, the corresponding 3D point can be recovered using the added constraint of the plane in $\mathbb{R}^3$ on which **P** is known

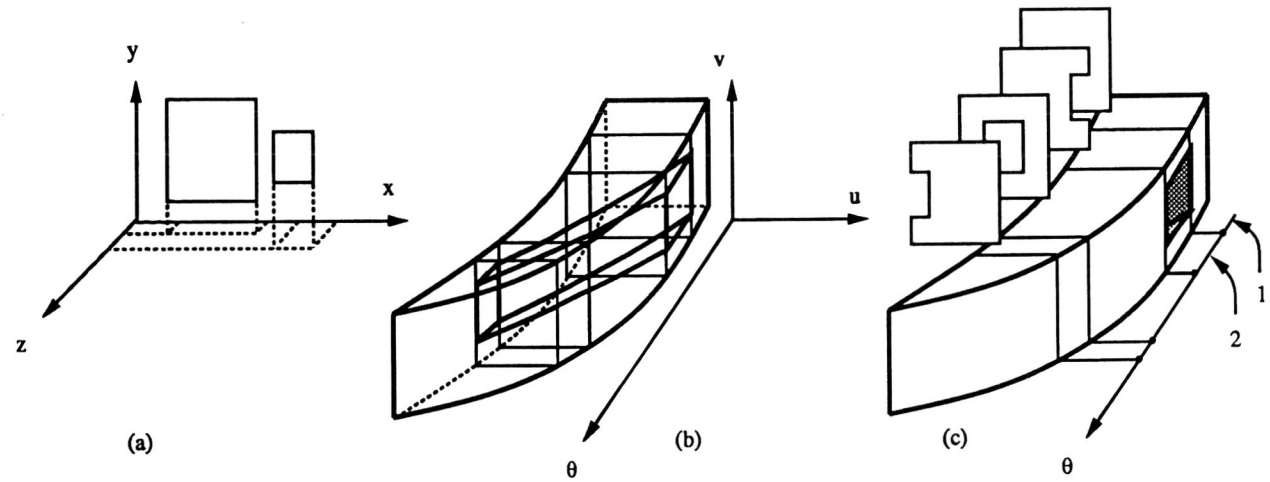

Figure 4. The asp for a large square face as occluded by a smaller face. (a) The two faces sit in 3D so that the small face is slightly in front of the larger one. (b) The asp for the large face has a hole from the removal of the asp for the small face. (c) The cross-sections above the asp show the appearance of the large face between visual events.

to lie. This information fully determines the linear transformation **T**. Thus **T** is a linear mapping from the image plane to the plane P.

3.3. Computing Shadow Regions

When the position of the viewer is different from the position of the light source, regions in the scene can lie in shadows. A shadow on a surface results from the occlusion of the light source by some other surface in the scene. Shadow region computation in polyhedral scenes has been incorporated into hidden surface algorithms using shadow volumes [Crow77] and polygon clipping [Athe78]. Most hidden-surface removal algorithms compute shadow regions by performing hidden-surface removal from the position of the light source [Joy88]. Similarly, a natural result of the viewer-centered asp is that the position of a (point) light source defines another viewpoint. The asp structure encodes appearance from all viewpoints so that the shadows generated by a light source can be computed by finding the cross-section of the asp for the light source position. The asp itself needs to be constructed only once, and movement of the light source results in another cross-section operation.

The asp represents the appearance of each face in the scene from all viewing directions. By treating the light source as an additional viewing direction, the asp can be used to obtain the appearance of a particular face from the viewpoint of the light source. The appearance of a face from the direction of the light source is the part of the face which is under direct illumination. A comparison of the appearance of a face from the viewing direction to the appearance from the light source direction gives the location of the shadows on the face which are cast from the light source. Fig. 5(a) shows a triangular and a rectangular face in 3D. The light source is positioned so that the triangular face casts a shadow on part of the rectangular face. Fig. 5(b) shows the appearance of the scene from the light source direction. From the position of the viewer, however, the appearance of the scene is different. As shown in Fig. 5(c), from direction of the viewer the rectangular face is only slightly occluded. Transforming the regions visible from the light source and projecting them from the viewer's position yields regions of full illumination and shadow (Fig. 5(d)).

The asp for a face can be used to directly compute the regions of a scene which lie in the shadows created by a single light source. Let **L** and **V** represent the light source position and the viewer position in viewpoint space, respectively. Let **f** be a face in the scene which lies on plane $P \subset \mathbb{R}^3$. Let $\mathbf{f_V}$ be the appearance of **f** from the viewing direction **V**, and let $\mathbf{f_L}$ be the appearance of **f** from the direction of the light source **L**. Both of these appearances can be extracted from the asp as a cross-section. The appearance of a polygon as encoded in the asp is always a set of closed polygons. Note that because of occlusion, a single polygon may actually appear to be a set of disjoint polygons. Thus $\mathbf{f_L}$ and $\mathbf{f_V}$ represent sets of polygons in the image plane. Furthermore, $\mathbf{f_L}$ represents the part of **f** which is illuminated by the light source **L**, and $\mathbf{f_V}$ represents the part of **f** which is visible to the viewer from the viewpoint **V**.

The computation of regions of the face **f** which lie in shadow hinges on a key observation: the polygons in each of the sets $\mathbf{f_V}$ and $\mathbf{f_L}$ are the orthographic projection of disjoint polygons in $\mathbb{R}^3$ which lie on the same plane containing **f**. That is, the transformation **T** can be applied to the edges and vertices in the image plane for both $\mathbf{f_V}$ and $\mathbf{f_L}$ in order to recover the corresponding 3D polygons on P. The two small triangular regions shown in the image plane in Fig. 6

correspond under **T** to the illuminated section of the large triangle in 3D. The part of the large triangle which is in the shadow of the rectangle is the triangular face with the illuminated sections removed. In general, the polygons under **T** are sets of closed polygons which lie on plane P in $\mathbb{R}^3$. Let $\mathbf{T}(\mathbf{f_V})$ and $\mathbf{T}(\mathbf{f_L})$ represent the sets of polygons in $\mathbb{R}^3$ on plane P corresponding to the transformation applied to $\mathbf{f_V}$ and $\mathbf{f_L}$, respectively. Then in general the intersection $\mathbf{T}(\mathbf{f_L}) \cap \mathbf{T}(\mathbf{f_V})$ represents the regions in $\mathbb{R}^3$ on **f** which are both visible and illuminated. The regions described by $\mathbf{T}(\mathbf{f_V}) - \mathbf{T}(\mathbf{f_L})$ represent those areas on **f** which are visible but not illuminated, i.e. shadow regions.

4. Interactive Display Using the Asp

The precomputation of the asp for each model face makes explicit the visual events which occur in viewpoint space. The viewpoints where EE-events and EEE-events begin and end represent transitions in the appearance of a face. Since all possible transitions in the appearance of a face are captured in the EE- and EEE-events [Plan88], the asp for a face encodes a complete description of the change in the appearance of the face through viewpoint space. Fig. 4(c) shows the viewpoints on the θ-axis where the EE-events and EEE-events begin and terminate. These events are precisely the viewpoints where appearance changes, so that with each event value a description can be stored of how the appearance has changed. The first segment of the asp (farthest from the viewer in Fig. 4(c)) corresponds to the unoccluded large square face. The first visual event along the θ axis corresponds to the two EE-events caused by the small face occluding the large one. At that visual event the description of the appearance of the large square face changes to include EE-event curves. In this way the asp is used to generate a sequence of visual events and a description of appearance between these events. Since we assume a 1D viewpoint space, the visual events in the viewpoint dimension can be sequentially ordered.

The algorithm for the display of the sequence exploits the precomputed information by only changing the representation of visible surfaces when a visual event occurs. The first image in the sequence of views is generated by taking an asp cross-section. As the viewpoint changes, visual events along the viewpath signal an update to a structure which stores the current appearance. A change in viewpoint which does not cross one of these visual event boundaries means that no topological change in the appearance has occurred. In this case the new view is a reprojection of the previous view from the new viewpoint, with no change in the structure of the hidden and visible surfaces.

When a visual event boundary is crossed, the change in appearance is explicitly represented in the asp. With each visual event is a description of how the appearance of visible faces in the scene change from the description before the event. This description can be precomputed for each event. Hence, for every event which occurs on-line, an update to the current appearance of the visible faces is made. The subsequent frames in the sequence are computed from the first by determining which visual events have occurred under the changed viewpoint, and then modifying the appearance of each face given the events. The frame-to-frame coherence in a sequence guarantees that only a small number of visual events will need to be processed between any two frames of the sequence.

These operations can be accomplished using two data structures derived from the asp: a list of visual events and an Edge Structure Graph (ESG). The ESG is a graph which

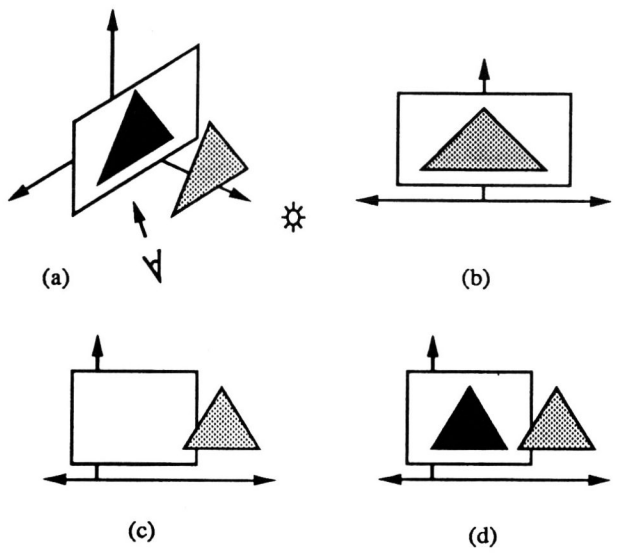

Figure 5. A 3D scene containing a single point light source. (a) The 3D scene. (b) The scene projected from the light source direction. (c) The scene as observed from the viewer's position without the shadow. (d) The scene observed by the viewer with the shadow.

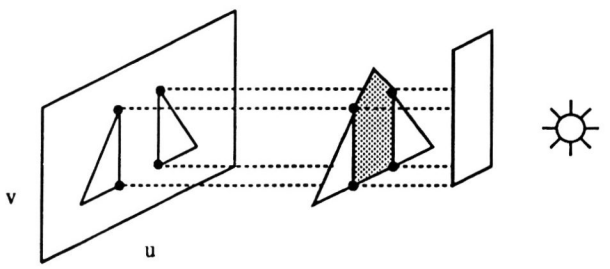

Figure 6. The regions of shadow and full illumination on the triangular face can be computed using the appearance of the face in the image and the transformation **T**.

describes the appearance of the faces in the scene in terms of an ordered list of curves in aspect space. For example, the ESG for the viewpoint interval 1 for the large square face in Fig. 4(c) is an ordered list of pointers to the four curves which correspond to the four vertices of the face. In viewpoint interval 2, the ESG for the large face changes to include four new pointers: two to the vertices defined by the EE-events with the small face, and two corresponding to the left two vertices of the small face. The ESG is constructed initially from the first frame of the sequence, and is modified for each visual event thereafter. The visual event list is generated from the aspect representation, and contains a complete, sorted list of visual events and corresponding updates to the ESG. The construction of both the visual event list and the initial state of the ESG is done off-line.

Shaded rendering of the sequence is separate from the hidden-surface computation. The exact appearance with hidden-surfaces removed is computed independently using the visual event list and the ESG. Any shading model can be used by the scan-conversion process once the visible polygon list is generated. In particular, the scene can be rendered as a hidden-line-eliminated scene by drawing only the boundaries of the visible polygons. Unlike depth-sorting algorithms, the hidden-surface computation is not dependent upon the shading process.

Generating a sequence which includes a moving light source amounts to the additional precomputation of a visual event list and ESG along the light source path. The change in the polygons in the shadow set for a light source can then be computed efficiently by updating the ESG according to the visual event list in the same way as the visible surfaces are maintained. For a static viewer and moving light source, the algorithm is the same as above. For both moving viewer and moving light source, two distinct ESGs and visual event lists are constructed.

The asp construction process is clearly the dominant computational cost. There are tight bounds on the time and space required to construct the asp for polyhedral scenes under more general viewing conditions [Plan90]. For this restricted case, the asp construction takes time $\Omega(n^2)$ to $O(n^4)$ depending on the visual complexity of the scene, where n is the number of faces in the scene. Only the most pathological scenes (i.e., grids) require the worst-case time bound, and in most cases the asp can be constructed in $\Omega(n^2)$ to $O(n^3)$ time. Since the algorithm requires storage for every visual event in the 1D viewpoint space, the storage required for events will be between $\Omega(n)$ and $O(n^3)$. Note that convex objects require linear storage for visual events. In practice, common objects have a visual complexity closer to the convex case. For those pathological scenes of high visual complexity, the storage required for visual events can become very large [Plan88].

5. Results and Extensions

A restricted prototype version of the preprocessing and on-line portions of this algorithm has been implemented [Plan90a, Seal89]. The implementation is written in C and was tested on a DECstation 3100 running Ultrix V2.1. The current prototype constructs the complete aspect representation for each face in a polyhedral scene as a set of disjoint edges surrounding the face. Thus the appearance of a face is represented as a collection of possibly disjoint segments which are not connected into closed polygons. The connectivity information has been omitted only for the purpose of initial simplicity in constructing the asp. The empirical results of the asp construction phase show that in practice the number of visual events is a small constant times the number of edges in the scene [Plan90a]. The space required to store the visual event list is correspondingly small.

The on-line portion of the program uses the visual event list from the aspect representation to display rotation sequences with hidden-lines removed. Timing analysis of the prototype program indicates that the real-time display of large polyhedral models is possible. Fig. 7 shows two of the models used in the prototype timing measurements. The

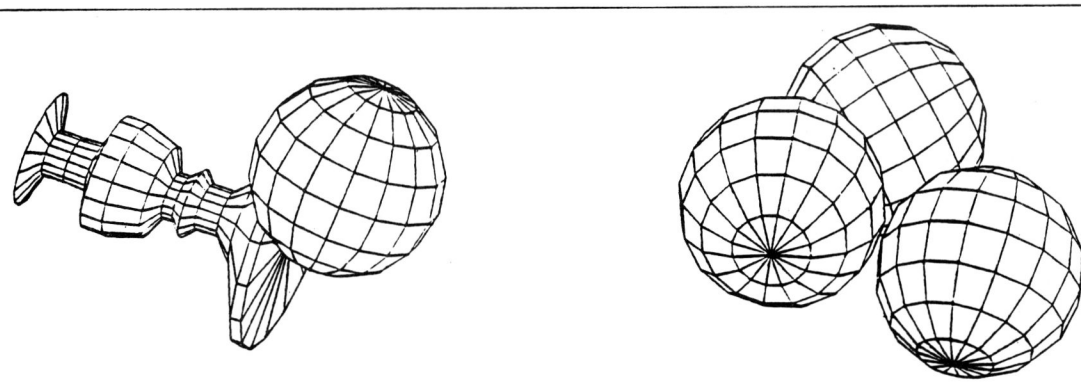

Figure 7. Two polyhedral models of moderate visual complexity. The candlestick with sphere contains 416 faces, and the three spheres contain a total of 384 faces.

model of the candlestick and sphere contains 416 faces, while the three spheres contain 384 faces. The prototype program can display these models at rates between 50 and 60 frames per second.

The animation of the rotation of shaded scenes with multiple light sources and shadows is a direct extension of this implementation. The key feature of this approach is the separation of the hidden-surface computation in the animation sequence from the application of the shading model and the scan-conversion process. The set of object resolution polygons representing the visible faces and the shadow polygons is computed efficiently using the visual event list derived from the aspect representation. Frame-to-frame coherence guarantees that only a small percentage of the visual events need to be processed between a single pair of frames. Rendering the scene with hidden lines removed could be done by drawing the edges of each of the visible surface polygons. The difference between hidden-line and hidden-surface elimination in this approach is only the raster display of the visible polygons. The primary cost of the shaded display is the cost associated with scan converting the visible surfaces and shadow polygons. This differs from most approaches where the hidden-surface computation itself dominates the total cost.

The construction of a complete list of visual events provides an exact representation of the change in appearance of the scene. This has several advantages over the brute force computation and storage of each frame independently. First, brute force computation of appearance must discretize the viewpath. This means that the viewer does not have interactive control over the "speed" and fineness of the display. Second, additional parameters such as shading and shadows which depend on light sources can be manipulated more dynamically using the asp. The brute force method must recompute the entire sequence for any change in shading or lighting. Finally, in practice, the asp requires less storage space than the storage of all of the frames in the brute force approach.

The asp can be computed under orthographic projection for two degrees of freedom in viewpoint. We are currently working on representations to allow arbitrary interactive viewer movement within this 2D viewpoint space. In addition, these algorithms can be extended to perspective projection models. The aspect representation under perspective projection can be constructed [Plan88]. Perspective projection adds another degree of freedom to aspect space, requiring the description of higher-dimensional surfaces. The asp under these models may require more time and space to compute. It is also worth noting that the sequence of viewpoints need not be a great circle. Any description of a view path which can be described in closed-form can be intersected with the aspect representation in aspect space to generate a visual event list.

References

[Athe78] Atherton, P., K. Weiler, and D. Greenberg, Polygon shadow generation, *Proc. SIGGRAPH*, 1978, 275-281.

[Broo86] Brooks, F., Walkthrough – A dynamic graphics system for simulating virtual buildings, *Proc. Workshop on Interactive 3D Graphics*, 1986, 9-21.

[Crow77] Crow, F. C., Shadow algorithms for computer graphics, *Proc. SIGGRAPH*, 1977, 242-248.

[Denb86] Denber, M. and P. Turner, A differential compiler for computer animation, *ACM Computer Graphics* **20**(4), 1986, 21-27.

[Farr85] Farrell, E., W. Yang, and R. Zappulla, Animated 3D CT imaging, *IEEE Computer Graphics and Applications* **5**(12), 1985, 26-32.

[Fuch83] Fuchs, H., G. D. Abram, and E. D. Grant, Near real-time shaded display of rigid objects, *Computer Graphics* **17**, July 1983, 65-69.

[Hubs81] Hubschman, H. and S. W. Zucker, Frame-to-frame coherence and the hidden surface computation: Constraints for a convex world, *Computer Graphics* **15**, 1981, 45-54.

[Joy88] Joy, K., C. Grant, N. Max, and L. Hatfield, *Tutorial: Computer Graphics: Image Synthesis*, IEEE Computer Society Press, Washington, D.C., 1988.

[Plan88] Plantinga, W. H., The asp: A continuous, viewer-centered object representation for computer vision, Ph.D. Dissertation, Computer Science Department, University of Wisconsin-Madison, 1988.

[Plan90a] Plantinga, W. H., W. B. Seales, and C. R. Dyer, Real-time hidden-line elimination for a rotating polyhedral scene using the aspect representation, *Proc. Graphics Interface*, 1990.

[Plan90] Plantinga, W. H. and C. R. Dyer, Visibility, occlusion, and the aspect graph, *Int. J. Computer Vision*, 1990. to appear,

[Seal89] Seales, W. B. and C. R. Dyer, Using the asp for the interactive viewing of polyhedral scenes, Technical Report 903, University of Wisconsin-Madison, December 1989.

[Shel82] Shelley, K. and D. Greenberg, Path specification and path coherence, *Computer Graphics* **16**(3), 1982, 157-166.

[Yan85] Yan, J., Advances in computer-generated imagery for flight simulation, *Computer Graphics and Applications* **5**(8), 1985, 37-51.