Unit 7
Visible Surface Detection
(Hidden Surface Removal)

Hidden surface removal or visible surface determination is the process used to determine which surfaces and parts of surfaces are not visible from a certain viewpoint.

Hidden surface determination is necessary to render an image correctly, so that one may not view features hidden behind the model itself, allowing only the naturally viewable portion of the graphics to be visible.

Visible surface detection methods are broadly classified according to whether they deal with objects or with their projected images.
These two approaches are:

1) ***Object space method:*** An object space method compares objects and parts of objects to each other within scene definition to determine which surfaces are visible.
2) ***Image space method:*** Visibility is decided point by point at each pixel position on the projection plane.

Most visible surface detection algorithm use image-space-method but in some cases object space methods are also used for it.

Algorithms

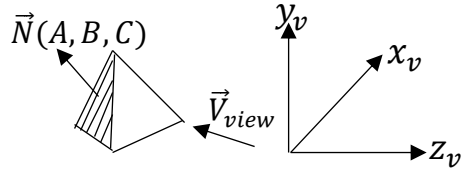
1. Back-face Detection (Plane equation Method)

A fast and simple object space method used to remove hidden surface from a 3D object drawing is known as "Plane equation method".
A point (x, y, z) is inside a surface with plane parameter A, B, C & D if

$$Ax + By + Cz + D < 0$$

i.e. it is in back face.
Let $\vec{N}$ be the normal vector for a polygon and $\vec{V}_{view}$ be the vector in viewing direction, then a polygon surface is back if

$$\vec{V}_{view} \cdot \vec{N} > 0$$



- For the left handed viewing system if the 'z' component of the normal vector is positive, then it is back face. If the 'z' component of the vector is negative then it is a front face.
- For the right handed viewing system if the 'z' component of the normal vector is negative, then it is back face. If the 'z' component of the vector is positive then it is a front face.

Limitations:

- This method works fine for convex polyhedral, but not necessarily for concave polyhedral or overlapping objects. So, we need to apply other methods to further determine where the obscured faces are partially or completely hidden by other objects (e.g. Using Depth-Buffer Method or Depth-sort Method).
- This method can only be used on solid objects modeled as a polygon mesh.

2. Depth Buffer Method (Z-Buffer Method)

- It is an image space method for detecting visible surface.
- The Z-buffer method compares surface depths of each pixel position on the projection plane. Normally z-axis is represented as depth.
- In this method, two buffers are used:
    1) Depth Buffer: Stores depth values for each (x, y) position.
    2) Frame Buffer: Stores the intensity values for each position.

Algorithm:

1) Set the buffer values:

DepthBuffer(x, y)=0,

FrameBuffer(x, y)=Background color

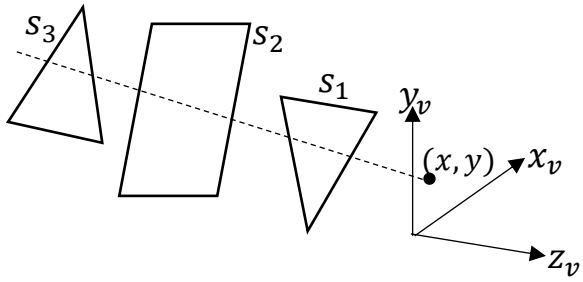
2) Process each polygon, one at a time as follows:

- *For each projected (x, y) pixel position of a polygon, calculate depth 'z'.*
- *If z>DepthBuffer(x,y), compute surface color and set*

DepthBuffer(x, y)=z,

FrameBuffer(x, y)=surfacecolor(x, y)

After all surface have been processed, the depth buffer contains depth values for visible surface and frame buffer contains corresponding color values for those surface.



In the figure, at view plane position (x, y), surface S_1 has the smallest depth from the view plane, so it is visible at that position.

Depth value for a surface position (x, y) are calculated from the plane equation as;

$$z = \frac{-Ax - By - D}{C}$$

Now, the depth z' of the next position (x+1, y) is obtained as; $z' = z - \frac{A}{C}$

Advantages:

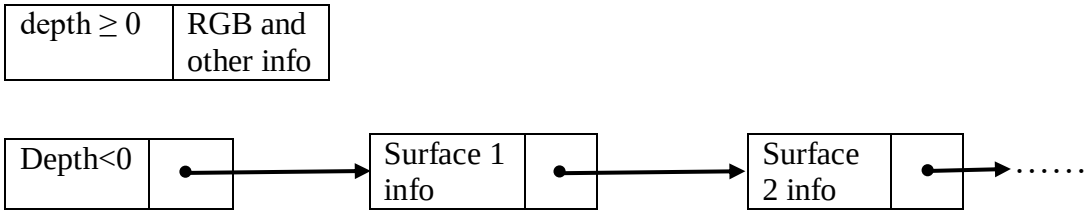
- It is easy to implement.
- It reduces the speed problem if implemented in hardware.
- It processes one object at a time

Disadvantages:

- It requires large memory
- It is a time consuming process as it requires comparison for each pixel instead of for the entire polygon.
- Deals only with opaque object but not for transparent object.

3. A-Buffer Method

- The A-buffer method is an extension of the depth-buffer method.
- This method is also known as **anti-aliased** or **area-averaged** or **accumulation buffer**.
- The A-buffer method is a visibility detection method developed at Lucas film studio for the surface rendering system REYES (Render Everything You Ever Saw).
- The A-buffer expands on the depth buffer method to allow transparencies.
- It consists of accumulation(A) buffer which as two fields:
  - ***a) Depth field:*** It stores a positive or negative real number.
  - ***b) Intensity field:*** It stores surface data or a pointer value.



If depth field is non negative (>=0), it indicates single surface. The intensity field then stores the RGB components of the surface color at that point and the percent of pixel coverage.

If depth is negative (<0) it indicates multiple surface. The intensity field then stores a pointer to a linked list of surface data.

Surface information in A-buffer includes:

- Depth - Surface identifier - Opacity parameter
- Percentage of area coverage - RGB intensity component
- Pointer to the next surface

Advantage:

- It provides anti-aliasing in addition to what Z-buffer does.

Disadvantage:

- It is slightly costly than Z-buffer method because it requires more memory in comparison to the Z-buffer method.

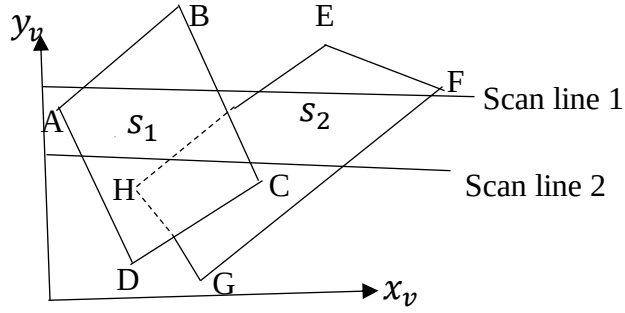
---

4. Scan-Line Method

- It is an image-space method for identifying visible surface.
- It computes and compares depth values along the various scan lines for the scene.
- Surfaces are processed using information stored in the polygon table.
- An active list of edges is formed for each scan line which stores only those edges that crosses the scan line in order of increasing 'x'.
- Also a flag is set for each surface that is set on or off to indicate whether a position along a scan line is either inside or outside the surface.
- Pixel position across each scan-line are processed from left to right.
- At the left intersection with a surface the surface flag is turned on and at the right intersection point the flag is turned off.
- We only need to perform depth calculation when more than one surface has its flag turned on at a certain scan-line position

The active list for scan line 1 contains edges AB, BC, EH & FG. Similarly for scan line 2, active list contains edges AD, EH, BC &FG.

	Edge pairs	Flags for s_1	Flags for s_2
For scan line 1	AB & BC	on	off
	BC & EH	off	off
	EH & FG	off	on
For scan line 2	AD & EH	on	off
	EH & BC	on	on
	BC & FG	off	on



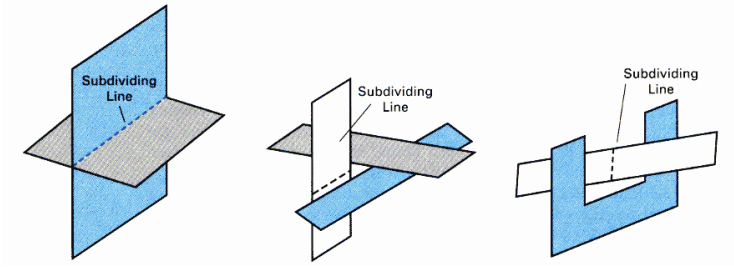
Here, between EH & BC, the flags for both surfaces are ON so depth calculation is necessary. But between other edge pairs no depth calculation is necessary.

Advantage:

- Any number of overlapping polygon surfaces can be processed with this method.

Limitation:

- The scan line method runs into trouble when surface cut through each other or otherwise cyclically overlap. Such surface need to be divided.



---

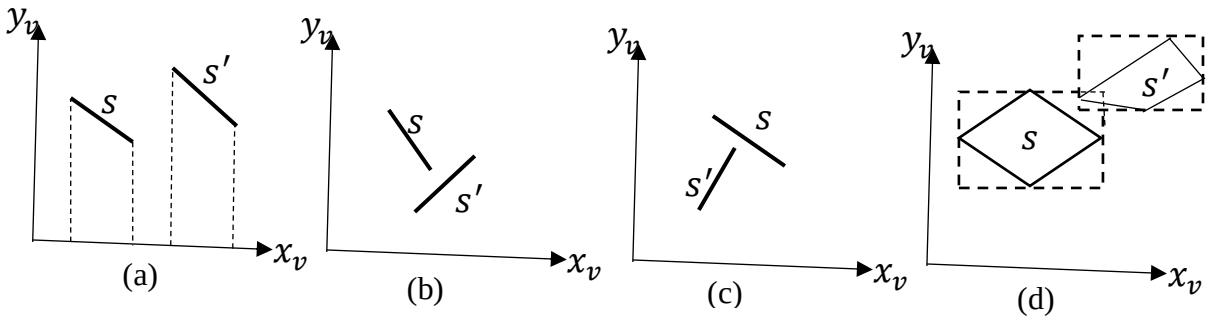
5. Depth-Sorting Method

- Also known as **Painter's Algorithm**.
- A visible surface detection method that uses both image-space and object-space operations.
- It performs the following two functions:
  a) Surfaces are stored in order of decreasing depth.
  b) Surfaces are scan-converted in order, staring with the surface of greatest depth.
- It follows following steps:
  i. All surfaces in the scene are ordered according to the smallest 'z' value on each surface.
  ii. The surface 'S' at the end of the list is then compared against all other surface to see if there are any depth overlap.
  iii. If no overlap occurs then the surface is scan converted and the process is repeated with the next surface.

When there is depth overlap with 'S' we make the following tests:

a) The bounding rectangle for the two surfaces do not overlap.

b) Surface 'S' is completely behind the overlapping surface relative to viewing position.
c) The overlapping surface is completely in front of 'S' related to viewing position.
d) The boundary edge projection of the two surfaces do not overlap.



If any of this test is true, no reordering is necessary. But if all four tests fail then we have to interchange surface S and S' in sorted list.

Drawback:

- If two or more surfaces alternately obscure each other it get into an infinite loop.

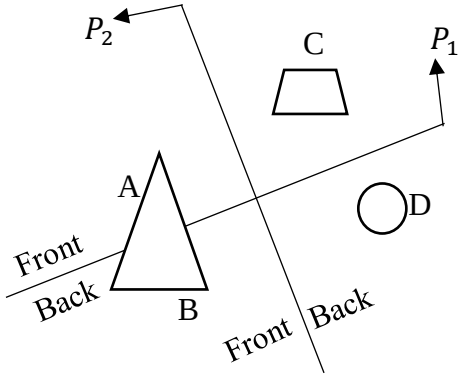
---

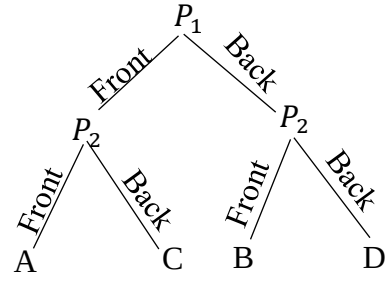
6. BSP Tree Method

Binary Space Partitioning (BSP) tree is an efficient method for determining visibility when view reference point changes but the scene are at fixed position.

- It involves identifying surfaces that are inside/front and outside/back w.r.t. the partitioning plane at each step of the space division, relative to the viewing direction.
- Start with any plane and find one set of objects behind and the rest in the front.
- In the tree, the objects are represented as terminal nodes with front objects as left branches and back objects as right branches.

This algorithm partitions the space into sub divisions until all object list contain no more than one object.





BSP tree

- Disadvantages:

- More polygon splitting may occur than in painter's algorithm.
- Appropriate partitioning hyperplane selection is quite complicated and difficult.

7. Area-Subdivision Method

In this method, the total viewing area is successively divided into smaller and smaller rectangles until each small area is simple i.e. it is a single pixel or is covered wholly by a part of a single visible surface or no surface at all.

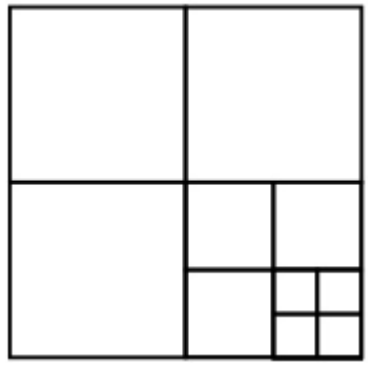


Fig: Dividing a square area into equal-sized quadrants at each step.

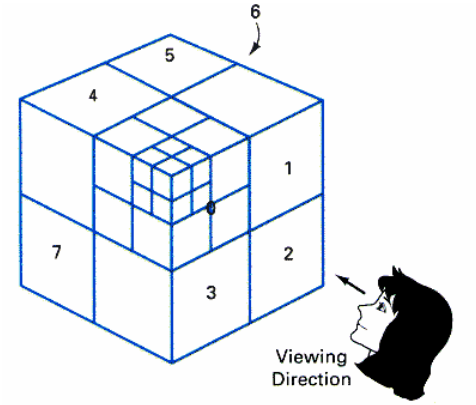
The procedure to determine whether we should subdivide an area into smaller rectangle is:

1) We first classify each of the surfaces, according to their relations with the area:
   - Surrounding surface- One that completely encloses the area.
   - Overlapping surface- One that is partly inside and partly outside the area.
   - Inside surface: One that is completely inside the area.
   - Outside surface: One that is completely outside the area.
2) Check the result from 1, if one of the following condition is true, then no further subdivision of this area is needed.
   a) All surface are outside the area.
   b) Only one surface is inside, overlapping or surrounding surface is in the area.
   c) A surrounding surface obscures all other surfaces within the area boundaries.

---

8. Octree Method

In this representation octree nodes are projected onto the viewing surface in a front-to-back order. Any surfaces toward the rear of the front octants (0, 1, 2, 3) or in the back octants (4, 5, 6, 7) may be hidden by the front surfaces.



Octants in Space

With the numbering method (0, 1, 2,3 ,4, 5, 6, 7), nodes representing octants 0, 1, 2,3 for the entire region are visited before the nodes representing octants 4, 5, 6, 7. Similarly the nodes for the front four sub-octants of octant 0 are visited before the nodes for the four back sub-octants.

When a color is encountered in an octree node, the corresponding pixel in the frame buffer is painted only if no previous color has been loaded into the same pixel position.

In most cases, both a front and a back octant must be considered in determining the correct color values for a quadrant. But

- If the front octant is homogeneously filled with some color, we do not process the back octant.
- If the front is empty, it is necessary only to process the rear octant.
- If the front octant has heterogeneous regions, it has to be subdivided and the sub-octants are handled recursively.

---

9. Ray-Casting Method

Ray tracing also known as ray casting is efficient method for visibility detection in the objects.
In this method, visibility of surface can be determined by tracing a ray of light from the center of projection (viewer's eye) to objects in the scene.

- Find out which objects the ray of light intersects.
- Then, determine which one of these objects is closer to the viewer.
- Then, set the pixel color to the object.

The ray tracing approach is an effective visibility detection method for scenes with curved surfaces, particularly spheres.

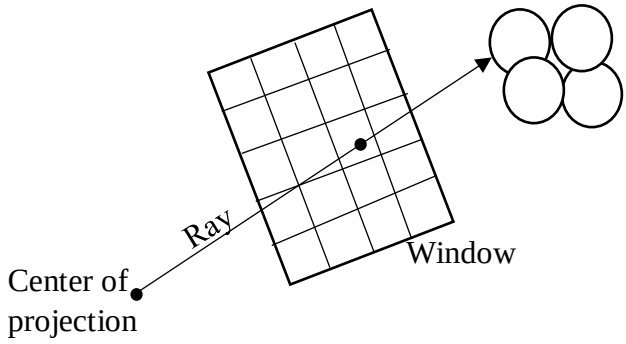


Fig: A ray is fired from the center of projection through each pixel to which the window maps, to determine the closest object intersected.

References

- **Donald Hearne and M.Pauline Baker**, "Computer Graphics, C Versions." Prentice Hall