Visible Surface Detection

By
Poonam Saini
Dept. of Computer Science & Engineering
Sir Padampat Singhania University
Udaipur

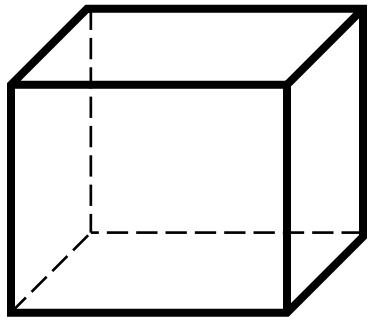
Visible Surface Detection

Visibility

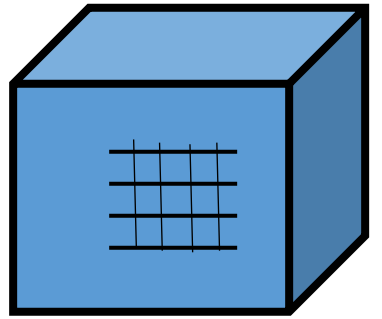
- Identify those parts of a scene that are visible from the chosen viewing position.
- Different approaches with limitations.
- Factors should be considered
  - Complexity of the scene
  - Type of object to be displayed
  - Available equipment
  - Whether static or animated displays are to be generated

Classification of visible surface detection Algorithms Visibility

❑Visible surface detection methods/ hidden surface elimination methods.



- ✓**Object space method:**
- ✓Compares objects and parts of the objects to each other within the scene
- ✓*Determine which part of the object are visible*
- ✓**Image space methods:**
- ✓Visibility is decided point by point at each pixel position on the projection plane.
- ✓*Determine per pixel which point of an object is visible*



Back surface Detection Method

- It is the fast and simple object space method for identifying the back faces of a polyhedron and is based on the inside outside test.
- Equation of a surface or Plane:

$$Ax+By+Cz+D=0$$

Inside Outside Test

- A point (x,y,z) is inside the polygon surface
  if $(Ax+By+Cz+D)< 0$
  where A, B, C, d are the plane parameters.
- When an inside point is along the line of sight to the surface, the polygon must be a back face.

Back surface Detection Method

Inside-outside test

If V . N >0 Then it is a Back Face

Where

- V is a vector in the viewing direction from the eye(camera)
- N is the normal vector to a polygon surface
- Example : Let V=(0,0,Vz) and N=Ai+Bj+Ck

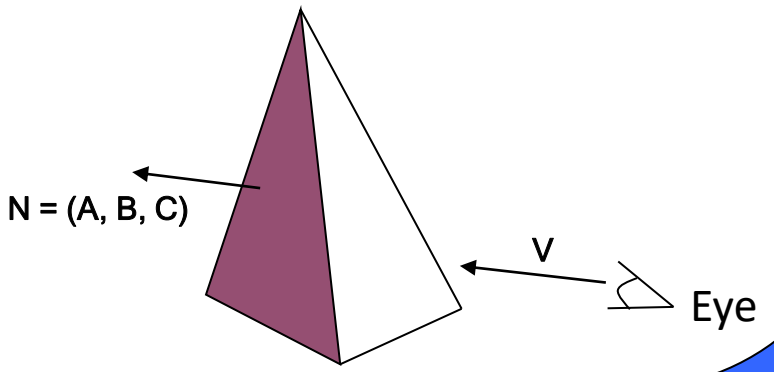
Where I, j, k be the unit vector along x, y and z axis respectively.

V=(0,0,1)

Therefore V . N = Vz . C = C

Condition for Back Face is

Sign(C)>=0



Back surface Detection Method

- To find the values of Coefficients A, B, C from vertex specification of the polygon.

Let (Xi, Yi, Zi) be the coordinates of the ith vertex

Assume total no. of vertices in the polygon = n

Then

A=$\sum_{i=1}^{n}(Yi - Yj)(Zi + Zj)$

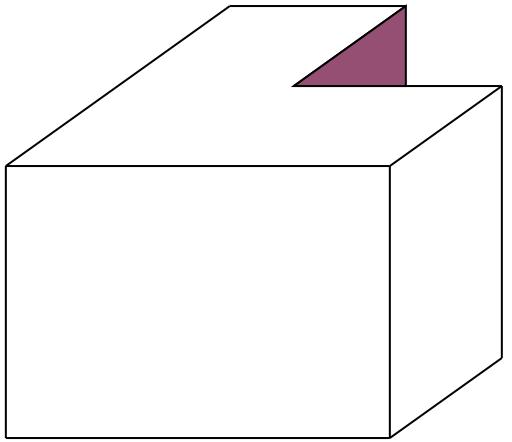
B=$\sum_{i=1}^{n}(Zi - Zj)(Xi + Xj)$

C=$\sum_{i=1}^{n}(Xi - Xj)(Yi + Yj)$

Where for i=n; j=1 otherwise j=i+1

Back surface Detection Method

- Disadvantages
  - ❖ Partially hidden surfaces are not detected by this method
  - ❖ Not useful for Ray tracing and few others
- In general, back-face removal can be expected to eliminate *about half of the surfaces* from further visibility tests



<View of a concave polyhedron with one face partially hidden by other surfaces>

Depth-Buffer/Z-Buffer Method

- It is a commonly used image space approach to detect visible surfaces and it compares surface depths at each pixel position on the projection plane.
- This procedure is also referred to as **Z-Buffer method** as the object depth is measured from the view plane along the z-axis of a viewing system.
- Each surface of a scene is processed separately, one point at a time across the surface.
- The method is usually applied to scenes containing only polygon surfaces because depth values can be computed very quickly and the method is easy to implement.

Depth-Buffer Method

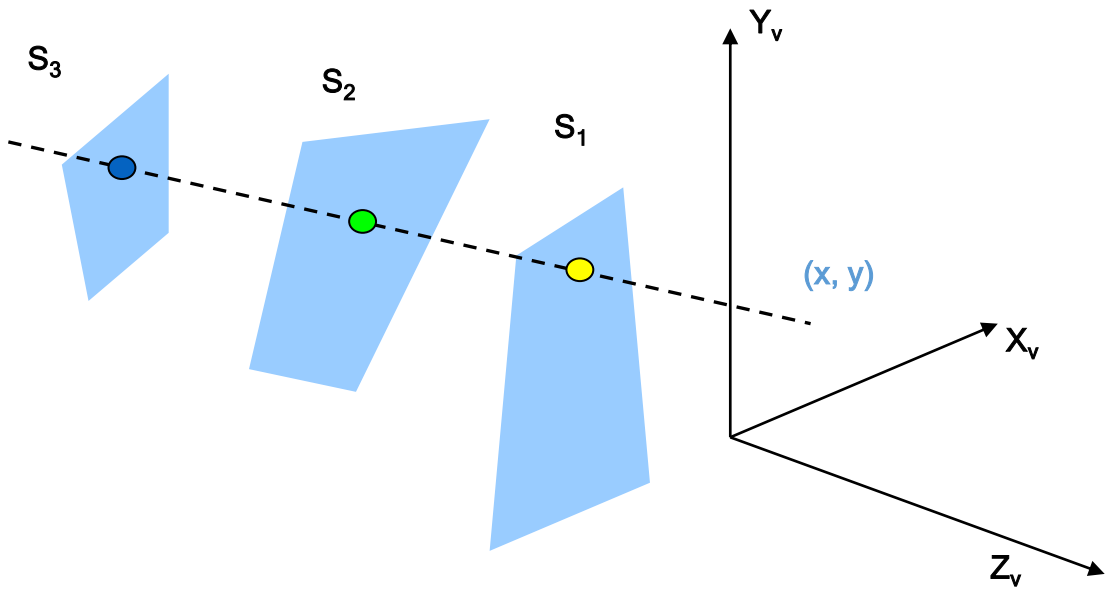
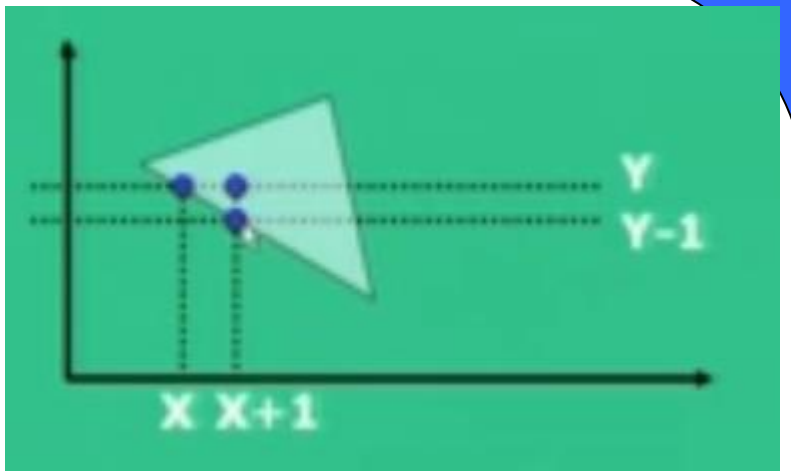


Fig: At view plane position(x,y), surfaces S1 has the smallest depth from the view plane and so is visible at the position and its surface intensity value at (x,y) is saved.

Depth Buffer & Refresh Buffer

- Two buffer areas are required
  - Depth buffer
    - Store depth values for each (X, Y) position
    - Initialize Depth(X,Y) with 0
  - Refresh buffer
    - Stores the intensity values for each position
    - Initialize Refresh(X,Y) with I_B where I_B is the background intensity

Depth Buffer Algorithm



- Initialize the depth buffer and refresh buffer

  depth(x, y) = 0, refresh(x, y) = I_B

- For each position on each polygon surface
  - Calculate the depth for each (x, y) position on the polygon
  - If z >depth(x, y), then set

    depth(x, y) = z, refresh(x, y) = $I_{surf}(x, y)$

- Depth values for a surface position(x,y) are calculated from the plane eq. for each surface:

Equation of Surface:	$AX+BY+CZ+D=0$	
Therefore	$Z_x = (-AX-BY-D)/C$	(1)
For the next position	$Z_{x+1} = (-A(X+1)-By-D)/C$	(2)
From (1) and (2)	$Z_{x+1} = Z_x-A/C$	

Example

- Given points p1(1,2,0), p2(3,6,20), p3(2,4,6) and a view point C(0,0,-10). Determine which points obscure the others when viewed from C.

Solution:

Let us find the line joining the viewpoint C and point p1 then finding which points lie on this line

Use parametric line equation for Cp1

$x = x0+(x1-x0)t = 0+(1-0)t = t$ (1)

$y = y0+(y1-y0)t = 0+(2-0)t = 2t$ (2)

$z = z0+(z1-z0)t = -10+(0+10)t = -10+10t$ (3)

Example (contd..)

- Given points p1(1,2,0), p2(3,6,20), p3(2,4,6) and a view point C(0,0,-10). Determine which points obscure the others when viewed from C.

Solution (Contd…)

- To determine p2(3,6,20) lies on line CP or not

If x=3 then t=3 Therefore from equation (2) and (3)

y=2t=6 and z=-10+10t=20

Therefore p2 lies on the projection line through c and p1

- To determine for p3(2,4,6)

x=2 then t=2

But y=4, z=10

Therefore p3 is not on the projection line.

Example (contd..)

- Given points p1(1,2,0), p2(3,6,20), p3(2,4,6) and a view point C(0,0,-10). Determine the point which is in front w.r.t. C
- Solution(Contd..)

As C occurs on the line at t=0

P1 occurs at t=1 and p2 occurs at t=3

By comparing t values C< p1<p2

Therefore p1 is in front of p2 w.r.t. C and p1 obscure p2

As p3 is not on this projection line so that it is neither obscure nor is obscured by p1 and p2

A-Buffer method

- A drawback of the depth buffer method is that it can only find one visible surface at each pixel position.
- It can not accumulate intensity values for more than one surface as is necessary if transparent surface are to be displayed.

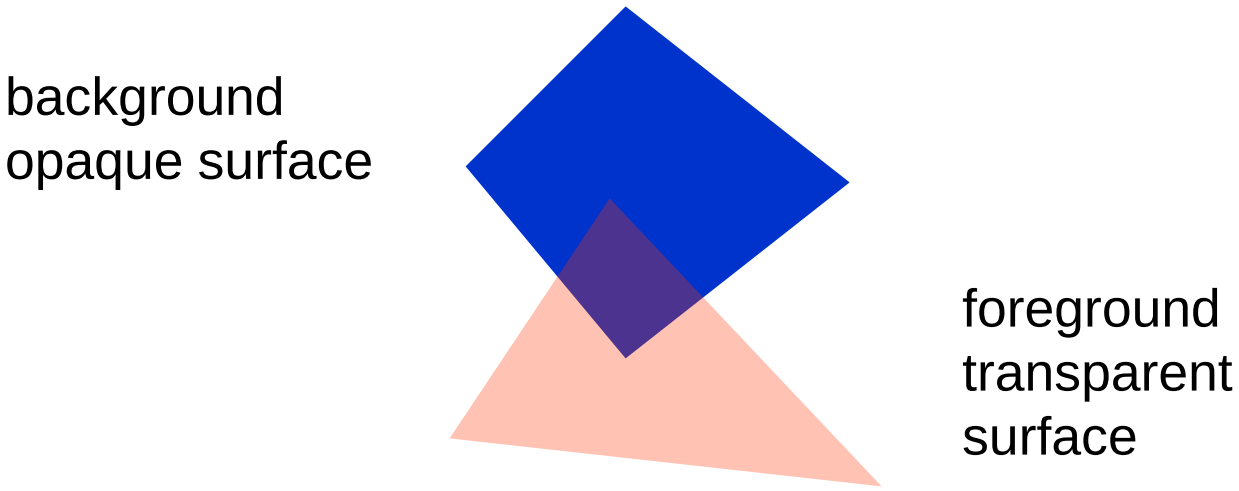


Fig: Viewing an opaque surface through a transparent surface requires multiple surface intensity contributions for pixel position.

A-Buffer method

- The A-Buffer method expands the depth buffer so that each position in the buffer can reference a linked list of surfaces.
- Thus more than one intensity can be taken into consideration at each pixel position. Each position in A-buffer has two fields:
  - ✓ Depth Field: It stores the real no.
  - ✓Intensity Field: It stores surface intensity values.
- If depth field is positive, the no. stored at that position is the depth of a single surface overlapping the corresponding pixel area.
- If depth field is negative, this indicates multiple surface contributions to the pixel intensities. The intensity field then stores a pointer to a linked list of surface data.

Scan Line Algorithm Steps

Step 1: The edges of all the polygons forming the object are processed into the edge table. Two arrays are used as depth[I,j] and intensity[i,j].

Step 2: Use XY algorithm(i.e. sort the pixel coordinates first on minimum Y value and then on maximum X value) and for each polygon in the object find all pixels on the current scan line I which lie within the polygons. For each such j values:

(a) Find the depth z of polygon at (i,j).

(b) If z<depth[i,j] then depth[i,j] is the intensity of the polygon under consideration.

Step 3: After processing all the polygons the array intensity[i,j] can be saved in the frame buffer.

Area Subdivision/Warnock's Algorithm

- It is an image space method but it is also use the object space method for depth comparison.
- This algorithm makes use of area coherence and is first given by Warnock.
- This method is resolution dependent.
- The method is applied by successively dividing the viewport area into smaller and smaller area until every small area entirely contain a polygon projected on viewport area or no polygon at all.
- This dividing process also stops when the are becomes the smallest size i.e. pixel.

Area Subdivision/Warnock's Algorithm Steps

Step 1: Initialize the area to be subdivided as the given viewport or the entire screen.

Step 2: Sort the polygon on minimum depth value and create a visible polygon list.

Step 3: Following tests are applied for the visibility of the polygon:

(a) If all the polygons forming an object projected on the screen fall on the outside then the sampling area is set with the background color.

(b) If only one polygon in the visible list and inside the sampling area then draw the polygon and give intensity with **desired values to each pixel which lies inside, other pixels set to background color.**

Area Subdivision/Warnock's Algorithm Steps

(c) If only one polygon in probable visible list and the polygon completely contains the sample area then color the entire sample area by intensity values of each pixel on the polygon.

(d) If the surrounding polygon is the closest polygon to the sample area then draw the polygon, color it with its color value.

(e) If the area under consideration is pixel then compute the z value of all the polygons in the polygon visible list for this pixel co-ordinates and set the color of that polygon which will have minimum z value.

Step 4: If no condition in step 3 is true then subdivide the area into four equal parts and for each area repeat steps 2 and 3.