RAMA UNIVERSITY

www.ramauniversity.ac.in

FACULTY OF ENGINEERING & TECHNOLOGY

BCS -504 Computer Graphics & Multimedia

Lecture-19

Mr. Dilip Kumar J Saini
Assistant Professor
Computer Science & Engineering

OUTLINE

- BACK FACE REMOVAL ALGORITHM
- BACK FACE REMOVED ALGORITHM
- Z-BUFFER ALGORITHM
- LIMITATIONS OF DEPTH BUFFER

Back Face Removal Algorithm

It is used to plot only surfaces which will face the camera. The objects on the back side are not visible. This method will remove 50% of polygons from the scene if the parallel projection is used. If the perspective projection is used then more than 50% of the invisible area will be removed. The object is nearer to the center of projection, number of polygons from the back will be removed.

It applies to individual objects. It does not consider the interaction between various objects. Many polygons are obscured by front faces, although they are closer to the viewer, so for removing such faces back face removal algorithm is used.

When the projection is taken, any projector ray from the center of projection through viewing screen to object pieces object at two points, one is visible front surfaces, and another is not visible back surface.

This algorithm acts a preprocessing step for another algorithm. The back face algorithm can be represented geometrically. Each polygon has several vertices. All vertices are numbered in clockwise. The normal M_1 is generated a cross product of any two successive edge vectors. M_1represent vector perpendicular to face and point outward from polyhedron surface

$N_1=(v_2-v_1)(v_3-v_2)$

If $N_1.P \geq 0$ visible

$N_1.P<0$ invisible

Back Face Removed Algorithm

Repeat for all polygons in the scene.

1. Do numbering of all polygons in clockwise direction i.e.
   v_1 v_2 v_3.....v_z
2. Calculate normal vector i.e. N_1
   $N_1=(v_2-v_1)*(v_3-v_2)$
3. Consider projector P, it is projection from any vertex
   Calculate dot product
   Dot=N.P
4. Test and plot whether the surface is visible or not.
   If Dot ≥ 0 then
   surface is visible
   else
   Not visible

Z-Buffer Algorithm

It is also called a **Depth Buffer Algorithm**. Depth buffer algorithm is simplest image space algorithm. For each pixel on the display screen, we keep a record of the depth of an object within the pixel that lies closest to the observer. In addition to depth, we also record the intensity that should be displayed to show the object. Depth buffer is an extension of the frame buffer. Depth buffer algorithm requires 2 arrays, intensity and depth each of which is indexed by pixel coordinates (x, y).

Z-Buffer Algorithm

Algorithm

1. For all pixels on the screen, set depth [x, y] to 1.0 and intensity [x, y] to a background value.

2. For each polygon in the scene, find all pixels (x, y) that lie within the boundaries of a polygon when projected onto the screen. For each of these pixels:

(a) Calculate the depth z of the polygon at (x, y)

(b) If z < depth [x, y], this polygon is closer to the observer than others already recorded for this pixel. In this case, set depth [x, y] to z and intensity [x, y] to a value corresponding to polygon's shading. If instead z > depth [x, y], the polygon already recorded at (x, y) lies closer to the observer than does this new polygon, and no action is taken.

3. After all, polygons have been processed; the intensity array will contain the solution.

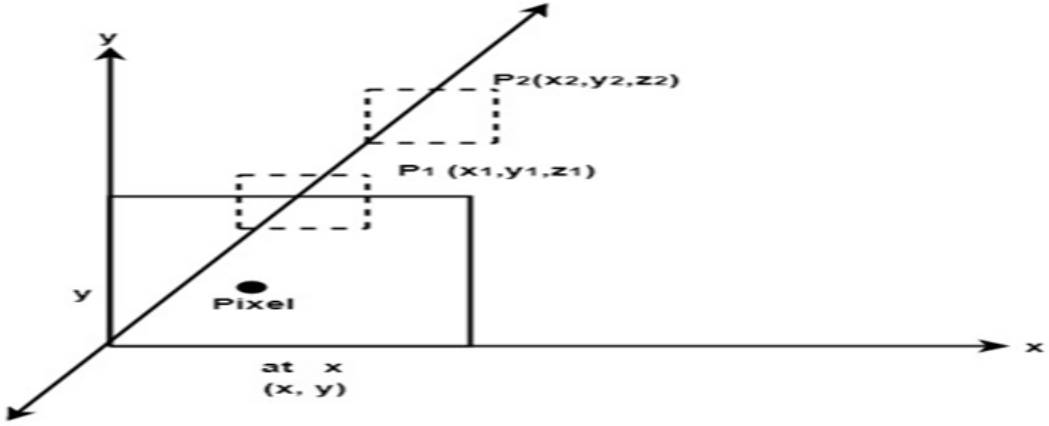
4. The depth buffer algorithm illustrates several features common to all hidden surface algorithms.

Z-Buffer Algorithm

5. First, it requires a representation of all opaque surface in scene polygon in this case.

6. These polygons may be faces of polyhedral recorded in the model of scene or may simply represent thin opaque 'sheets' in the scene.

7. The IInd important feature of the algorithm is its use of a screen coordinate system. Before step 1, all polygons in the scene are transformed into a screen coordinate system using matrix multiplication.



Limitations of Depth Buffer

1. The depth buffer Algorithm is not always practical because of the enormous size of depth and intensity arrays.
2. Generating an image with a raster of 500 x 500 pixels requires 2, 50,000 storage locations for each array.
3. Even though the frame buffer may provide memory for intensity array, the depth array remains large.
4. To reduce the amount of storage required, the image can be divided into many smaller images, and the depth buffer algorithm is applied to each in turn.
5. For example, the original 500 x 500 faster can be divided into 100 rasters each 50 x 50 pixels.
6. Processing each small raster requires array of only 2500 elements, but execution time grows because each polygon is processed many times.
7. Subdivision of the screen does not always increase execution time instead it can help reduce the work required to generate the image. This reduction arises because of coherence between small regions of the screen.

Multiple Choice Question

MUTIPLE CHOICE QUESTIONS:

Sr no	Question	Option A	Option B	OptionC	OptionD
1	Virtual reality is the system which produce ……. in such a way that we feel that our surrounding is what we are set in display devices	scene	images	picture definition	none of these
2	In virtual reality user can step into a scene and interact with the……..	environment	area	rotation	projection
3	Virtual reality can also be produce with ……..... instead of head set	stereoscopic glass	video monitor	both a & b	none of these
4	Virtual reality provides…….. cost virtual reality system.	high	low	both a & b	none of these
5	Raster graphics systems having additional processing unit like ……	video controller	display controller	both a & b	none of these

REFERENCES

- http://www.engppt.com/search/label/Computer%20Graphics