Visible Surface Detection Methods

(Hidden surface elimination)

Visible surface detection or Hidden surface removal is major concern for realistic graphics for identifying those parts of a scene that are visible from a choosen viewing position. Several algorithms have been developed. Some require more memory, some require more processing time and some apply only to special types of objects.

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

These two approaches are

- **Object-Space methods:** Compares objects and parts of objects to each other within the scene definition to determine which surface as a whole we should label as visible.

- **Image-Space methods**: 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.

Differentiate between Object space and Image space method

Object Space	Image Space
1. Image space is object based. It concentrates on geometrical relation among objects in the scene.	1. It is a pixel-based method. It is concerned with the final image, what is visible within each raster pixel.
2. Here surface visibility is determined.	2. Here line visibility or point visibility is determined.
3. It is performed at the precision with which each object is defined, No resolution is considered.	3. It is performed using the resolution of the display device.
4. Calculations are not based on the resolution of the display so change of object can be easily adjusted.	4. Calculations are resolution base, so the change is difficult to adjust.
5. These were developed for vector graphics system.	5. These are developed for raster devices.
6. Object-based algorithms operate on continuous object data.	6. These operate on object data.
7. Vector display used for object method has large address space.	7. Raster systems used for image space methods have limited address space.
8. Object precision is used for application where speed is required.	8. There are suitable for application where accuracy is required.
9. It requires a lot of calculations if the image is to enlarge.	9. Image can be enlarged without losing accuracy.
10. If the number of objects in the scene increases, computation time also increases.	10. In this method complexity increase with the complexity of visible parts.

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" and applied to each side after any rotation of the object takes place. It is commonly known as back-face detection of a polyhedron is based on the "inside-outside" tests.

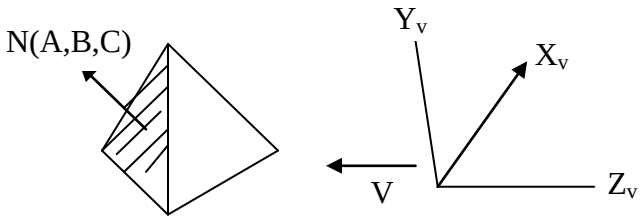
A point (x, y, z) is inside a polygon surface if

$Ax + By + Cz + D < O$

We can simplify this test by considering the normal vector N to a polygon surface which has Cartesian components (A, B, C)

If V is the vector in viewing direction from the eye position then this polygon is a back face if,

V.N>0



In the equation $Ax + By + Cz + D = 0$, if A,B,C remains constant, then varying value of D results in a whole family of parallel planes. One of which (D = 0) contains the origin of the co-ordinates system and ,

If D > 0, plane is behind the origin(Away from observer)

If D < 0, plane is in front of origin(towards the observer)

If we clearly defined our object to have centered at origin, the all those surface that are viewable will have negative D and unviewable surface have positive D.

So , simply our hidden surface removal routine defines the plane corresponding to one of 3D surface from the co-ordinate of 3 points on it and computing D , visible surface are detected.

DEPTH-BUFFER-METHOD:

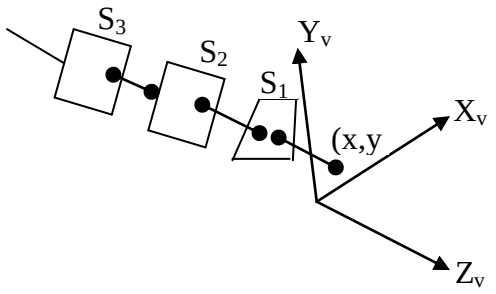
Depth Buffer Method is the commonly used image-space method for detecting visible surface. It is also know as z-buffer method. It compares surface depths at each pixel position on the projection plane. It is called z-buffer method since object depth is usually measured from the view plane along the z-axis of a viewing system.

Each surface of 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 method is easy to implement. This method can be apply to non planer surfaces.

With object description converted to projection co-ordinates, each (x, y, z) position on polygon surface corresponds to the orthographic projection point (x, y) on the view plane. Therefore for each pixel position (x, y) on the view plane, object depth is compared by z. values.

With objects description converted to projection co-ordinates, each (X,Y,Z) position on polygon surface correspond to the orthographic projection point (X,Y) on the view plane. The object depth is compared by Z-values.

In figure, three surface at varying distance from view plane X_vY_v, the projection along (x,y) surface S_1 is closest to the view-plane so surface intensity value of S_1 at (x,y) is saved.



In Z-buffer method, two buffers area are required. A depth buffer is used to store the depth value for each (x,y) position or surface are processed, and a refresh buffer stores the intensity value for each position. Initially all the position in depth buffer are set to 0, and refresh buffer is initialize to background color. Each surface listed in polygon table are processed one scan line at a time, calculating the depth (z-val) for each position (x,y). The calculated depth is compared to the value previously stored in depth buffer at that position. If calculated depth is greater than stored depth value in depth buffer, new depth value is stored and the surface intensity at that position is determined and placed in refresh buffer.

Algorithm: Z-buffer

1. Initialize depth buffer and refresh buffer so that for all buffer position (x,y)
   depth (x,y) =0, refresh (x,y) = $I_{background.}$
2. For each position on each polygon surface, compare depth values to previously stored value in depth buffer to determine visibility.
   - Calculate the depth Z for each (x,y) position on polygon
   - If Z> depth (x,y) then
     depth (x,y) =Z
     refresh (x,y) = $I_{surface}$ (x,y)

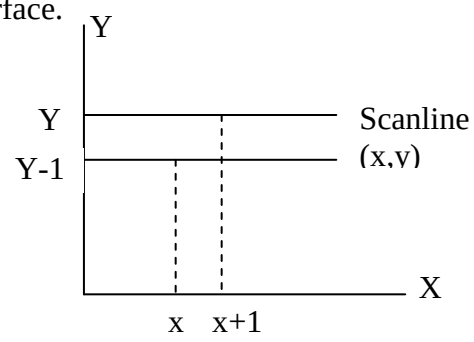
Where $I_{background}$ is the intensity value for background and $I_{surface}$ (x,y) is intensity value for surface at pixel position (x,y) on projected plane. After all surface are processed, the depth buffer contains the depth value of the visible surface and refresh buffer contains the corresponding intensity values for those surface. The depth value of the surface position (x,y) are calculated by plane equation of surface.

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

Let Depth Z' at position (x+1,y)

$$Z' = \frac{-A(x+1) - By - D}{C}$$

$$\Rightarrow Z' = Z - A/C \quad \text{———} \quad (1)$$



$-A/C$ is constant for each surface so succeeding depth value across a scan line are obtained form preceding values by simple calculation.

Advantages

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

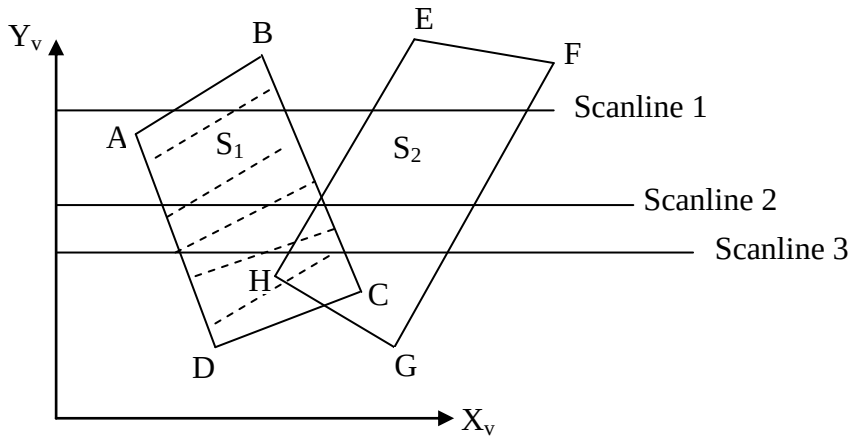
Disadvantages

- It requires large memory.
- It is time consuming process.

SCAN LINE METHOD :

This is image-space method for removing hidden surface which is extension of the scan line polygon filling for polygon interiors. Instead of filling one surface we deal with multiple surface here.

As each scan line is processed, all polygon surface intersecting that line are examined to determine which are visible. We assume that polygon table contains the co-efficient of the plane equation for each surface as well as vertex edge, surface information, intensity information for the surface, and possibly pointers to the edge table.



- In figure above, the active edge list for scan line 1 contains information from edge table for edge AB, BC, EH, FG.
- For positions along this scan line between edge AB and BC, only the flag for surface S_1 is on
- Therefore no depth calculation must be made using the plane coefficients for two surface and intensity information for surface S_1 is entered from the polygon table into the refresh buffer.

→ Similarly between EH&FG. Only the flag for S_2 is on. No other positions along scan line 1 intersect surface. So intensity values in the other areas are set to background intensity

→ For Scanline 2 & 3, the active edge list contains edges AD, EH BC, FG. Along scaline 2 from edge AD, to edge EH only surface flag for S_1 is on, but between edges EH& BC, the flags for both surface is on. In this interval, depth calculation is made using the plane coefficients for the two surfaces.

For example, if Z of surface S_1 is less than surface S_2, So the intensity of S_1 is loaded into refresh buffer until boundary BC is encountered. Then the flag for surface S_1 goes off and intensities for surface S_2 are stored until edge FG is passed.

Any no of overlapping surface are processed with this scan line methods.

DEPTH SORTING METHOD:

This method uses both object space and image space method. In this method the surface representation of 3D object are sorted in of decreasing depth from viewer. Then sorted surface are scan converted in order starting with surface of greatest depth for the viewer.

The conceptual steps that performed in depth-sort algorithm are

1. Sort all polygon surface according to the smallest (farthest) Z co-ordinate of each.

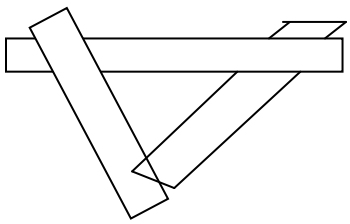
2. Resolve any ambiguity this may cause when the polygons Z extents overlap, splitting polygons if necessary.

3. Scan convert each polygon in ascending order of smaller Z-co-ordinate i.e. farthest surface first (back to front)

In this method, the newly displayed surface is partly or completely obscure the previously displayed surface. Essentially, we are sorting the surface into priority order such that surface with lower priority (lower z, far objects) can be obscured by those with higher priority (high z-value).

This algorithm is also called "**Painter's Algorithm**" as it simulates how a painter typically produces his painting by starting with the background and then progressively adding new (nearer) objects to the canvas.

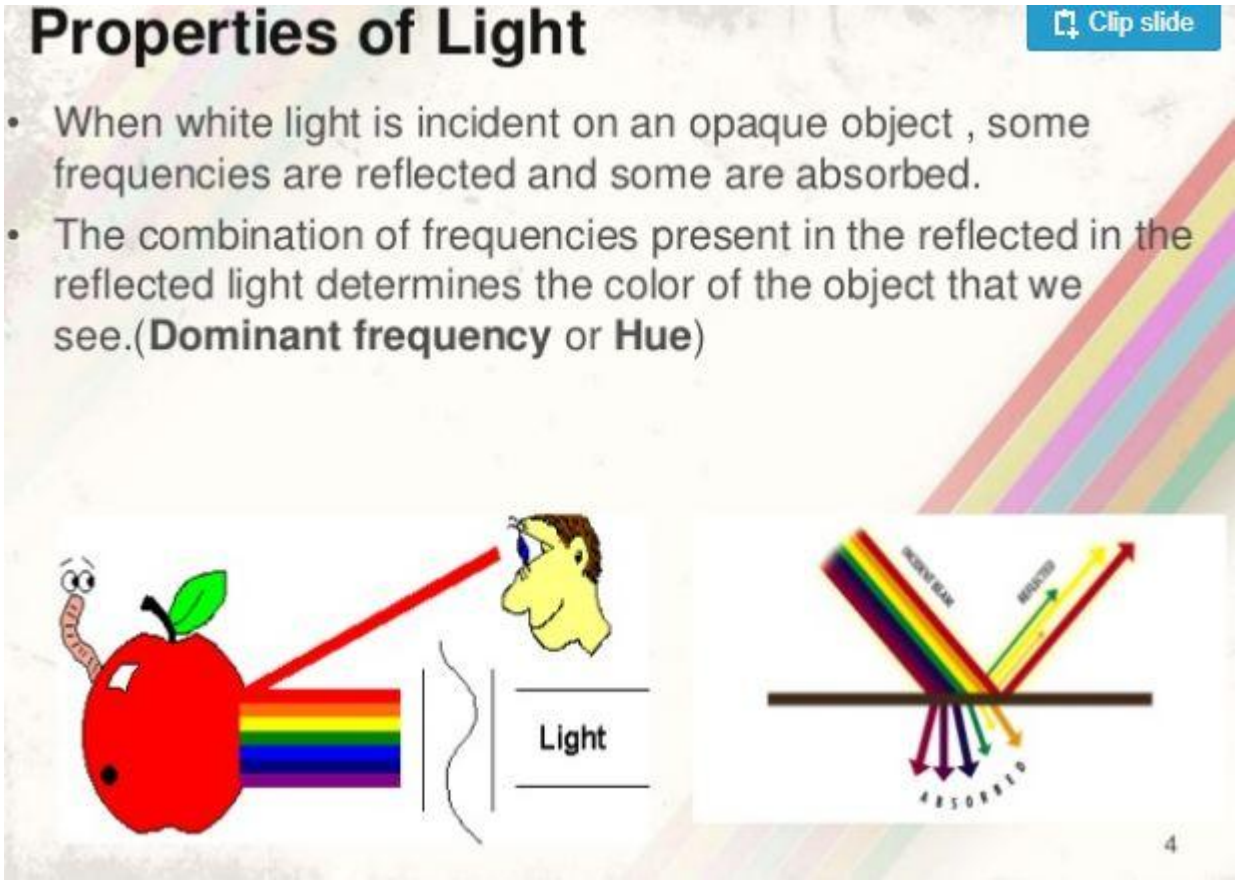
Problem: One of the major problem in this algorithm is intersecting polygon surfaces. As shown in fig. below.



- Different polygons may have same depth.
- The nearest polygon could also be farthest.

We cannot use simple depth-sorting to remove the hidden-surfaces in the images.

Solution: For intersecting polygons, we can split one polygon into two or more polygons which can then be painted from back to front. This needs more t6ime to compute intersection between polygons. So it becomes complex algorithm for such surface existence.



Properties of Light

- In the wave model of electromagnetic radiation, Light can be described as oscillating transverse electric & magnetic fields propagating through space.

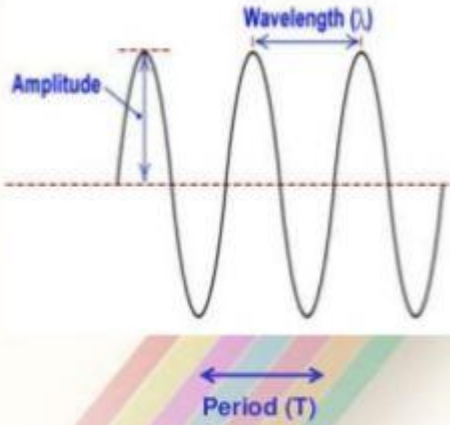
For each Spectral Color :

- Rate of oscillation = frequency = f
- Time between any two consecutive positions on the wave that have same amplitude = Period of wave= T
- Period is the inverse of frequency

$$T = 1/f$$

- Distance travelled from beginning of one oscillation to the beginning of the next oscillation = wavelength = λ
- The wavelength and frequency are inversely proportional to each other with the proportionality constant as the speed of light (c)

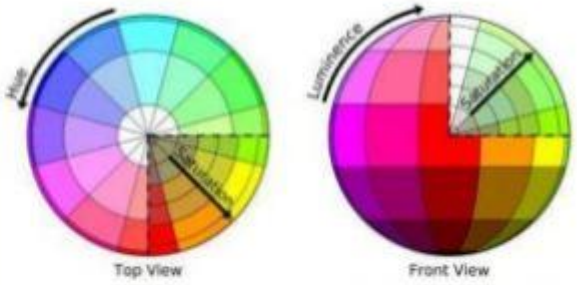
$$c = \lambda f$$



5

Characteristics of Color

1. Dominant Frequency (Hue)
   The color we see (red, green, purple).



2. Brightness
   The total light energy, how bright is the color (How bright are the lights illuminating the object?)

3. Purity (Saturation)
   Purity describes how close a light appears to be to a pure spectral color, such as pink is less saturated than red.

Chromaticity refers to the two properties (purity & hue) together.

6

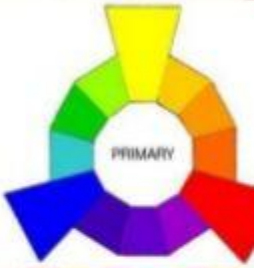
Color Model

•A color model is an abstract mathematical model describing the way colors can be represented as tuples of numbers, typically as three or four values or color components. *[Wikipedia]*

•Any method for explaining the properties or behavior of color within some particular context is called a Color Model.*[Hearn, Baker ,computer graphics with OpenGL]*

Color Model

Primary Colors
Sets of colors that can be combined to make a useful range of colors



Color Gamut
Set of all colors that we can produce from the primary colors.

Complementary Colors
Pairs of colors which, when combined in the right proportions, produce white.

Example, in the RGB model: red & cyan , green & magenta , blue & yellow.

•No finite set of real primary colors can be combined to produce all possible visible colors.

•However, given a set of three primary colors, we can characterize any fourth color using color-mixing processes.

8

Color Model

ADDITIVE

SUBTRACTIVE

Additive color

Uses light to display color. Mixing begins with black and ends with white; as more color is added, the result is lighter and tends to white. Used for computer displays

Example: The RGB colors are light primaries and colors are created with light.

A subtractive color

Uses ink to display color. Mixing means that one begins with white and ends with black; as one adds color, the result gets darker and tends to black. Used for printed material

It is called 'subtractive' because its wavelength is less than sum of the wavelengths of its constituting colors.

Example: The CMYK color system is the color system used for printing.

10

RGB Model

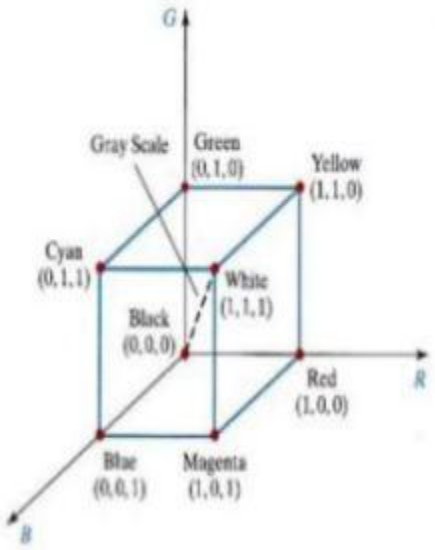
•The red, green, and blue (RGB) color space is widely used throughout computer graphics.

•Additive Color Model.

•Unit Cube defined on R, G & B axes.

•The Origin (0,0,0) represents black and the diagonally opposite vertex (1,1,1) is White.

•Vertices of the cube on the axes represent primary colors, and the remaining vertices are the complementary color points for each of the primary colors.



•Shades of gray are represented along the main diagonal.

16

YIQ model

•YIQ model is used for US TV broadcast.

•This model was designed to separate chrominance (I and Q) from luminance (Y).

•This was a requirement in the early days of color television when black-and-white sets still were expected to pick up and display what were originally color pictures

•The Y-channel contains luminance information (sufficient for black-and-white television sets) while the I and Q channels carried the color information.

YIQ model

•A color television set would take these three channels, Y, I, and Q, and map the information back to R, G, and B levels for display on a screen.

•The advantage of this model is that more bandwidth can be assigned to the Y-component (luminance) because the human visual system is more sensitive to changes in luminance than to changes in hue or saturation

Convert From RGB To YIQ

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.595716 & -0.274453 & -0.321263 \\ 0.211456 & -0.522591 & 0.311135 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Convert From YIQ To RGB

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & 0.9563 & 0.6210 \\ 1 & -0.2721 & -0.6474 \\ 1 & -1.1070 & 1.7046 \end{bmatrix} \begin{bmatrix} Y \\ I \\ Q \end{bmatrix}$$

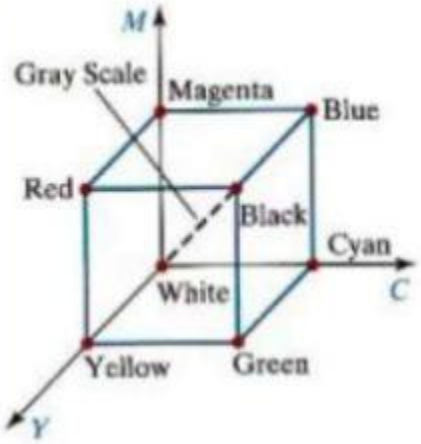
CMY and CMYK Model

•Subtractive Color Model.

•Stands for cyan-magenta-yellow.

•Used for hardcopy devices (ex. Printers).

•A printed color that looks red absorbs the other two components G and B and reflects R.



•Thus the C-M-Y coordinates are just the complements of the R-G-B coordinates.

CMY and CMYK Model

In additive color models such as RGB, white is the "additive" combination of all primary colored lights, while black is the absence of light.

In the CMYK model, it is the opposite: white is the natural color of the paper or other background, while black results from a full combination of colored inks.

RGB To CMY

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

CMY To RGB

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} C \\ M \\ Y \end{bmatrix}$$

Color Models Applications

Color Model Application Area	Color Model Application Area
RGB	- Computer graphics - Image processing - Image Analysis - Image Storage
CMY(K)	Printing
HSV, HSL	- Human visual perception - Computer graphics processing - Computer Vision - Image Analysis - Design image - Human vision - Image editing software - Video editor
YIQ	- TV broadcasting - Video system