Chapter -4
Visible surface Detection and Surface Rendering

Visible-Surface Detection Methods:

To identify those parts of a scene that is visible from a chosen viewing position.Surfaces which is obscured by other opaque surfaces along the line of projection are invisible to the viewer.

Characteristics of approaches:

- Require large memory size?

- Require long processing time?

- Applicable to which types of objects?

Considerations:

- Complexity of the scene

- Type of objects in the scene

-Available equipment

Classification of Visible-Surface Detection Algorithms:

Two methods:
1. Object space method
2. Image space method

Object-space Methods

Compare objects and parts of objects to each other within the scene definition to determine which surfaces, as a whole, we should label as visible: For each object in the scene we process the image as below.

1. Determine those part of the object whose view is unobstructed by other parts of it or any other object with respect to the viewing specification.

2. Draw those parts in the object color.

- Compare each object with all other objects to determine the visibility of the object parts.

- If there are n objects in the scene, complexity = O(n2)

- Calculations are performed at the resolution in which the objects are defined (only limited by the computation hardware).

- Process is unrelated to display resolution or the individual pixel in the image and the result of the process is applicable to different display resolutions.

- Display is more accurate but computationally more expensive as compared to image space methods because step 1 is typically more complex, eg. Due to the possibility of intersection between surfaces.

- Suitable for scene with small number of objects and objects with simple relationship with each other.

Image-space Methods (Mostly used)

Visibility is determined point by point at each pixel position on the projection plane. For each pixel in the image we do

1. Determine the object closest to the viewer that is pierced by the projector through the pixel
2. Draw the pixel in the object color.

- For each pixel, examine all n objects to determine the one closest to the viewer.
- If there are p pixels in the image, complexity depends on n and p (O(np)).
- Accuarcy of the calculation is bounded by the display resolution.
- A change of display resolution requires re-calculation

Back-Face Detection Method

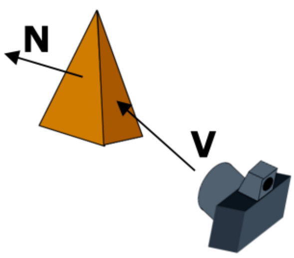
In a solid object, there are surfaces which are facing the viewer (front faces) and there are surfaces which are opposite to the viewer (back faces). These back faces contribute to approximately half of the total number of surfaces. Since we cannot see these surfaces anyway, to save processing time, we can remove them before the clipping process with a simple test. Each surface has a normal vector. If this vector is pointing in the direction of the center of projection, it is a front face and can be seen by the viewer. If it is pointing away from the center of projection, it is a back face and cannot be seen by the viewer. The test is very simple, if the z component of the normal vector is positive, then, it is a back face. If the z component of the vector is negative, it is a front face. Note that this technique only caters well for non overlapping convex polyhedral. For other cases where there are concave polyhedron or overlapping objects, we still need to apply other methods to further determine where the obscured faces are partially or completely hidden by other objects (eg.Using Depth-Buffer Method or Depth-sort Method).

A point (x, y,z) is "inside" a polygon surface with plane parameters A, B, C, and D if

$$A_x + B_y + C_z + D < 0 \qquad \text{----------------(1)}$$

When an inside point is along the line of sight to the surface, the polygon must be a back face .

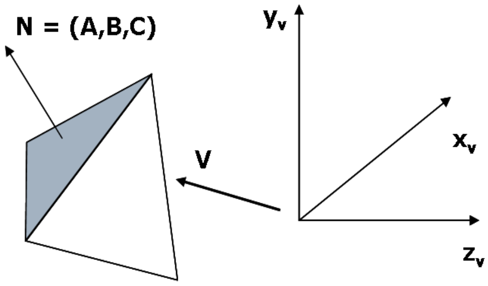
We can simplify this test by considering the normal vector N to a polygon surface, which has Cartesian components *(A, B, C).* In general, if V is a vector in the viewing direction from the eye position, as shown in Fig.,



then this polygon is a back face if V . N > 0

Furthermore, if object descriptions have been converted to projection coordinates and our viewing direction is parallel to the viewing z_v. axis, then $V = (0, 0, V_z)$ and $V . N = V_zC$ so that we only need to consider the sign of C, the ; component of the normal vector N.

In a right-handed viewing system with viewing direction along the negative z_v axis in the below Fig. the polygon is a back face if C < 0.



Thus, in general, we can label any polygon as a back face if its normal vector has a z component value

$$C <= 0$$

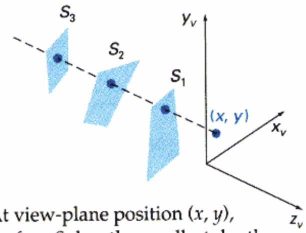
By examining parameter C for the different planes defining an object, we can immediately identify all the back faces.

- Convex polyhedron?→ no problem because either completely visible or completely hidden
- Concave polyhedron?→ some problem; needs additional faces
- More than one objects? in scene → problem
- Eliminates about half of the polygon surface in a scene from further visibility tests

Depth-Buffer Method (Z-Buffer Method)

This approach compare surface depths at each pixel position on the projection plane. This procedure is also referred to as the z-buffer method.

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. But the method can be applied to non planar surfaces. With object descriptions converted to projection coordinates, each (x, y, z) position on a 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 depths can be compared by comparing z values. The figure shows three surfaces at varying distances along the orthographic projection line from position (x,y) in a view plane taken as the (x_v,y_v) plane. Surface S_1, is closest at this position, so its surface intensity value at (x, y) is saved.



At view-plane position (x, y), surface S_1 has the smallest depth from the view plane and so is visible at that position.

Object depth is usually measured from the view plane along the z axis of a viewing system. This method requires 2 buffers: one is the image buffer and the other is called the z-buffer (or the depth buffer). Each of these buffers has the same resolution as the image to be captured. As surfaces are processed, the image buffer is used to store the color values of each pixel position and the z-buffer is used to store the depth values for each (x,y) position.

Algorithm:

1. Initially each pixel of the z-buffer is set to the maximum depth value (the depth of the back clipping plane).

   i.e depth (x, y)=0

2. The image buffer is set to the background color.

   i.e refresh(x , y)=$I_{backgnd}$

3. Surfaces are rendered one at a time.

4. For the first surface, the depth value of each pixel is calculated.

5. If this depth value is smaller than the corresponding depth value in the z-buffer (ie. it is closer to the view point), both the depth value in the z-buffer and the color value in the image buffer are replaced by the depth value and the color value of this surface calculated at the pixel position. i.e

   If z > depth(x, y), then set

   depth (x, y)=z

   refresh(x,y)= I_{surf}(x, y)

6. Repeat step 4 and 5 for the remaining surfaces.

7. After all the surfaces have been processed; each pixel of the image buffer represents the color of a visible surface at that pixel. This method requires an additional buffer (if compared with the Depth-Sort Method) and the overheads involved in updating the buffer. So this method is less attractive in the cases where only a few objects in the scene are to be rendered.

- Simple and does not require additional data structures.
- The z-value of a polygon can be calculated incrementally.
- No pre-sorting of polygons is needed.
- No object-object comparison is required.
- Can be applied to non-polygonal objects.
- Hardware implementations of the algorithm are available in some graphics workstation.
- For large images, the algorithm could be applied to, e.g., the 4 quadrants of the image separately, so as to reduce the requirement of a large additional buffer.

<u>Calculation of depth values :</u>

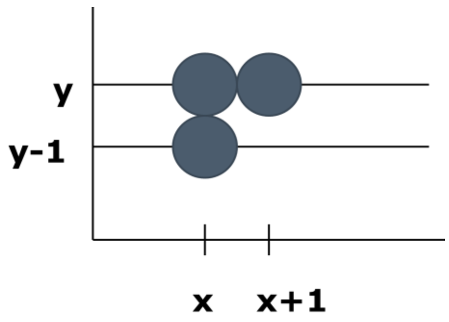
Depth values for a surface position (x, y) are calculated from the plane equation for each surface:

$$z = \frac{-Ax - By - D}{C} \qquad \text{......................}(1)$$

For any scan line adjacent horizontal positions across the line differ by1, and a vertical y value on an adjacent scan line differs by 1. If the depth of position(x, y) has been determined to be z, then the depth z' of the next position (x +1, y) along the scan line is obtained from Eq. (1) as

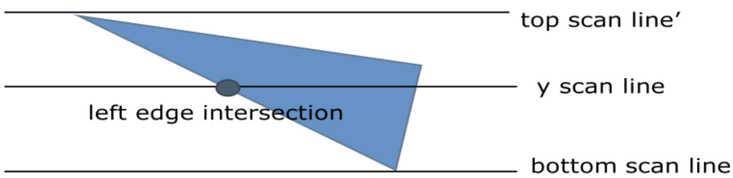
$$z' = \frac{-A(x+1) - By - D}{C} \qquad \text{..}(2)$$

$$z' = z - \frac{A}{C} \qquad \text{..}(3)$$



On each scan line, we start by calculating the depth on a left edge of the polygon that intersects that scan line in the below fig. Depth values at each successive position across the scan line are then calculated by Eq. (3).

Scan lines intersecting a polygon surface we first determine the y-coordinate extents of each polygon, and process the surface from the topmost scan line to the bottom scan line. Starting at a top vertex, we can recursively calculate x positions down a left edge of the polygon as x' = x - l/m, where m is the slope of the edge.



Depth values down the edge are then obtained recursively as

$$z' = z + \frac{\frac{A}{m} + B}{C}$$

If we are processing down a vertical edge, the slope is infinite and the recursive calculations reduce to

$$z' = z + \frac{B}{C}$$

A- BUFFER METHOD

An extension of the ideas in the depth-buffer method is the A- buffer method. The A buffer method represents an antialiased, area- averaged, accumulation-buffer method developed by Lucas film for implementation in the surface-rendering system called REYES (an acronym for "Renders Everything You Ever Saw").

A drawback of the depth-buffer method is that it can only find one visible surface at each pixel position. 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 surface intensity can be taken into consideration at each pixel position, and object edges can be ant aliased. Each position in the A-buffer has two fields:

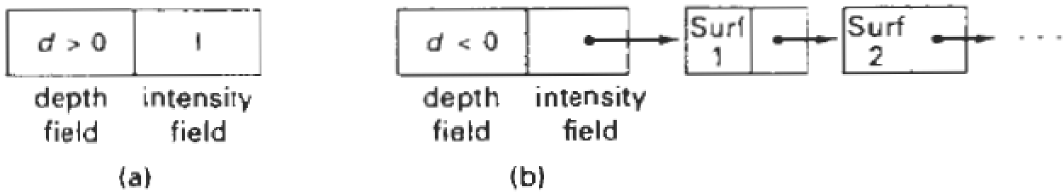
1)depth field - stores a positive or negative real number

2)intensity field - stores surface-intensity information or a pointer value.

If the depth field is positive, the number stored at that position is the depth of a single surface overlapping the corresponding pixel area. The intensity field then stores the RCB components of the surface color at that point and the percent of pixel coverage, as illustrated in Fig.A

If the depth field is negative, this indicates multiple-surface contributions to the pixel intensity. The intensity field then stores a pointer to a linked list of surface data, as in Fig. B.

Organization of an A-buffer pixel position



(A) single surface overlap of the corresponding pixel area
(B) multiple surface overlap

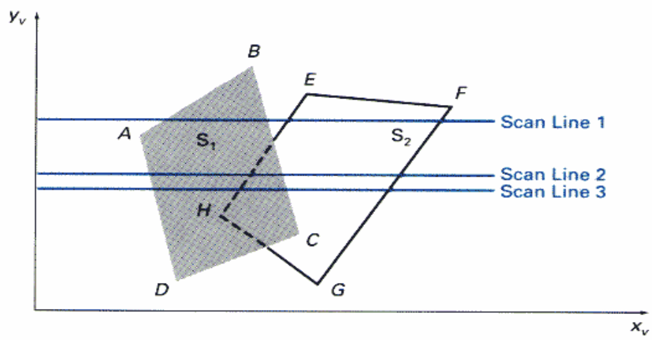
Data for each surface in the linked list includes

- RGB intensity components
- opacity parameter (percent of transparency)
- depth
- percent of area coverage
- other surface-rendering parameters
- pointer to next surface

.

Scan-Line Method

In this method, as each scan line is processed, all polygon surfaces intersecting that line are examined to determine which are visible. Across each scan line, depth calculations are made for each overlapping surface to determine which is nearest to the view plane. When the visible surface has been determined, the intensity value for that position is entered into the image buffer.



Scan lines crossing the projection of two surfaces, S_1 and S_2, in the view plane. Dashed lines indicate the boundaries of hidden surfaces.

The figure illustrates the scan-line method for locating visible portions of surfaces for pixel positions along the line. The active list for scan line 1 contains information from the edge table for edges AB, BC, *EH*, and *FG*. For positions along this scan line between edges AB and BC, only the flag for surface S_1 is on.

Therefore no depth calculations are necessary, and intensity information for surface S_1, is entered from the polygon table into the refresh buffer.

Similarly, between edges EH and FG, only the flag for surface S2 is on. No other positions along scan line 1 intersect surfaces, so the intensity values in the other areas are set to the background intensity.

For scan lines 2 and 3 , the active edge list contains edges AD, EH, BC, and FG. Along scan line 2 from edge AD to edge EH, only the flag for surface S_1, is on. But between edges EH and BC, the flags for both surfaces are on.

In this interval, depth calculations must be made using the plane coefficients for the two surfaces. For this example, the depth of surface S_1 is assumed to be less than that of S_2, so intensities for surface S_1, are loaded into the 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 number of overlapping polygon surfaces can be processed with this scan-line method. Flags for the surfaces are set to indicate whether a position is inside or outside, and depth calculations are performed when surfaces overlap.

List Priority Algorithms

- Determines a visibility ordering for objects ensures that a correct picture results if objects are rendered in that order. e.g. if no object overlaps in 'z' then we need only to sort objects by increasing 'z' and render them in that order.
- Farther objects are obscured by closer ones as pixels from the closer polygons overwrite those of more distant ones.
- If objects overlap in 'z', we may still be able to determine a correct order.
- If objects cyclically overlap, or penetrate each other, then there is no correct order.
- Hybrids that combine both object and image precision operations.
- Depth comparison and object splitting are done with object precision.
- Scan conversion(which relies on ability of graphics device to overwrite pixels of previously drawn objects) is done with image precision

Depth-Sort Method

Using both image-space and object-space operations, the depth- sorting method performs the following basic functions:

1. Sort all surfaces according to their distances from the view point.

2. Render the surfaces to the image buffer one at a time starting from the farthest surface.

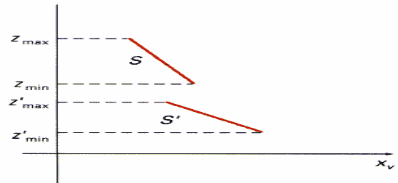
3. Surfaces close to the view point will replace those which are far away.

4. After all surfaces have been processed, the image buffer stores the final image.

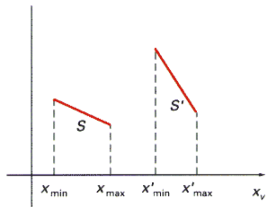
The basic idea of this method is simple. When there are only a few objects in the scene, this method can be very fast. However, as the number of objects increases, the sorting process can become very complex and time consuming.
Example: Assuming we are viewing along the z axis. Surface S with the greatest depth is then compared to other surfaces in the list to determine whether there are any overlaps in depth. If no depth overlaps occur, S can be scan converted. This process is repeated for the next surface in the list. However, if depth overlap is detected, we need to make some additional comparisons to determine whether any of the surfaces should be reordered

We make the following tests for each surface that overlaps with S. If any one of these tests is true, no reordering is necessary for that surface. The tests are listed in order of increasing difficulty.
1. The bounding rectangles in the xy plane for the two surfaces do not overlap
2. Surface S is completely behind the overlapping surface relative to the viewing position.
3. The overlapping surface is completely in front of S relative to the viewing position.
4. The projections of the two surfaces onto the view plane do not overlap.

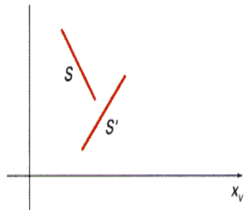
We perform these tests in the order listed and proceed to the next overlapping surface as soon as we find one of the tests is true. If all the overlapping surfaces pass at least one of these tests, none of them is behind S. No reordering is then necessary and S is scan converted.



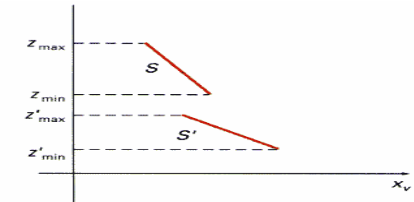
Two surfaces with no depth overlap.



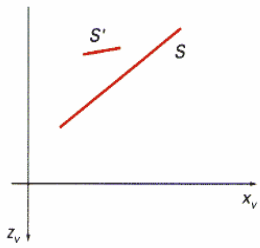
Two surfaces with depth overlap but no overlap in the *x* direction.



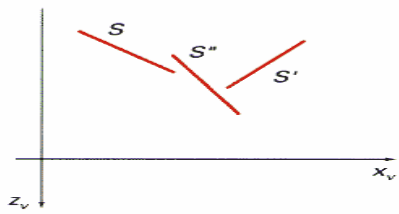
Surface *S* is completely behind ("inside") the overlapping surface *S*′.



Two surfaces with no depth overlap.



Surface *S* has greater depth but obscures surface *S*′.

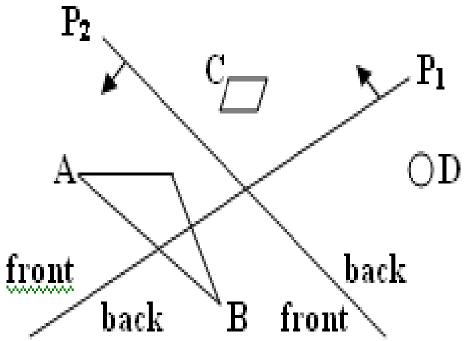


Three surfaces entered into the sorted surface list in the order *S*, *S*′, *S*″ should be reordered *S*′, *S*″, *S*.

Binary Space Partitioning

A binary space partitioning (BSP) tree is an efficient method for determining object visibility by painting surfaces onto the screen from back to front as in the painter's algorithm. The BSP tree is particularly useful when the view reference point changes, but object in a scene are at fixed position.

Applying a BSP tree to visibility testing involves identifying surfaces that are "inside" or "outside" the partitioning plane at each step of space subdivision relative to viewing direction. It is useful and efficient for calculating visibility among a static group of 3D polygons as seen from an arbitrary viewpoint. In the following figure,



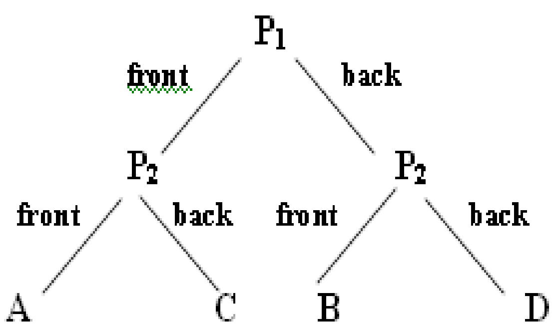


Fig: A region of space (a) partitioned with two planes P_1 and P_2 to form the BSP tree representation in (b).

- Here plane P_1 partitions the space into two sets of objects, one set of object is back and another set is in front of partitioning plane relative to viewing direction. Since one object is intersected by plane P_1, we divide that object into two separate objects labeled A and B. Now object A & C are in front of P_1, B and D are back of P_1.
- We next partition the space with plane P_2 and construct the binary free as fig (b). In this tree, the objects are represented as terminal nodes, with front object as left branches and behind object as right branches.
- When BSP tree is complete, we process the tree by selecting surface for displaying in order back to front. So foreground object are painted over back ground objects.

Octree Method

When an octree representation is used for viewing volume, hidden surface elimination is accomplished by projecting octree nodes into viewing surface in a front to back order. Following figure is the front face of a region space is formed with octants 0, 1, 2, 3. Surface in the front of these octants are visible to the viewer. The back octants 4, 5, 6, 7 are not visible. After octant sub-division and construction of octree, entire region is traversed by depth first traversal.

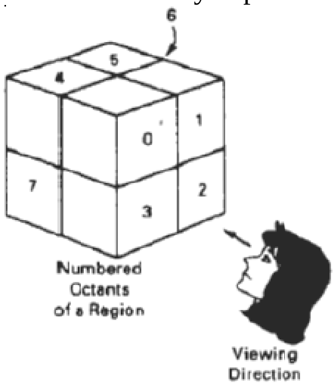


Fig1: Objects in octants 0, 1, 2, and 3 obscure objects in the back octants (4, 5, 6, 7) when the viewing direction is as shown.

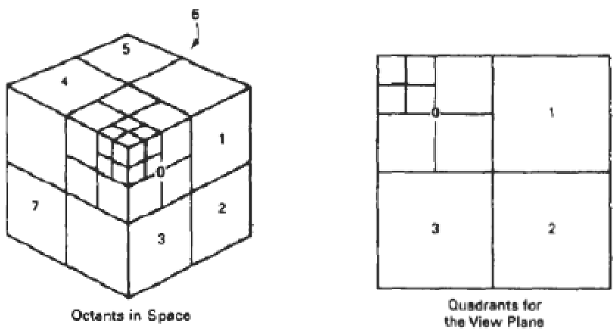


Fig2: Octant divisions for a region of space and the corresponding quadrant plane.

Different views of objects represented as octrees can be obtained by applying transformations to the octree representation that reorient the object according to the view selected.

Fig2 depicts the octants in a region of space and the corresponding quadrants on the view plane. Contributions to quadrant 0 come from octants 0 and 4. Color values in quadrant 1 are obtained from surfaces in octants 1 and 5, and values in each of the other two quadrants are generated from the pair of octants aligned with each of these quadrants.

Ray Casting (Ray Tracing)

Ray tracing also known as ray casting is efficient method for visibility detection in the objects. It can be used effectively with the object with curved surface. But it is also used for polygon surfaces.

Trace the path of an imaginary ray from the viewing position (eye) through viewing plane t object in the scene.

- Identify the visible surface by determining which surface is intersected first by the ray.
- Can be easily combined with lightning algorithms to generate shadow and reflection.
- It is good for curved surface but too slow for real time application.

Ray casting, as a visibility detection tool, is based on geometric optics methods, which trace the paths of light rays. Since there are an infinite number of light rays in a scene and we are interested only in those rays that pass through pixel positions, we can trace the light-ray paths backward from the pixels through the scene. The ray-casting approach is an effective visibility-detection method for scenes with curved surfaces, particularly spheres.

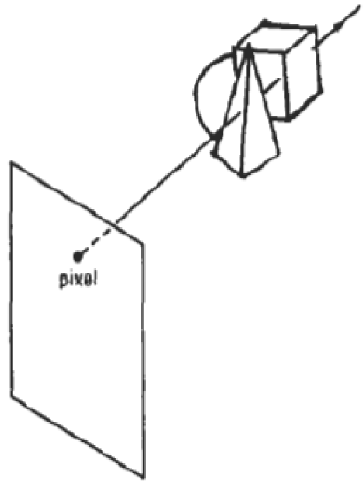
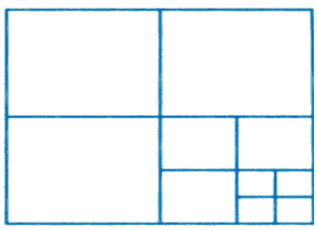


Fig: A ray along the line of sight from a pixel position through a scene.

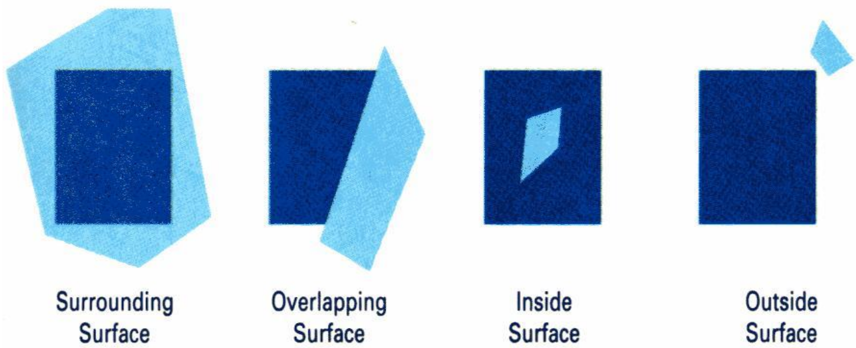
- In ray casting, we process pixels one at a time and calculate depths for all surfaces along the projection path to that pixel.
- In fact, Ray casting is a special case of *ray-tracing algorithms* that trace multiple ray paths to pick up global reflection and refraction contributions from multiple objects in a scene. With ray casting, we only follow a ray out from each pixel to the nearest object.

Area Subdivision Algorithms

The area-subdivision method takes advantage of area coherence in a scene by locating those view areas that represent part of a single surface. The total viewing area is successively divided into smaller and smaller rectangles until each small area is simple, ie. it is a single pixel, or is covered wholly by a part of a single visible surface or no surface at all.



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



Possible relationships between polygon surfaces and a rectangular area.

The procedure to determine whether we should subdivide an area into smaller rectangle is:
We first classify each of the surfaces, according to their relations with the area:

1. Surrounding surface - a single surface completely encloses the area
2. Overlapping surface– a single surface that is partly inside and partly outside the area
3. Inside surface - a single surface that is completely inside the area
4. Outside surface - a single surface that is completely outside the area.

To improve the speed of classification, we can make use of the bounding rectangles of surfaces for early confirmation or rejection that the surfaces should be belong to that type.
Check the result from 1., that, if any of the following condition is true, then, no subdivision of this area is needed.

a. All surfaces 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.

For cases b and c, the color of the area can be determined from that single surface.

Illumination and Surface Rendering:

- Realistic displays of a scene are obtained by perspective projections and applying natural lighting effects to the visible surfaces of object.
- An illumination model is also called lighting model and some times called as a shading model which is used to calculate the intensity of light that we should see at a given point on the surface of a object.
- A surface-rendering algorithm uses the intensity calculations from an illumination model.

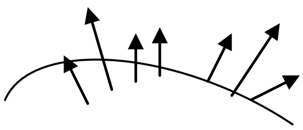
Light Sources:

- Sometimes light sources are referred as light emitting object and light reflectors. Generally light source is used to mean an object that is emitting radiant energy e.g. Sun.

Point Source: Point source is the simplest light emitter e.g. light bulb.

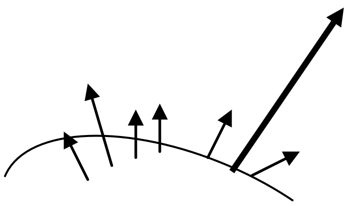
Distributed light source: Fluorescent light

- When light is incident on an opaque surface part of it is reflected and part of it is absorbed.
- Surface that are rough or grainy, tend to scatter the reflected light in all direction which is called diffuse reflection.



Diffuse reflection

- When light sources create highlights, or bright spots, called specular reflection



Specuular reflection

Illumination models:

Illumination models are used to calculate light intensities that we should see at a given point on the surface of an object. Lighting calculations are based on the optical properties of surfaces, the background lighting conditions and the light source specifications. All light sources are considered to be point sources, specified with a co-ordinate position and an intensity value (color). Some illumination models are:

1. **Ambient light**:

- This is a simplest illumination model. We can think of this model, which has no external light source-self-luminous objects. A surface that is not exposed directly to light source still will be visible if nearby objects are illuminated.
- The combination of light reflections form various surfaces to produce a uniform illumination is called <u>ambient light</u> or background light.
- Ambient light has no spatial or directional characteristics and amount on each object is a constant for all surfaces and all directions. In this model, illumination can be expressed by an illumination equation in variables associated with the point on the object being shaded. The equation expressing this simple model is

$$I = K_a$$

Where I is the resulting intensity and K_a is the object's intrinsic intensity.

If we assume that ambient light impinges equally on all surface from all direction, then

$$I = I_a K_a$$

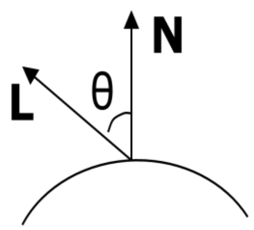
Where I_a is intensity of ambient light. The amount of light reflected from an object's surface is determined by K_a, the ambient-reflection coefficient.

K_a ranges from 0 to 1.

2. Diffuse reflection:

Objects illuminated by ambient light are uniformally illuminated across their surfaces even though light are more or less bright in direct proportion of ambient intensity. Illuminating object by a point light source, whose rays enumerate uniformly in all directions from a single point. The object's brightness varies form one part to another, depending on the direction of and distance to the light source.

- The fractional amount of the incident light that is diffusely reflected can be set for each surface with parameter K_d, the coefficient of diffuse-reflection.
- Value of K_d is in interval 0 to 1. If surface is highly reflected, K_d is set to near 1. The surface that absorbs almost incident light, K_d is set to nearly 0.
- Diffuse reflection intensity at any point on the surface if exposed only to ambient light is
  $$Iambdiff = I_d K_d$$
- Assuming diffuse reflections from the surface are scattered with equal intensity in all directions, independent of the viewing direction (surface called. "Ideal diffuse reflectors") also called Lambertian reflectors and governed by Lambert's cosine law.
  $$Idiff = K_d I_l \cos\theta$$



If N is unit vector normal to the surface & L is unit vector in the direction to the point slight source then

$$I_l, diff = K_d I_l (N.L)$$

In addition, many graphics packages introduce an ambient reflection coefficient K_a to modify the ambient-light intensity I_a

$$Idiff = K_a I_a + K_d I_l (N.L)$$

2. **Specular reflection and pong model**.

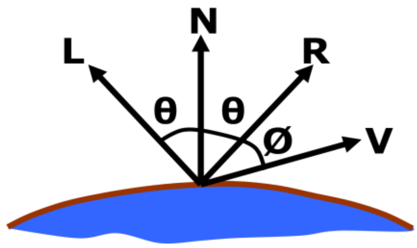
When we look at an illuminated shiny surface, such as polished metal, a person's forehead, we see a highlight or bright spot, at certain viewing direction. Such phenomenon is called specular reflection. It is the result of total or near total reflection of the incident light in a concentrated region around the " specular reflection angle = angle of incidence".

Specular reflection and pong model.

When we look at an illuminated shiny surface, such as polished metal, a person's forehead, we see a highlight or bright spot, at certain viewing direction. Such phenomenon is called specular reflection. It is the result of total or near total reflection of the incident light in a concentrated region around the " specular reflection angle = angle of incidence".

Specular reflection and pong model.

When we look at an illuminated shiny surface, such as polished metal, a person's forehead, we see a highlight or bright spot, at certain viewing direction. Such phenomenon is called specular reflection. It is the result of total or near total reflection of the incident light in a concentrated region around the " specular reflection angle = angle of incidence".



Let SR angle = angle of incidence as in figure.

L → unit vector pointing to light source

N → unit surface normal vector

R → unit vector in direction of specular reflection

V → unit vector pointing viewer

$\phi -$ the viewing angle relative to the specular reflection direction.

For ideal reflector (perfect mirror), incident light is reflected only in the specular reflection direction.

$\therefore V \;\&\; R$ concide $(\phi = 0)$

- Shiny surface have narrow ϕ and dull surface wider ϕ.
- An empirical model for calculating specular-reflection range developed by Phong Bui Tuong-called "Phong specular reflection model (or simply Phong model), sets the intensity of specular reflection proportional to $\cos^{ns}\phi \rightarrow 0 to 90^0$.
- Specular reflection parameter n_s is determined by type of surface
- Very shiny surface with large value n_s (say 100 or more) and dull surface , smaller n_s (down to 1)

Rough surface such as chalk , $n_s = 1$

- Intensity of specular reflection depends upon material properties of the surface and θ. Other factors such as the polarization and color of the incident light.
- For monochromatic specular intersity variations can approximated by SR coefficient $w(\theta)$ Fresnal's law of reflection describe specular reflection intensity with θ and using $w(\theta)$, Phong specular reflection model as

$$I_{spec} = w(\theta) I_l \cos^{n_s} \phi$$

Where I_l is intensity of light source. ϕ is viewing angle relative to SR direction R.

For a glass, we can replace $w(\theta)$ with constant K_s specular reflection coefficient.

So , $I_{spec} = K_s I_l \cos^{n_s} \phi$

] $= K_s I_l (V.R)^{n_s}$ Since $\cos\phi$ = V.R

Polygon (surface) Rendering Method

- Application of an illumination model to the rendering of standard graphics objects those formed with polygon surfaces are key technique for polygon rendering algorithm.
- Calculating the surface normal at each visible point and applying the desired illumination model at that point is expensive. We can describe more efficient shading models for surfaces defined by polygons and polygon meshes.
- Scan line algorithms typically apply a lighting model to obtain polygon surface rendering in one of two ways. Each polygon can be rendered with a single intensity, or the intensity can be obtained at each point of the surface using an interpolating scheme.

1. **Constant Intensity Shading**: (Flat Shading)

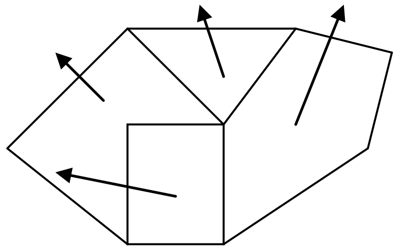
The simplest model for shading for a polygon is constant intensity shading also called as **Faceted Shading** Or flat shading. This approach implies an illumination model once to determine a single intensity value that is then used to render an entire polygon. Constant shading is useful for quickly displaying the general appearance of a curved surface.

This approach is valid if several assumptions are true:

1. The light source is at infinity, so **N.L** is constant across the polygon face.
2. The viewer is at infinity, so **N.V** is constant across the polygon face.
3. The polygon represents the actual surface being modeled and is not an approximation to a curved surface.

Even if all conditions are not true, we can still reasonably approximate surface-lighting effects using small polygon facets with fast shading and calculate the intensity for each facet, at the centre of the polygon of course constant shading does not produce the variations in shade a

cross the polygon that should occur.



2. **Interpolated Shading**:

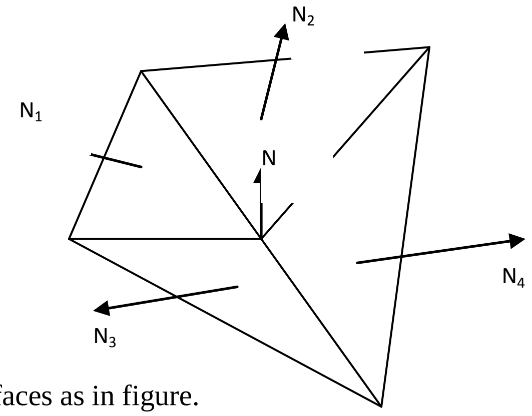
An alternative to evaluating the illumination equation at each point on the polygon, we can use the interpolated shading, in which shading information is linearly interpolated across a triangle from the values determined for its vertices. Gouraud generalized this technique for arbitrary polygons. This is particularly easy for a scan line algorithm that already interpolates the z- value across a span from interpolated z-values computed for the span's endpoints.

Gouraud Shading:

Gouraud shading , also called intensity interpolating shading or color interpolating shading, eliminates intensity discontinuities that occur in flat shading. Each polygon surface is rendered with Gouraud shading by performing following calculations.

1. Determine the average unit normal vector at each vertex. At each polygon vertex, we obtain a normal vertex by averaging the surface normals of all polygons sharing the vertex as:

$$N_v = \frac{\sum_{k=1}^{n} N_k}{|\sum_{k=1}^{n} N_k|}$$

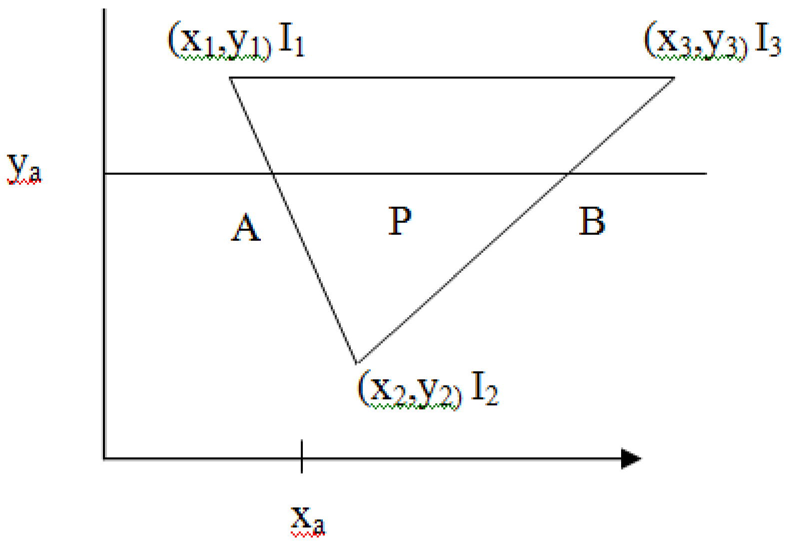


Here $N_v = \frac{N_1 + N_2 + N_3 + N_4}{|N_1 + N_2 + N_3 + N_4|}$

Where N_v is normal vector at a vertex sharing Four surfaces as in figure.

2. Apply illumination model to calculate each vertex intensity.
3. Linearly interpolate the vertex intensity over the surface of the polygon.

Once N_v is known, intensity at the vertices can obtain from lighting model.



-Here in figure, the intensity of vertices 1,2,3 are I_1 , I_2, I_3 . are obtained by averaging normals of each surface sharing the vertices and applying a illumination model.

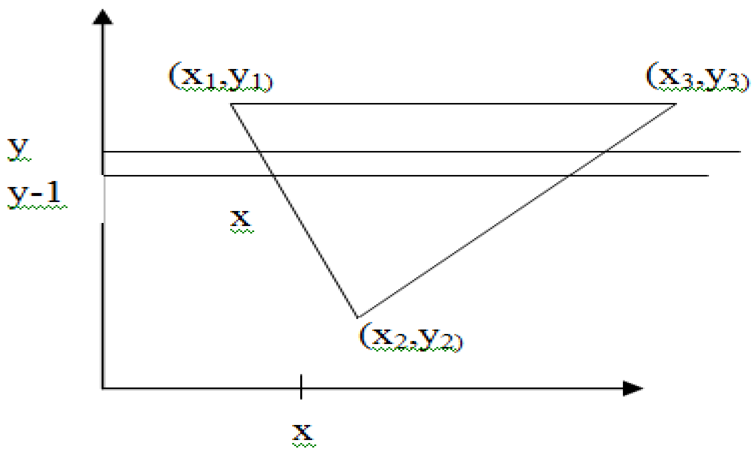
- For each scan line , intensity at intersection of line with Polygon edge are linearly interpolated from the intensities at the edge end point.So Intensity at intersection point A, I_a is obtained by linearly interpolating intensities of I_1 and I_2 as`

$$I_a = \frac{y_a - y_2}{y_1 - y_2} I_1 + \frac{y_1 - y_a}{y_1 - y_2} I_2$$

Similarly, the intensity at point B is obtained by linearly interpolating intensities at I_2 and I_3 as

$$I_b = \frac{y_a - y_2}{y_3 - y_2} I_3 + \frac{y_3 - y_a}{y_3 - y_2} I_2$$

The intensity of a point P in the polygon surface along scan-line is obtained by linearly interpolating intensities at I_a and I_b as,



$$I_p = \frac{x_p - x_a}{x_b - x_a} I_b + \frac{x_b - x_p}{x_b - x_a} I_a$$

Then incremental calculations are used to obtain Successive edge intensity values between scan-

lines as :

$$I = \frac{y - y_2}{y_1 - y_2} I_1 + \frac{y_1 - y}{y_1 - y_2} I_2$$

Then we can obtain the intensity along this edge for next scan line at y -1 position as

$$I' = \frac{y - 1 - y_2}{y_1 - y_2} I_1 + \frac{y_1 - (y - 1)}{y_1 - y_2} I_2$$

$$= I + \frac{I_2 - I_1}{y_1 - y_2}$$

Similar calculations are made to obtain intensity successive horizontal pixel.

Advantages: Removes intensity discontinuities at the edge as compared to constant shading.

Disadvantages: Highlights on the surface are sometimes displayed with anomalous shape and linear intensity interpolation can cause bright or dark intensity streak called mach-bands.

Phong Shading:

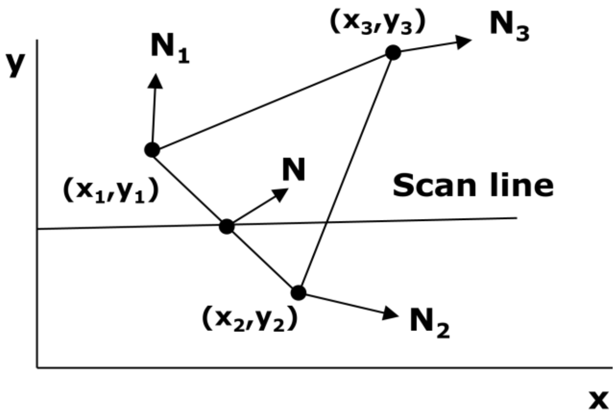
A more accurate method for rendering a polygon surface is to interpolate normal vector and then apply illumination model to each surface point. This method is called Phong shading or normal vector interpolation method for shading. It displays more realistic highlights and greatly reduce the mach band effect.

A polygon surface is rendered with Phong shading by carrying out following calculations.

- Determine the average normal unit vectors at each polygon vertex.

- Linearly interpolate vertex normals over the surface of polygon.

-Apply illumination model along each scan line to calculate the pixel intensities for the surface point.



In figure, N_1, N_2, N_3 are the normal unit vectors at each vertex of polygon surface. For scan-line that intersect an edge, the normal vector N can be obtained by vertically interpolating normal vectors of the vertex on that edge as.

$$N = \frac{y - y_2}{y_1 - y_2} N_1 + \frac{y_1 - y}{y_1 - y_2} N_2$$

Incremental calculations are used to evaluate normals between scanlines and along each in- dividual scan line as in Gouraud shading. Phong shading produces accurate results than the direct interpolation but it requires considerably more calculations.

Fast Phong Shading:

Fast Phong shading approximates the intensity calculations using a Taylor series expansion and Triangular surface patches. Since Phong shading interpolates normal vectors from vertex normals , we can express the surface normal N at any point (x,y) over a triangle as

N = Ax + By + C

Where A,B,C are determined from the three vertex equations.

$N_k = Ax_k + By_k + C$, $k = 1,2,3$ for (x_k, y_k) vertex.

Omitting the reflectivity and attenuation parameters

$$I_{diff}(x,y) = \frac{L.N}{|L|.|N|} = \frac{L.(Ax + By + C)}{|L|.|Ax + By + C|} = \frac{(L.A)x + (L.B)y + (L.C)}{|L|.|Ax + By + C|}$$

Re writing this

$$I_{diff}(x,y) = \frac{ax + by + c}{(dx^2 + exy + fy^2 + gx + hy + i)^{\frac{1}{2}}} \text{--------------------(1)}$$

Where a,b,c,d….. are used to represent the various dot products as

$$a = \frac{L.N}{|L|} \text{………..and so on}$$

Finally, denominator of equation (1) can express as Taylor series expansions and relations terms up to second degree in x,y. This yields,

$$I_{diff}(x,y) = T_5 x^2 + T_4 xy + T_3 y^2 + T_2 x + T_1 y + T_0$$

Where each T_k is a function of parameters a,b,c,d,……. And so forth.

This method still takes twice as long as in Gouraud shading. Normal Phong shading takes six to seven times that of Gouraud shading.

Chapter – 5

Computer Animation

Virtual Reality:

Virtual reality has been notoriously difficult to define over the years. Many people take "virtual" to mean fake or unreal, and "reality" to refer to the real world. This results in an oxymoron. The actual definition of virtual, however, is "to have the effect of being such without actually being such". The definition of "reality" is "the property of being real", and one of the definitions of "real" is "to have concrete existence". Using these definitions "virtual reality" means "to have the effect of concrete existence without actually having concrete existence", which is exactly the effect achieved in a good virtual reality system. There is no requirement that the virtual environment match the real world. Inspired by these considerations, for the virtual windtunnel we adapt the following definition:

Virtual reality is the use of computer technology to create the effect of an interactive three-dimensional world in which the objects have a sense of spatial presence.

In this definition, "spatial presence" means that the objects in the environment effectively have a location in three-dimensional space relative to and independent of your position. Note that this is an effect, not an illusion. The basic idea is to present the correct cues to your perceptual and cognitive system so that your brain interprets those cues as objects "out there" in the three-dimensional world. These cues have been surprisingly simple to provide using computer graphics: simply render a three-dimensional object (in stereo) from a point of view which matches the positions of your eyes as you move about. If the objects in the environment interact with you then the effect of spatial presence is greatly heightened. Note also that we do not require that the virtual reality experience be "immersive". While for some applications the sense of immersion is highly desirable, we do not feel that it is required for virtual reality. The main point of virtual reality, and the primary difference between conventional three-dimensional computer graphics and virtual reality is that in virtual reality you are working with.

Computer Animation

It will not be an exaggeration to say that animation can bring the dullest of the features to life. It has the magic of injecting energy and emotions into the most seemingly inanimate objects. Over the years the advancement of technology has made animation, a very attractive and much sought after component of multimedia.

What is Animation?

'To animate' literally means to give life to. Animating is moving something that cannot move on it's own. Animation adds to graphics the dimensions of time, which tremendously increase the potential of transmitting the desired information. In order to animate something the animator has to be able to specify directly or indirectly how the 'thing' has to move through time and space.

For detailed refer chapter _8 of textbook

Chapter- 1 (Remain)

Hardware Concepts

<u>Input Devices</u>

1. Tablet:

A tablet is digitizer. In general a digitizer is a device which is used to scan over an object, and to input a set of discrete coordinate positions. These positions can then be joined with straight-line segments to approximate the shape of the original object. A tablet digitizes an object detecting the position of a movable stylus (pencil-shaped device) or puck (link mouse with cross hairs for sighting positions) held in the user's hand. A tablet is flat surface, and its size of the tablet varies from about 6 by 6 inches up to 48 by 72 inches or more. The accuracy of the tablets usually falls below 0.2 mm. There are mainly three types of tablets.

a. Electrical tablet:

A grid of wires on ¼ to ½ inch centers is embedded in the tablet surface and electromagnetic signals generated by electrical pulses applied in sequence to the wires in the grid induce an electrical signal in a wire coil in the stylus (or puck). The strength of the signal induced by each pulse is used to determine the position of the stylus. The signal strength is also used to determine roughly how far the stylus is from the tablet. When the stylus is within ½ inch from the tablet, it is taken as "near" otherwise it is either "far" or "touching". When the stylus is "near" or "touching", a cursor is usually shown on the display to provide visual feedback to the user. A signal is sent to the computer when the tip of the stylus is pressed against the tablet, or when any button on the puck is pressed. The information provided by the tablet repeats 30 to 60 time per second.

b. Sonic tablet:

The sonic tablet uses sound waves to couple the stylus to microphones positioned on the periphery of the digitizing area. An electrical spark at the tip of the stylus creates sound bursts. The position of the stylus or the coordinate values is calculated using the delay between when the spark occurs and when its sound arrives at each microphone. the main advantage of sonic tablet is that it does not require a dedicated working area for the microphones can be placed on any surface to form the "tablet" work area. This facilitates digitizing drawing on thick books. Because in an electrical tablet this is not convenient for the stylus can not get closer to the tablet surface.

c. Resistive tablet:

The tablet is just a piece of glass coated with a thin layer of conducting material. When a buttery-powered stylus is activated at certain position, it emits high-frequency radio signals, which induces the radio signals on the conducting layer. The strength of he signal received at the edges of the tablet is used to calculate the position of the stylus.

Several types of tablets are transparent, and thus can be backlit for digitizing x-rays films and photographic negatives. The resistive tablet can be used to digitize the objects on CRT because it can be curved to the shape of the CRT. The mechanism used in the electrical or sonic tablets can also be used to digitize the 3D objects.

2. Touch panel

The touch panel allows the users to point at the screen directly with a finger to move the cursor around the screen, or to select the icons. Following are the mostly used touch panels.

a. Optical touch panel

It uses a series of infra-red light emitting diodes (LED) along one vertical edge and along one horizontal edge of the panel. The opposite vertical and horizontal edges contain photo-detectors to form a grid of invisible infrared light beams over the display area. Touching the screen breaks one or two vertical and horizontal light beams, thereby indicating the finger's position. The cursor is then moved to this position, or the icon at this position is selected. It two parallel beams are broken, the finger is presumed to be centered between them; if one is broken, the finger is presumed to be on the beam. There is a low-resolution panel, which offers 10 to 50 positions in each direction.

b. Sonic panel:

Bursts of high-frequency sound waves traveling alternately horizontally and vertically are generated atthe edge of the panel. Touching the screen causes part of each wave to be reflected back to its source. The screen position at the point of contact is then calculated using the time elapsed between when the wave is emitted and when it arrives back at the source. This is a high-resolution touch panel having about 500 positions in each direction.

c. Electrical touch panel:

It consists of slightly separated two transparent plates one coated with a thin layer of conducting material and the other with resistive material. When the panel is touched with a finger, the two plates are forced to touch at the point of contact thereby creating the touched position. The resolution of this touch panel is similar to that of sonic touch panel.

3. Light pen

It is a pencil-shaped device to determine the coordinates of a point on the screen where it is activated such as pressing the button. In raster display, Y is set at Y_{max} and X changes from 0 to X_{max} for the first scanning line. For second line, Y decreases by one and X again changes from 0 to $X_{max,}$ and so on. When the activated light pen "sees" a burst of light at certain position as the electron beam hits the phosphor coating at that position, it generates a electric pulse, which is used to save the video controller's X and Y registers and interrupt the computer. By reading the saved values, the graphics package can determine the coordinates of the position seen by the light pen. Because of the following drawbacks the light pens are not popular now a days.

- Light pen obscures the screen image as it is pointed to the required spot
- Prolong use of it can cause arm fatigue
- It can not report the coordinates of a point that is completely black. As a remedy one can display a dark blue field in place of the regular image for a single frame time
- It gives sometimes false reading due to background lighting in a room

4. Keyboard

A keyboard creates a code such as ASCII uniquely corresponding to a pressed key. It usually consists of alphanumeric keys, function keys, cursor-control keys, and separate numeric pad. It is used to move the cursor, to select he menu item, pre-defined functions. In computer graphics keyboard is mainly used for entering screen coordinates and text, to invoke certain functions. Now-a-days ergonomically designed keyboard (Ergonomic keyboard) with removable palm rests is available. The slope of each half of the keyboard can be adjusted separately.

5. Mouse

A mouse is a small hand-held device used to position the cursor on the screen. Mice are relative devices, that is, they can be picked up, moved in space, and then put down gain without any change in the reported position. For this, the computer maintains the current mouse position, which is incremented or decremented by the mouse movements. Following are the mice, which are mostly used in computer graphics.

a. Mechanical mouse

When a roller in the base of this mechanical mouse is moved, a pair of orthogonally arranged toothed wheels, each placed in between a LED and a photo detector, interrupts the light path. the number of interrupts so generated are used to report the mouse movements to the computer.

b. Optical mouse

The optical mouse is used on a special pad having a grid of alternating light and dark lines. A LED on the bottom of the mouse directs a beam of light down onto the pad, from which it is reflected and sensed by the detectors on the bottom of the mouse. As the mouse is moved, the reflected light beam is broken each time a dark line is crossed. The number of pulses so generated, which is equal to the number of lines crossed, are used to report mouse movements to the computer.

Displaydevices

The display devices used in graphics system is video monitor. The most common video monitor is based on CRT technology.

Cathode Ray Tube (CRT)

- CRT are the most common display devices on computer today. A CRT is an evacuated glass tube, with a heating element on one end and a phosphor-coated screen on the other end.
- When a current flows through this heating element (filament) the conductivity of metal is reduced due to high temperature. These cause electrons to pile up on the filament.
- These electrons are attracted to a strong positive charge from the outer surface of the focusing anode cylinder.
- Due to the weaker negative charge inside the cylinder, the electrons head towards the anode forced into a beam and accelerated by the inner cylinder walls in just the way that water is speeds up when its flow though a small diameter pipe.
- The forwarding fast electron beam is called Cathode Ray. A cathode ray tube is shown in figure below.

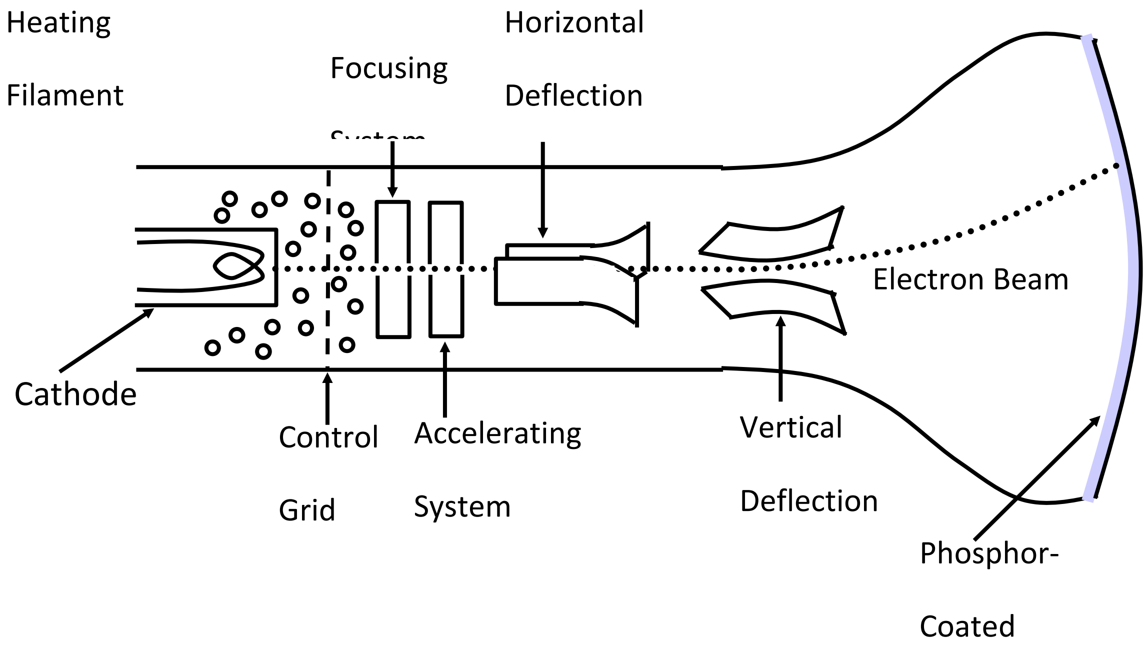


Figure :Cathode Ray Tube

- There are two sets of weakly charged deflection plates with oppositely charged, one positive and another negative. The first set displaces the beam up and down and the second displaces the beam left and right.
- The electrons are sent flying out of the neck of bottle (tube) until the smash into the phosphor coating on the other end.
- When electrons strike on phosphor coating, the phosphor then emits a small spot of light at each position contacted by electron beam. The glowing positions are used to represent the picture in the screen.
- The amount of light emitted by the phosphor coating depends on the no of electrons striking the screen. The brightness of the display is controlled by varying the voltage on the control grid.

<u>Persistence</u>:

- How long a phosphor continues to emit lightafter the electron beam is removed
- Persistence of phosphor is defined as **the time** it takesfor emitted light to decay to **1/10 (10%)** of its original intensity. Range of persistence of different phosphors can react many seconds.
- Phosphors for graphical display have persistence of 10 to 60 microseconds. Phosphors with low persistence are useful for animation whereas high persistence phosphor is useful for highly complex, static pictures.

<u>Refresh Rate</u>:

- Light emitted by phosphor fades very rapidly, so to keep the drawn picture glowing constantly, it is required to redraw the picture repeatedly and quickly directing the electron beam back over the some point. The no of times/sec the image is redrawn to give a feeling of non-flickering pictures is called refresh-rate.
- If Refresh rate decreases, flicker develops.
- For refresh displays, it depends on picture complexity
- Refresh rate above which flickering stops and steady it may be called as critical fusion frequency(CFF).

Resolution:

Maximum number of points displayed horizontally and vertically without overlap on a display screen is called resolution. In other ways , resolution is referred as the no of points per inch(dpi/pixel per inch)

Raster-Scan Display

- Raster Scan Display is based on television technology. In raster-scan the electron beam is swept across the screen, one row at a time from top to bottom. No of scan line per second is called horizontal scan rate.
- As electron beam moves across each row, the beam intensity is turned on and off to create a pattern of illuminated spots. Picture definition is stored in a memory called frame buffer or refresh buffer. Frame buffer holds all the intensity value for screen points.

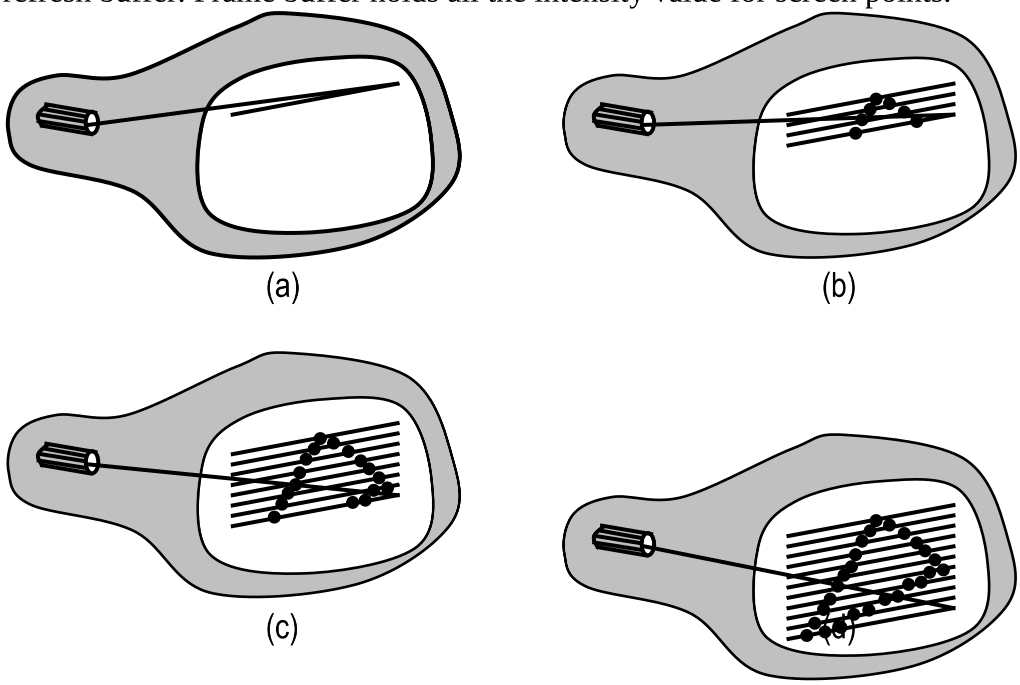


Figure: A raster-scan system displays an object as a set of points across each screen scan line

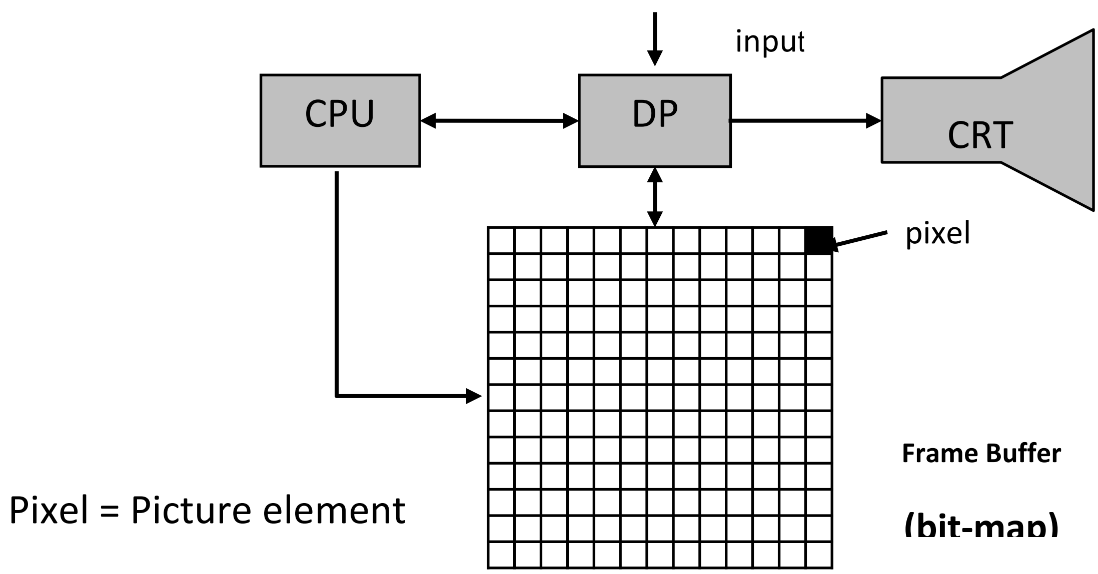


Figure: Raster Scan display system

- The stored intensity value is retrieved from frame buffer and painted on the scan line at a time. Home television are common examples using raster display
- Intensity range for pixel position depends on capability of raster system. For B/W system each point on screen are either on or off, so only one bit per pixel is needed to control the pixel intensity. To display color with varying intensity level, additional bits are needed. Up to 24 to 32 bit per pixel are included in high quality systems, which require more space of storage for the frame buffer, depending upon the resolution of the system.
- A system with 24 bit pixel and screen resolution 1024 × 1024 require 3 megabyte of storage in frame buffer.
  1024*1024 pixel = 1024*1024*24 bits = 3 MB

- The frame butter in B/W system stores a pixel with one bit per pixel so it is termed as bitmap. The frame buffer in multi bit per pixel storage, is called pixmap.
- Refreshing on Raster-Scan display is carried out at the rate of 60 or higher frames per second. 60 frames per second is also termed as 60 cycle per second usually used unit Hertz (HZ)
- Returning of electron beam from right end to deft end after refreshing each scan line is **<u>horizontal retrace</u>** . At the end of each frame, the electron beam returns to the top left corner to begin next frame called **<u>vertical retrace</u>**.

Interlaced: Display in two pass with interlacing.

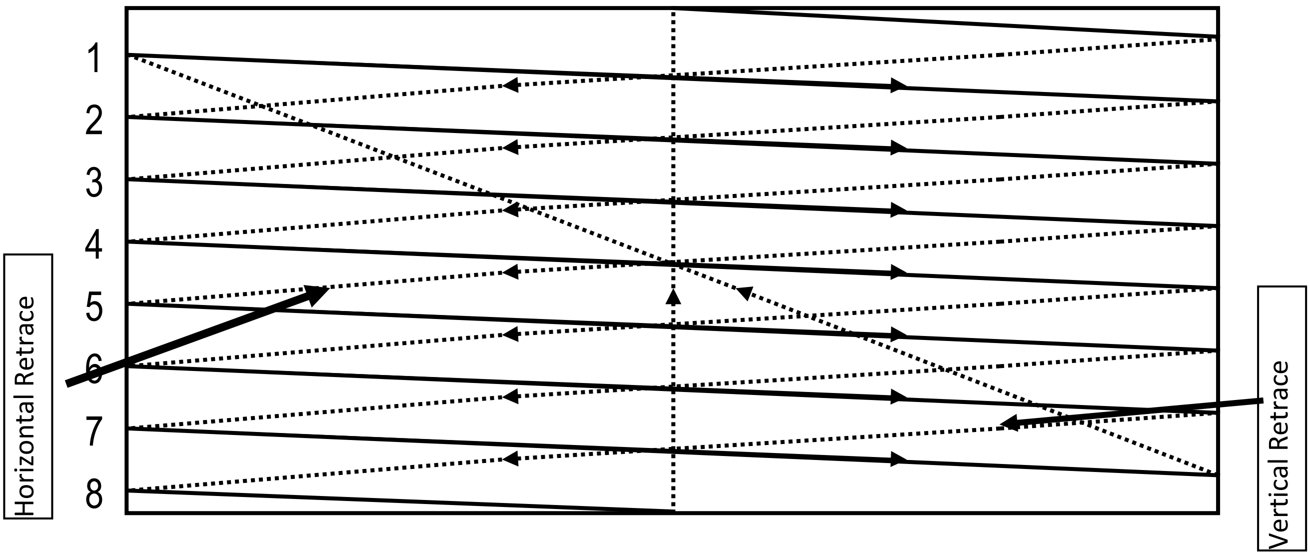


Figure: Horizontal retrace and Vertical retrace

***Question**: Consider a RGB raster system is to be designed using 8 inch by 10 inch screen with a resolution of 100 pixels per inch in each direction. If we want to store 6 bits per pixel in the frame buffer, How much storage(in bytes) do we need for the frame buffer?*

Solution: Size of screen = 8 inch × 10 inch.

Pixel per inch(Resolution) = 100.

Then, Total no of pixels = 8×100 × 10 × 100 pixels

Bit per pixel storage = 8

Therefore Total storage required in frame buffer = $(800 \times 1000 \times 8)$ bits

$= (800 \times 1000 \times 8)/8$ Bytes

= 800000 Bytes.

Frame Buffer Architecture of Raster Display

1. Indexed-color frame buffer.

In indexed –color frame buffer,

- Each pixel uses one byte in frame buffer.
- Each byte is an index into a color map.
- Each pixel may be one of 3^{24} colors, but only 256 color can be displayed at a time.
- There is a look-up table which has as many entries as there are pixel values.
- The table entry value is used to control the intensity or color of the CRT.

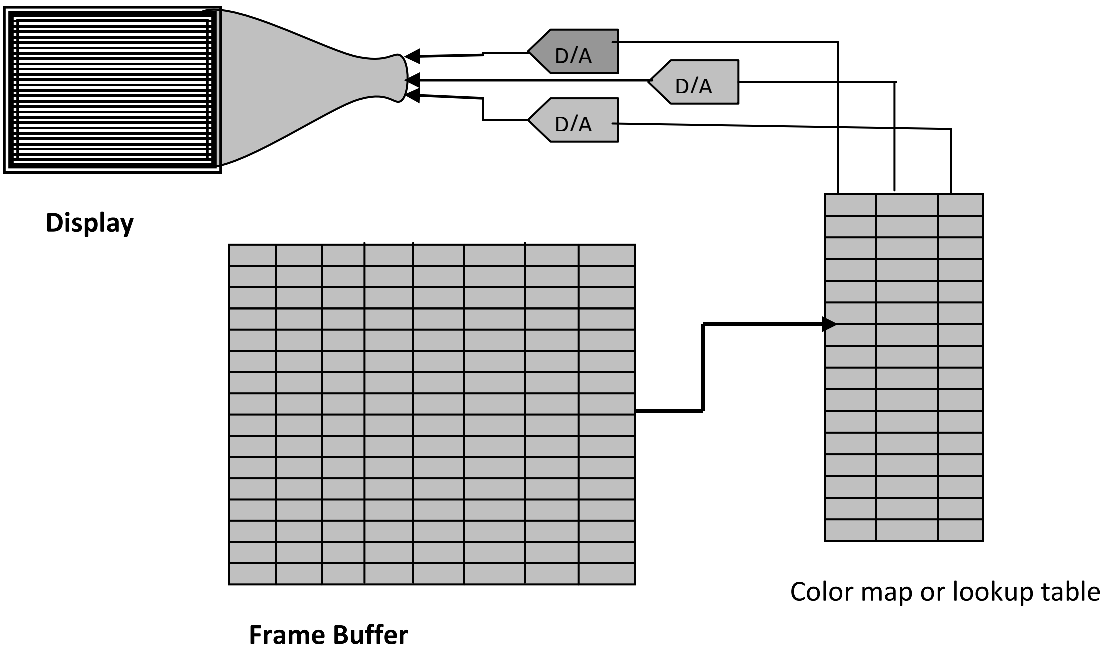


Figure: Indexed color frame buffer

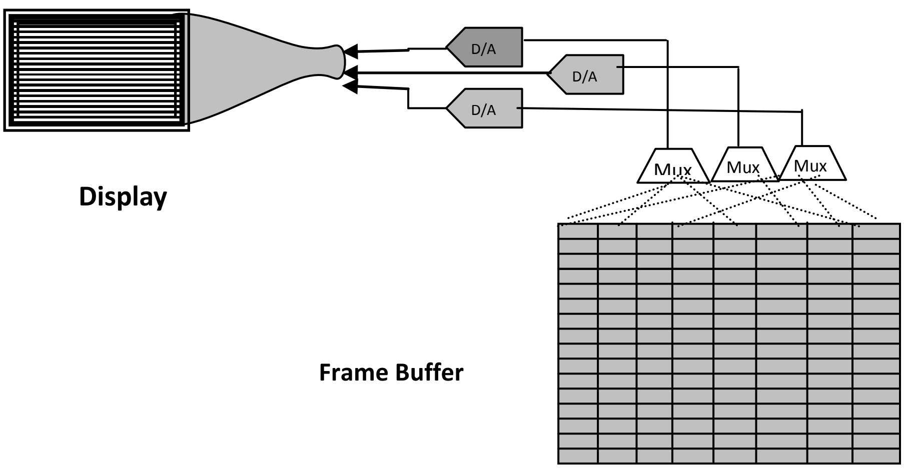


Figure : True Color Frame Buffer

2. **True-color frame buffer**: (24 bit or above): In true color frame buffer,

- Each pixel requires at least 3-bytes, one for each primary color (R,G,B)
- Sometimes combined with a look-up table per primary.
- Each pixel can be one of 2^{24} colors.

3 High-color frame buffer

- Popular PC/SVGA standard
- Pixels are packed in a short i.e. each primary color use 5 bit.
- Each pixel can be one of 2^{15} colors

	Red	Green	Blue

Random scan display: (Vector display)

In random scan system, the CRT has the electron beam that is directed only to the parts of the screen where the picture is to be drawn. It draws a picture one line at a time, so it is also called **vector display** (or stroke writing or calligraphic display). The component lines of a picture are drawn and refreshed by random scan system in any specified order.

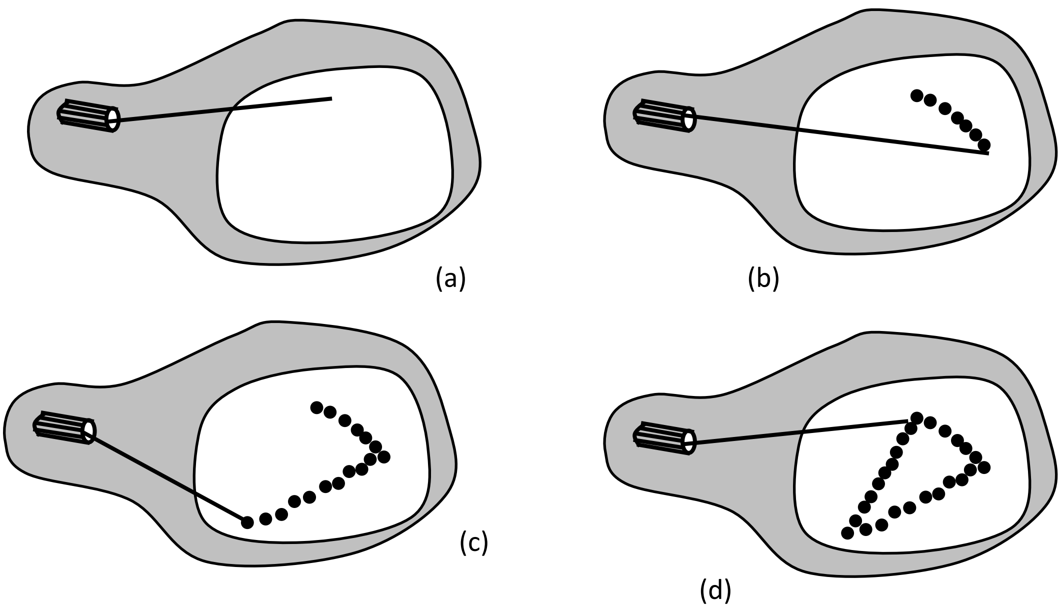


Figure: Random Scan Display

- The refresh rate of vector display depends upon the no of lines to be displayed for any image. Picture definition is stored as a set of line drawing instructions in an area of memory called the refresh display file (Display list or display file)
- To display a picture, the system cycles through the set of commands (line drawing) in the display file. After all commands have been processed, the system cycles back to the first line command in the list.
- Random scan systems are designed for drawing all component lines 30 to60 times per second. Such systems are designed for line-drawing applications and can not display realistic shaded scenes. Since CRT beam directly follows the line path, the vector display system produce smooth line.

Color CRT

In color CRT, the phosphor on the face of CRT screen are laid in to different fashion. Depending on the technology of CRT there are two methods for displaying the color pictures into the screen.

1. Beam penetration method 2. Shadow mask method

Beam Penetration method:

This method is commonly used for random scan display or vector display. In random scan display CRT, the two layers of phosphor usually red and green are coated on CRT screen. Display color depends upon how far electrons beam penetrate the phosphor layers.

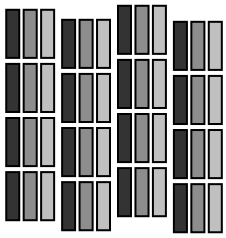
Slow electron excite only red layer so that we can see red color displayed on the screen pixel where the beam strikes. Fast electron beam excite green layer penetrating the red layer and we can see the green color displayed at the corresponding position. Intermediate is combination of red and green so two additional colors are possible – orange and yellow.

So only four colors are possible so no good quality picture in this type of display method.

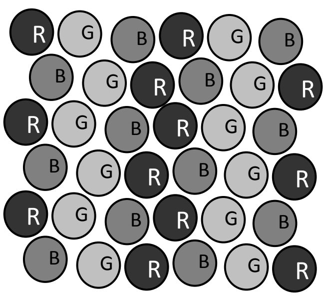
Shadow Mask Method:

Shadow mask method is used for raster scan system so they can produce wide range of colors. In shadow mask color CRT, the phosphor on the face of the screen are laid out in a precise geometric pattern. There are two primary variations.

1. The stripe pattern of inline tube
2. The delta pattern of delta tube



Stripe pattern



Delta Pattern

- In color CRT, the neck of tube, there are three electron guns, one for each red, green and blue colors. In phosphor coating there may be either strips one for each primary color, for a single pixel or there may be three dots one for each pixel in delta fashion.
- Special metal plate called a shadow mask is placed just behind the phosphor coating to cover front face.
- The mask is aligned so that it simultaneously allow each electron beam to see only the phosphor of its assigned color and block the phosphor of other two color.

Depending on the pattern of coating of phosphor, two types of raster scan color CRT are commonly used using shadow mask method.

1. **Delta-Delta CRT:**

- In delta-delta CRT, three electron beams one for each R,G,B colors are deflected and focused as a group onto shadow mask, which contains a series of holes aligned with the phosphor dots.
-

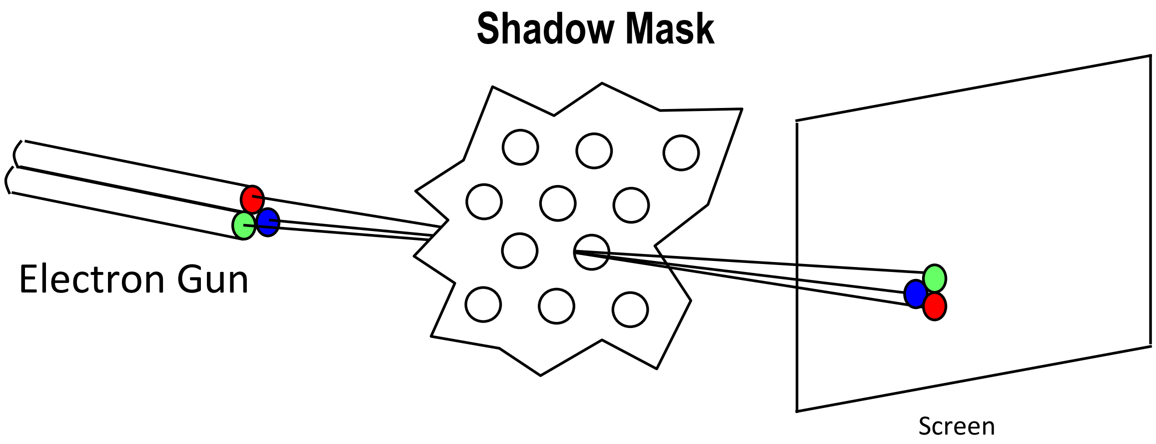


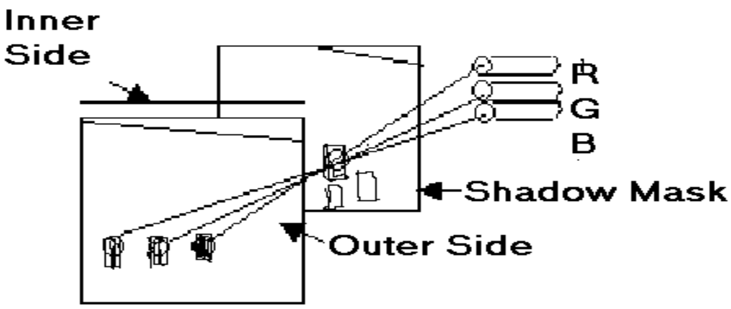
Figure: Shadow mask in Delta-Delta CRT

- Inner side of viewing has several groups of closely spaced red ,green and blue phosphor dot called triad in delta fashion.
- Thin metal plate adjusted with many holes near to inner surface called shadow mask which is mounted in such a way that each hole aligned with respective triad.
- Triad are so small that is perceived as a mixture of colors. When three beams pass through a hole in shadow mask, they activate the dot triangle to illuminate an small spot colored on the screen.
- The color variation in shadow mask CRT can be obtained by varying the intensity level of the three electron guns.

The main draw back of this CRT is due to difficulty for the alignment of shadow mask hole and respective triads.

A precision inline CRT:

This CRT uses strips pattern instead of delta pattern. Three strips one for each R, G, B color are used for a single pixel along a scan line so called inline. This eliminates the drawbacks of delta-delta CRT at the cost of slight reduction of image sharpness at the edge of the tube.



− Normally 1000 scan lines are displayed in this method. Three beams simultaneously expose three inline phosphor dots along scan line.

Architecture of Raster Scan System:

The raster graphics systems typically consists of several processing units. CPU is the main processing unit of computer systems. Besides CPU, graphics system consists of a special purpose processor called video controller or display processor. The display processor controls the operation of the display device.

The organization of raster system is as shown below

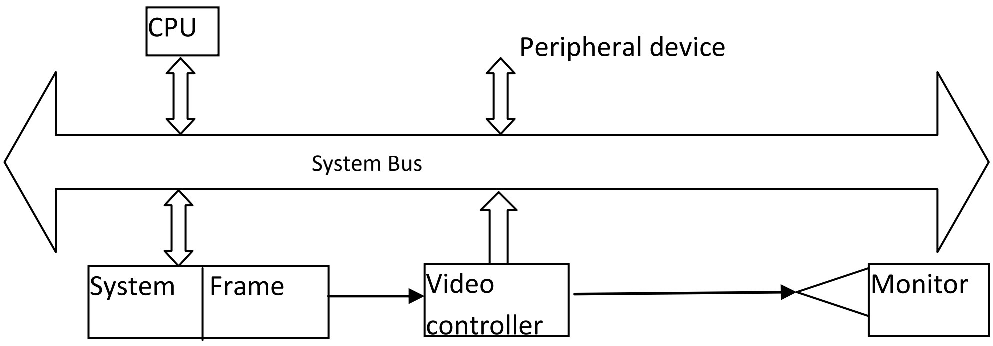


Figure: A simple Raster System.

- A fixed area of system memory is reserved for the frame buffer. The video controller has the direct access to the frame buffer for refreshing the screen.
- The video controller cycles through the frame buffer, one scan line at a time, typically at 60 times per second or higher. The contents of frame buffer are used to control the CRT beam's intensity or color.

The video controller:

The video controller is organized as in figure below. The raster-scan generator produces deflection signals that generate the raster scan and also controls the X and Y address registers, which in turn defines memory location to be accessed next. Assume that the frame buffer is addressed in X from 0 to X_{max} and in Y from 0 to Y_{max} then, at the start of each refresh cycle, X address register is set to 0 and Y register is set to 0 (top scan line).

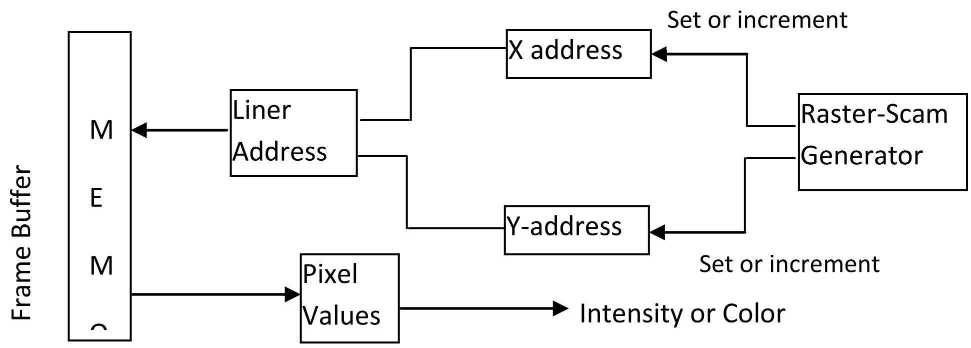


Figure: Organization of Video Controller.

As first scan line is generated, the X address is incremented up to X_{max}. Each pixel value is fetched and used to control the intensity of CRT beam. After first scan line X address is reset to 0 and Y address is incremented by 1. The process is continued until the last scan line ($Y=Y_{max}$) is generated.

Raster-Scan Display Processor:

The raster scan with a peripheral display processor is a common architecture that avoids the disadvantage of simple raster scan system. It includes a separate graphics processor to perform graphics functions such as scan conversion and raster operation and a separate frame buffer for image refresh.

The display processor has its own separate memory called display processor memory.

- System memory holds data and those programs that execute on the CPU, and the application program, graphics packages and OS.
- The display processor memory holds data plus the program that perform scan conversion and raster operations.
- The frame buffer stores displayable image created by scan conversion and raster operations. The organization is given below in figure

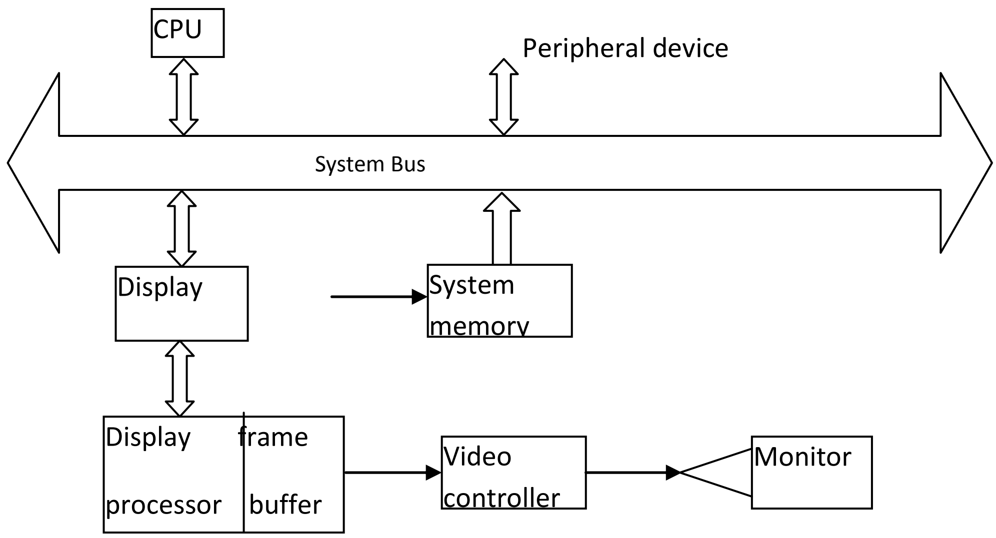


Figure: Architecture Raster scan system with display processor

2. **Vector Display System.**

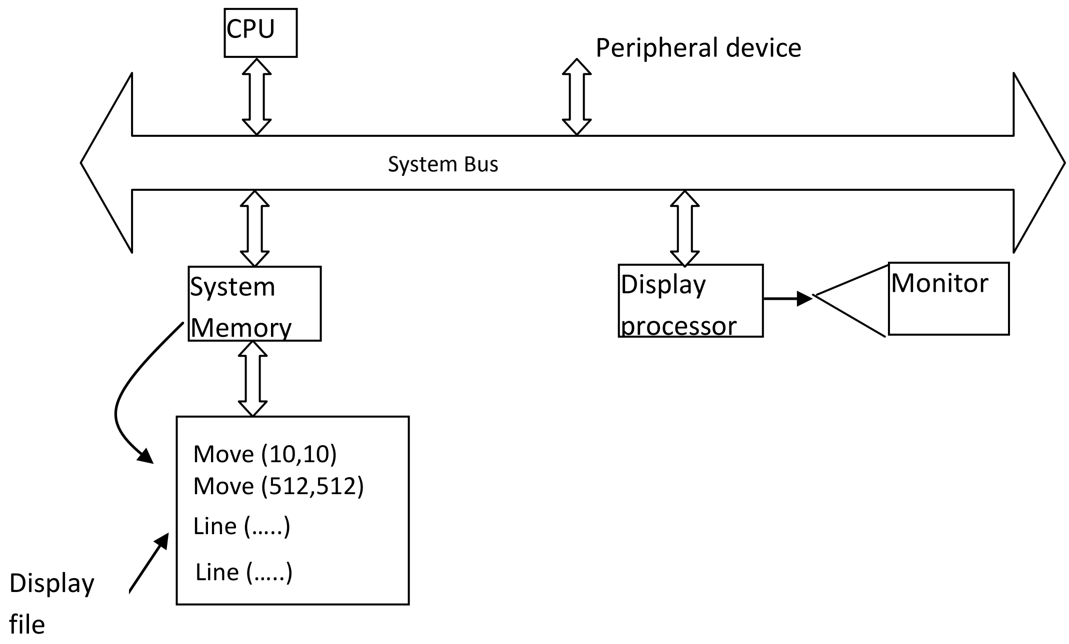


Figure : Architecture of Vector Display System

- Vector display system consists of several units along with peripheral devices. The display processor is also called as graphics controller.
- Graphics package creates a display list and stores in systems memory (consists of points and line drawing commands) called display list or display file.
- Refresh time around so cycle per second.
- Vector display technology is used in monochromatic or beam penetration color CRT.
- Graphics are drawn on a vector display system by directing the electron beam along component line.

Advantages:

- Can produce output with high resolutions.
- Better for animation than raster system since only end point information is needed.

Disadvantages:

- Cannot fill area with pattern and manipulate bits.
- Refreshing image depends upon its complexity.