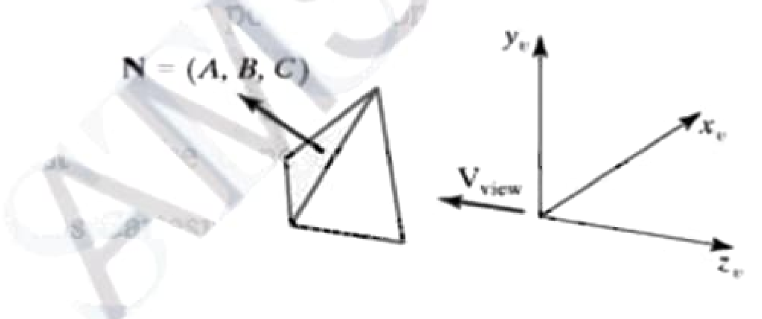
Back Face detection:

- A fast and simple object-space method for identifying the back faces of a polyhedron is based on the "inside-outside" tests.
- A point (x, y, z) is "inside" a polygon surface with plane parameters A, B, C, and D if When an inside point is along the line of sight to the surface, the polygon must be a back face
- This test can be done 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 (or "camera") position, then this polygon is a back face if

  V.N > 0
- Furthermore, if object descriptions are converted to projection coordinates and your viewing direction is parallel to the viewing z-axis, then

  $V = (0, 0, V_z)$ and $V.N = V_Z C$
- In a right-handed viewing system with viewing direction along the negative Z_V axis, the polygon is a back face if $C < 0$.

  Also, any face cannot be seen whose normal has z component $C = 0$, since your viewing



A polygon surface with plane parameter $C < 0$ in a right-handed viewing coordinate system is identified as a back face when the viewing direction is along the negative z_v axis.

direction is towards that polygon.

Depth Buffer Method:

- It is an image-space approach. The basic idea is to test the Z-depth of each surface to determine the closest (visible) surface.
- In this method each surface is processed separately one pixel position at a time across the surface. The depth values for a pixel are compared and the closest (smallest y) surface determines the color to be displayed in the frame buffer.
- It is applied very efficiently on surfaces of polygon. Surfaces can be processed in any order. To override the closer polygons from the far ones, two buffers named frame buffer and depth buffer are used.
- **Depth buffer** is used to store depth values for (x, y) position, as surfaces are processed ($0 \leq \text{depth} \leq 1$).
- The **frame buffer** is used to store the intensity value of color value at each position (x, y).
- The z-coordinates are usually normalized to the range [0, 1]. The 0 value for z-coordinate indicates back clipping pane and 1 value for z-coordinates indicates front clipping pane.

Algorithm

1. Initialize the depth and refresh buffer

   Depthbuffer (x, y) = 0

   Framebuffer (x, y) = background color

2. For each position on each polygon surface, compare depth values to previously stored values in the depth buffer to determine visibility.

   For each projected (x, y) pixel position of a polygon, calculate depth z. If Z > depthbuffer (x, y)

   set depthbuffer (x, y) = z,

   framebuffer (x, y) = surfacecolor (x, y)

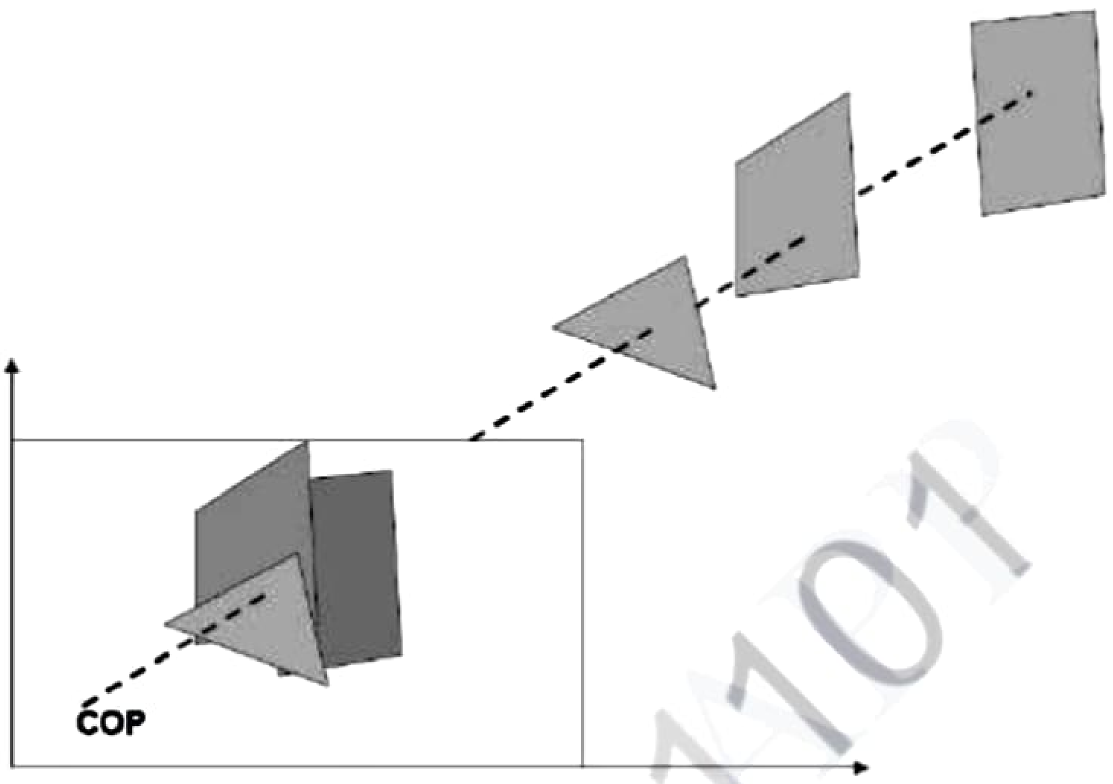


Figure 3.7 A view plane position

A buffer Method:

- The A-buffer method is an extension of the depth-buffer method.
- The A-buffer method is a visibility detection method developed at Lucas film Studios for the rendering system Renders Everything You Ever Saw (REYES).
- The A-buffer expands on the depth buffer method to allow transparencies. The key data structure in the A-buffer is the accumulation buffer.
- Each position in the A-buffer has two fields −
- **Depth field** − It stores a positive or negative real number
- **Intensity field** − It stores surface-intensity information or a pointer value

If depth >= 0, the number stored at that position is the depth of a single surface overlapping the corresponding pixel area. The intensity field then stores the RGB components of the surface color at that point and the percent of pixel coverage.

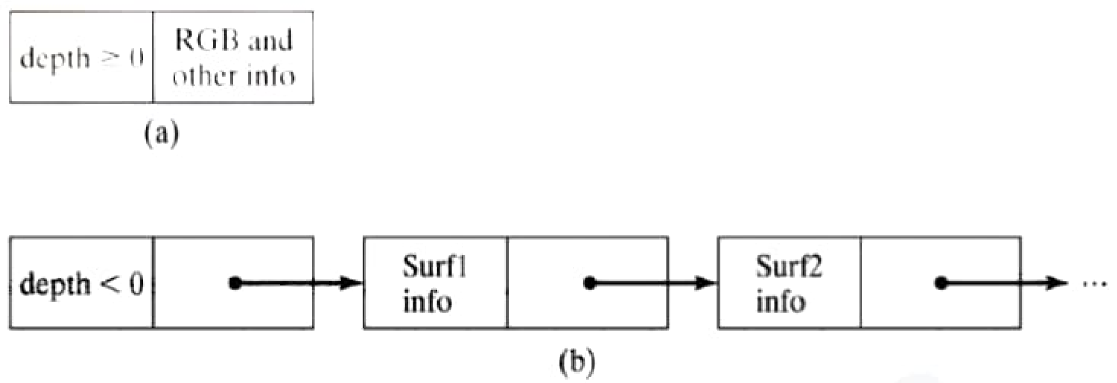


Fig.3.8 : Organization of an A buffer pixel position (a) Single-surface overlap (b) Multiple-surface overlap

- If depth < 0, it indicates multiple-surface contributions to the pixel intensity. The intensity field then stores a pointer to a linked list of surface data. The surface buffer in the A-buffer includes
  - RGB intensity components
  - Opacity Parameter
  - Depth
  - Percent of area coverage
  - Surface identifier

Scan line Method:

- It is an image-space method to identify visible surface.
- This method has depth information for only single scan-line. In order to require one scan-line of depth values, it is necessary to group and process all polygons intersecting a given scan-line at the same time before processing the next scan-line.
- Two important tables -edge table and polygon table, are maintained for this.
- **The Edge Table** − It contains coordinate endpoints of each line in the scene, the inverse slope of each line, and pointers into the polygon table to connect edges to surfaces.

- **The Polygon Table** – It contains the plane coefficients, surface material properties, other surface data, and may be pointers to the edge table.

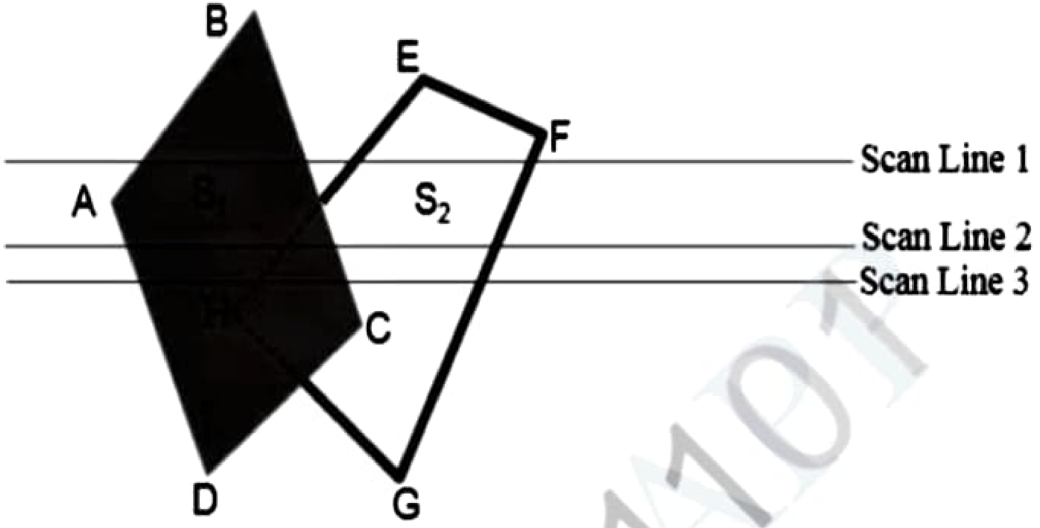


Fig.3.9 scan line intersecting a polygon surface

- To facilitate the search for surfaces crossing a given scan-line, an active list of edges is formed. The active list stores only those edges that cross the scan-line in order of increasing x.
- Also a flag is set for each surface to indicate whether a position along a scan-line is either inside or outside the surface.