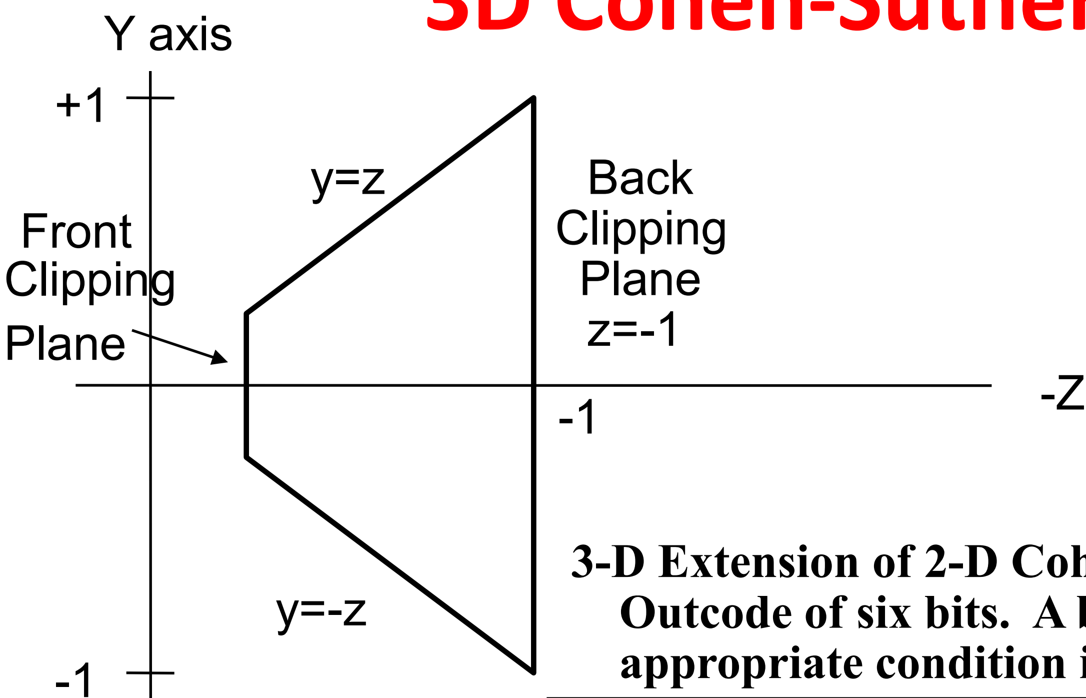
3D Clipping
&
Viewing process

3D Cohen-Sutherland Algorithm



3-D Extension of 2-D Cohen-Sutherland Algorithm, Outcode of six bits. A bit is true (1) when the appropriate condition is satisfied

Bit 1 - Point is above view volume	**$y > -z$**
Bit 2 - Point is below view volume	**$y < z$**
Bit 3 - Point is right of view volume	**$x > -z$**
Bit 4 - Point is left of view volume	**$x < z$**
Bit 5 - Point is behind view volume	**$z < -1$**
Bit 6 - Point is in front of view volume	**$z > zmin$**

3D Cohen-Sutherland Algorithm

- A line is trivially accepted if both endpoints have a code of all zeros.
- A line is trivially rejected if the bit-by-bit logical AND of the codes is not all zeros.
- Otherwise Calculate intersections.

On the y = z plane from parametric equation of the line:

$$y_0 + t(y_1 - y_0) = z_0 + t(z_1 - z_0)$$

Solve for t and calculate x and y. Already know z = y

3D Cohen-Sutherland Algorithm

- If t not between [0, 1], then the intersection in not between P0 and P1
  - $t = (z0-y0)/[(y1 - y0)-(z1-z0)]$
- If t between [0, 1], then use t to find x and y
  - $x = x0 + [(x1 - x0)(z0-y0)]/[(y1-y0)-(z1-z0)]$
  - $y = y0 + [(y1 - y0)(z0-y0)]/[(y1-y0)-(z1-z0)]$

3D viewing process

- Specify a 3D view volume
  - **Nper = [Sper][SHper][T(-PRP)][R][T(-VRP)]**
- Clip against view volume
- Project onto a 2D viewing plane
  - **Mper : d = -PRPn**
- Apply 2D viewing transformations to map window contents into 2D-image viewport
  - V = $[T^{-1}][S][T]$

3D viewing Transformation 3D to 2D

3D to 2D

P'(x',y',z',w) = [**M**per][**N**per][**P(**x,y,z,1**)**]

2D to Image

P''(x,y,1) = [**V**][**P(**x'/w,y'/w,1**)**]