UNIVERSITY OF ASIA PACIFIC

Department of Computer Science & Engineering

Assignment

Course Code : CSE 425

Course Title : Computer Graphics

Submitted by:

Name : Sheikh Nafez Sadnan

Reg. No. : 20101106

Roll No. : 106

Section : B

Submitted To:

Shammi Akhtar

Assistant Professor

Department of CSE

University of Asia Pacific

Composite Transformation

4.2

a) From 4.1,

$$R_{30^\circ} = \begin{pmatrix} \cos 30^\circ & -\sin 30^\circ \\ \sin 30^\circ & \cos 30^\circ \end{pmatrix} = \begin{bmatrix} \sqrt{3}/2 & -1/2 \\ 1/2 & \sqrt{3}/2 \end{bmatrix}$$

b) New coordinates of the point $P(2,-4)$

$$\begin{bmatrix} \sqrt{3}/2 & -1/2 \\ 1/2 & \sqrt{3}/2 \end{bmatrix} \begin{bmatrix} 2 \\ -4 \end{bmatrix} = \begin{bmatrix} \sqrt{3}+2 \\ 1-2\sqrt{3} \end{bmatrix}$$

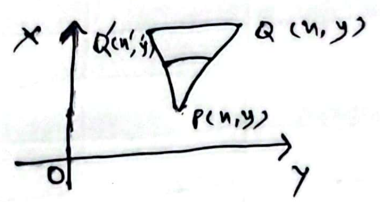
4.3 Describe transformation that rotate object

(i) Translate center point P to the origin.

(ii) Perform θ degree rotation about the origin.

(iii) Translate P back to origin.

$$R_{\theta P} = T_v \cdot R_\theta \, T_{-v}$$



4.4 General form of matrix rotation:

$$R_{\theta P} = \begin{bmatrix} 1 & 0 & h \\ 0 & 1 & k \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -h \\ 0 & 1 & -k \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \cos\theta & -\sin\theta & [-h\cos\theta + k\sin\theta + h] \\ \sin\theta & \cos\theta & [-h\sin\theta + (-k\cos\theta) + k] \\ 0 & 0 & 1 \end{bmatrix}$$

4.5 (a) 45° rotation of triangle A (0,0), B (1,1), C (5,2)

$$ABC = \begin{bmatrix} 0 & 1 & 5 \\ 0 & 1 & 2 \\ 1 & 1 & 1 \end{bmatrix} \qquad R_{45°} = \begin{bmatrix} \cos 45° & -\sin 45° & 0 \\ \sin 45° & \cos 45° & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \frac{\sqrt{2}}{2} & -\sqrt{2}/2 & 0 \\ \sqrt{2}/2 & \sqrt{2}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$A'B'C' = R_{45°}\,[ABC] = \begin{bmatrix} \sqrt{2}/2 & -\sqrt{2}/2 & 0 \\ \sqrt{2}/2 & \sqrt{2}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 5 \\ 0 & 1 & 2 \\ 1 & 1 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 & 3\sqrt{2}/2 \\ 0 & \sqrt{2} & 7\sqrt{2}/2 \\ 1 & 1 & 1 \end{bmatrix}$$

Now, $A' = (0,0)$, $B' = (0, \sqrt{2})$, $C' = (\frac{3}{2}\sqrt{2}, \frac{7}{2}\sqrt{2})$

(b) From prob 4.4, the rotation matrix is given by $R_{45°}$

$P = T_v \cdot R_{45°} \cdot T_{-v}$ where $v = -I - J$ so,

$$R_{45°,P} = \begin{pmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & 0 \\ \sqrt{2}/2 & \sqrt{2}/2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} \sqrt{2}/2 & -\sqrt{2}/2 & -1 \\ \sqrt{2}/2 & \sqrt{2}/2 & (\sqrt{2}-1) \\ 0 & 0 & 1 \end{pmatrix}$$

$$\text{Now, } [A'B'C'] = R_{45°,P} \cdot [ABC] = \begin{pmatrix} \sqrt{2}/2 & -\sqrt{2}/2 & -1 \\ \sqrt{2}/2 & \sqrt{2}/2 & (\sqrt{2}-1) \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 & 5 \\ 0 & 1 & 2 \\ 1 & 2 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} -1 & -1 & (\frac{3}{2}\sqrt{2}-1) \\ (\sqrt{2}-1) & (2\sqrt{2}-1) & (\frac{7}{2}\sqrt{2}-1) \\ 1 & 1 & 1 \end{pmatrix}$$

So $A' = (-1, \sqrt{2}-1)$, $B' = (-1, 2\sqrt{2}-1)$, $C' = (\frac{3}{2}\sqrt{2}-1, \frac{7}{2}\sqrt{2}-1)$

4.6

(a) Transformation (origin) a units in x direction P(x, y) point

(ax, y)

Now, $S_{a,1,P}$ $\begin{bmatrix} a & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} ax \\ y \end{bmatrix}$

(b) b units in Y direction : $S_{1,b,P}$ so,

$$\begin{bmatrix} 1 & 0 \\ 0 & b \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x \\ by \end{bmatrix}$$

(c) a units in x and b units in Y direction, $S_{a,b,P}$

$$\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} ax \\ by \end{bmatrix}$$

4.7

Scaling matrix to fixed point P(h, k)

$$S_{ab.P} = T_v . S_{ab} . T_{-v}$$

$$= \begin{bmatrix} 1 & 0 & h \\ 0 & 1 & k \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & -h \\ 0 & 1 & -k \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} a & 0 & -ah+h \\ 0 & b & -bk+k \\ 0 & 0 & 1 \end{bmatrix}$$

4.8

$$S_{2,2,C} = T_v . S_{2,2} . T_{-v}$$

$$= \begin{bmatrix} 1 & 0 & 5 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & -5 \\ 0 & 1 & -2 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 2 & 0 & -5 \\ 0 & 2 & -2 \\ 0 & 0 & 1 \end{bmatrix}$$

$$[ABC] = \begin{bmatrix} 0 & 1 & 5 \\ 0 & 1 & 2 \\ 1 & 1 & 1 \end{bmatrix}$$

$$[A'B'C'] = S_{22.C} . [ABC] = \begin{bmatrix} 2 & 0 & -5 \\ 0 & 2 & -2 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 0 & 1 & 5 \\ 0 & 1 & 2 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} -5 & -3 & 5 \\ -2 & 0 & 2 \\ 1 & 1 & 1 \end{bmatrix}$$

Schaum's outline

6.1

(a) We can find the required transformation T by composing two rotation matrics:

$T = R_{\theta_y J} \cdot R_{\theta_x I}$

$$= \begin{pmatrix} \cos\theta_y & 0 & \sin\theta_y & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta_y & 0 & \cos\theta_y & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta_x & -\sin\theta_x & 0 \\ 0 & \sin\theta_x & \cos\theta_x & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} \cos\theta_y & \sin\theta_y \sin\theta_x & \sin\theta_y \cos\theta_x & 0 \\ 0 & \cos\theta_x & -\sin\theta_x & 0 \\ -\sin\theta_x & \cos\theta_y \sin\theta_x & \cos\theta_y \cos\theta_x & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

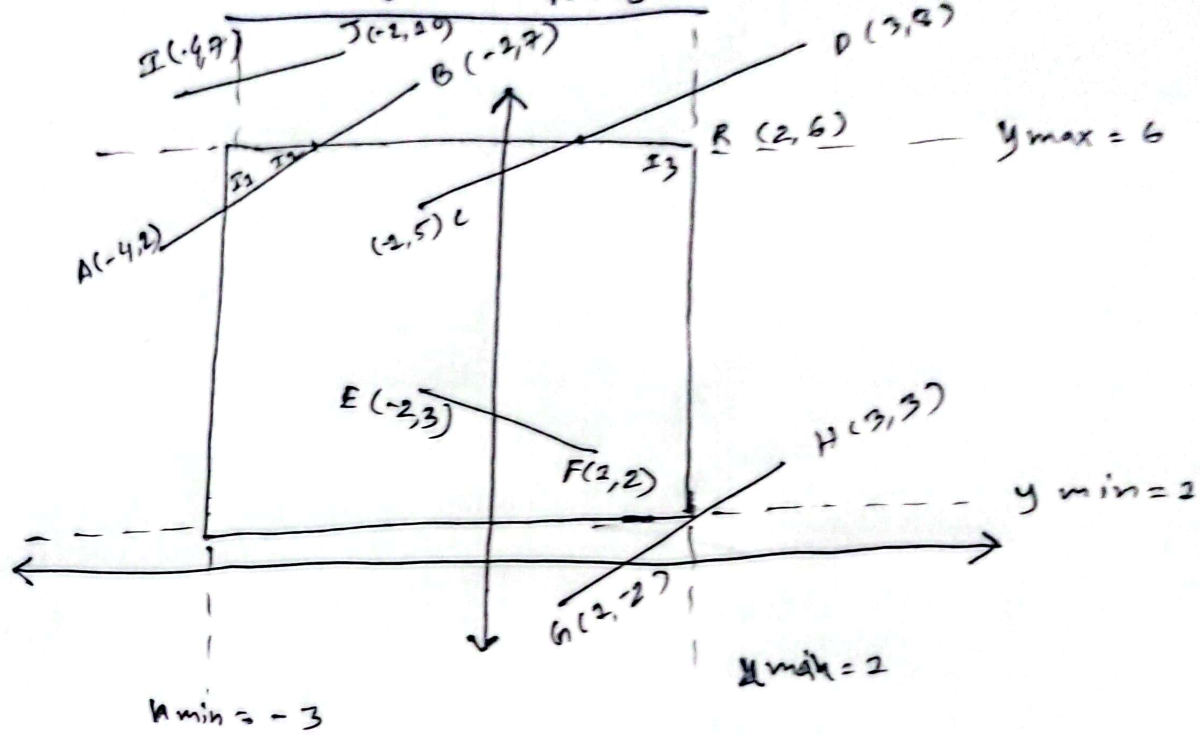
(b) We multiply $R_{\theta_I} \cdot R_{\theta_x J}$ to obtain the matrix

$$\begin{pmatrix} \cos\theta_y & 0 & \sin\theta_y & 0 \\ \sin\theta_x \sin\theta_y & \cos\theta_x & -\sin\theta_x \cos\theta_y & 0 \\ -\cos\theta_x \sin\theta_y & \sin\theta_x & \cos\theta_x \cos\theta_y & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

This is not the same matrix as in part a, thus the order of rotation matters.

Viewing & Clipping

5.6



Bit 1 = Sign (y - ymax) = Sign (y - 6)

Bit 2 = Sign (ymin - y) = Sign (1 - y)

Bit 3 = Sign (x - xmax) = Sign (x - 2)

Bit 4 = Sign (xmin - x) = Sign (-3 - x)

Sign (a) = { 1 positive (a) ; 0 other

A (-4,2) → 0001

B (-1,7) → 1000

C (-2,5) → 0000

D (3,8) → 1010

E (-2,3) → 0000

F (1,2) → 0000

G (1,-2) → 0100

H (3,3) → 0010

I (-4,7) → 1001

J (-2,10) → 1000

5.8

E or F = 0000 → visible

I and J = 1000 → not visible

A and B = 0000 → clipping candidate

C and D = 0000 → "

G and H = 0000 → "

5.7

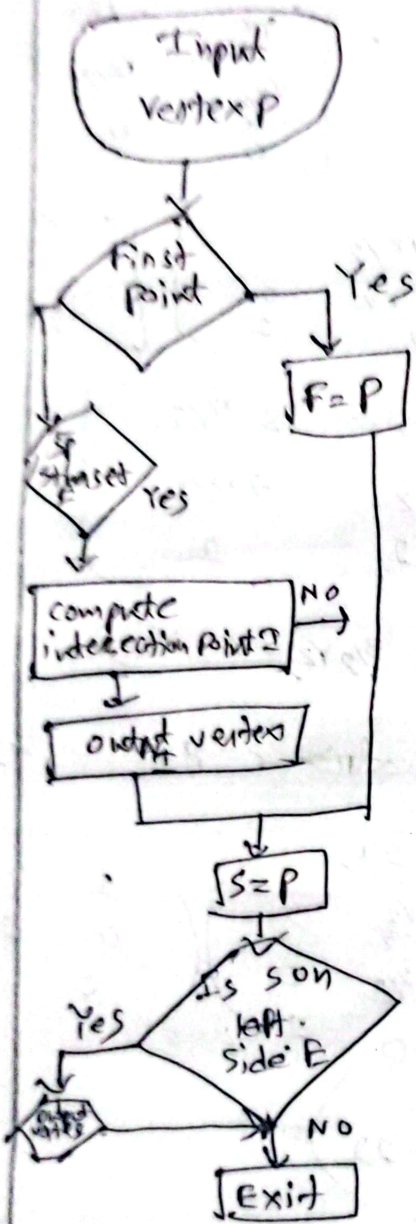
In clipping AB the code for A is 0001. To push the 1 to 0, we clip against the boundary line $x_{min} = -3$. The resulting intersection point is I_1 $(-3, 3\ 2/3)$. We clip (do not display) $\overline{AI_1}$ and work on $\overline{I_1B}$. The code for I_1 is 0000. The clipping catagory for I_1B is 3 since (0000) AND (1000) is (0000), Now B is outside the window, so we push the 1 to a 0 by clipping against the line $y_{max} = 6$.

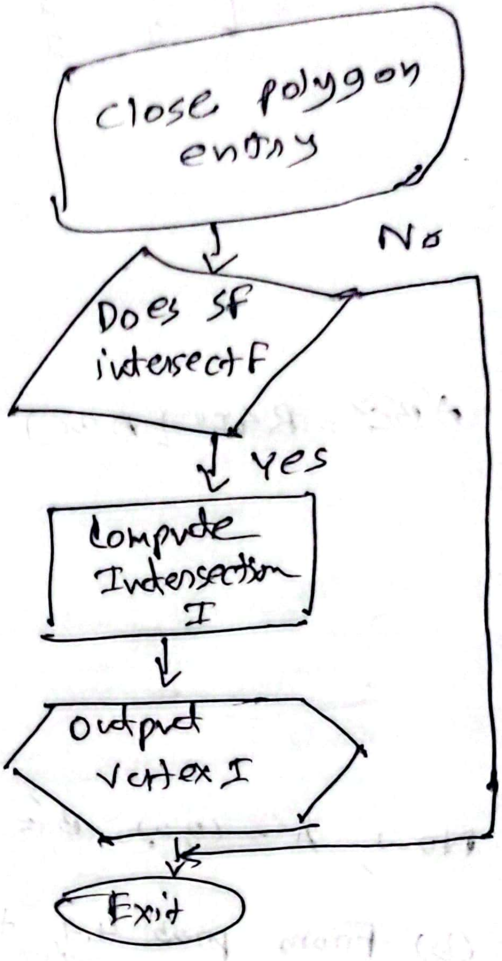
The resulting intersection is I_2 $(-1\ 3/5, 6)$. Thus $\overline{I_2B}$ is clippinting. The code for I_2 is 0000. The remaining segment $\overline{I_1I_2}$ is displayed since both endpoints lie in the window.

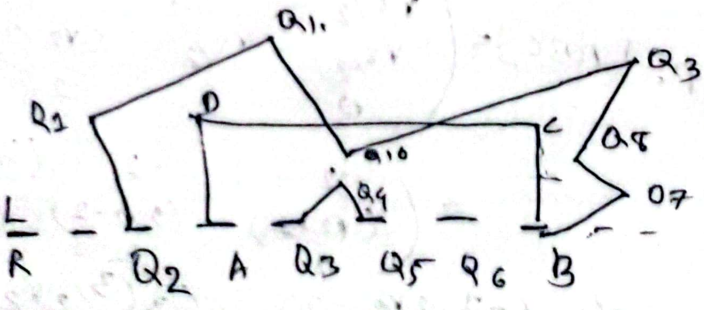
For clipping $\overline{CD}$, we start with D, it is outside the window is 1010. We push the first 1 to a 0 by clipping against the line $y_{max} = 6$. The resulting intersection I_3 is $(1/3, 6)$ and code 0000. For $\overline{GH}$, we push 1 to a 0 by clipping against line $y_{min} = 1$. Since (0010) AND (0010) = 0010.

5.26

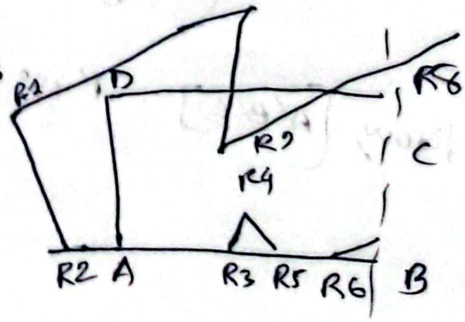
Clip the polygon P_1 ---- P_8 using Sutherland - Hodgman algo:





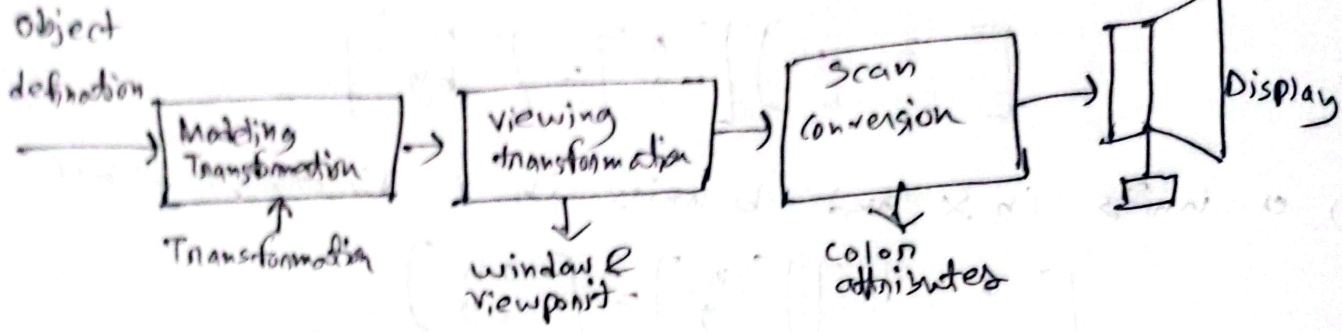


Clip against AB

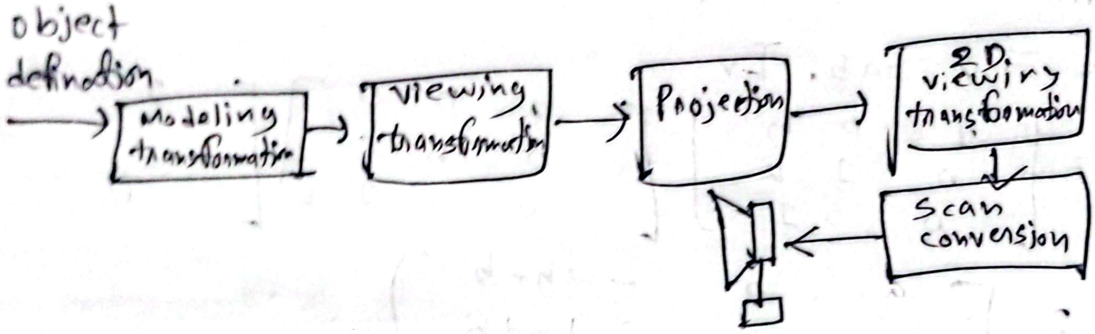


Clip against BC

2D Graphics Pipeline:



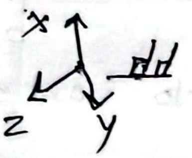
3D Graphics Pipeline:



Projection

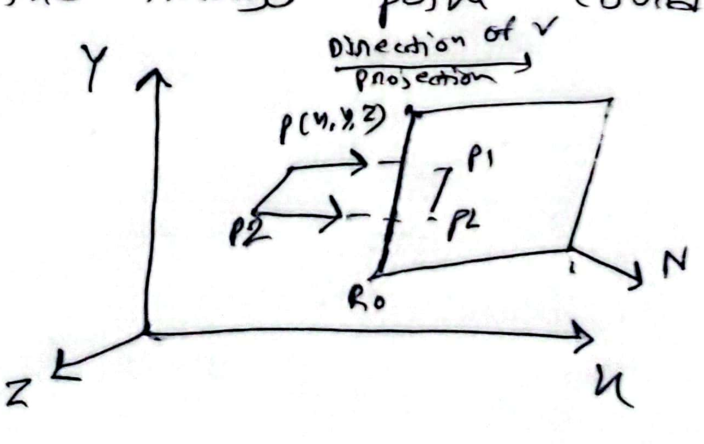
* Taxonomy of projection:

- Projection
  - Parallel (projection line parallel to each other)
    - Oblique (Chend with same angle with view point)
      - General
        - Cavalier
        - cabinet
    - Orthographics (90° with view point)
      - multiview
      - Axonometric (not make 90° angle with principle axis)
  - perspective (projection approaching to the fix point)
    - one point (one principle vanishing point)
    - two point
    - three point



* mathematical description of parallel projection:

Projection vector V, view plane reference point R_0, view plane normal N, The object point P is located at (x, y, z) in world coordinate. determine the image point coordinated P(x', y', z')

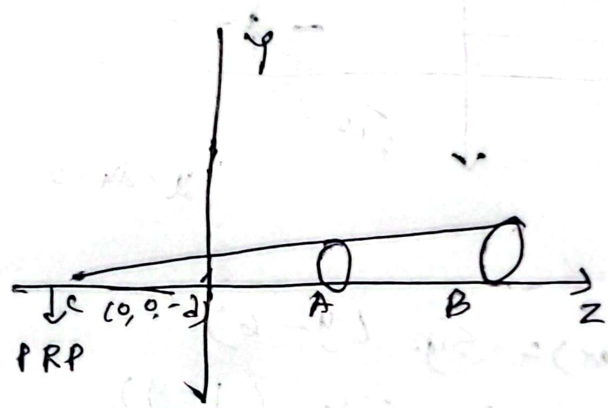


* perspective anomalies:

① projection lines are drawn from the object points towards view plane.

* PRP or center projection point where projection line coverage

* perspective = PRP finite point



② Vanishing plane: parallel lines that are not parallel to the viewing plane coverage to vanishing point.

8.8 Three dimentional cohen- Sutherland algo:

a) For the cononical parallel view volume each bit is set (1) true or flase (0)

Bit 1 = endpoint is above view volume = Sign (y-1)
Bit 2 = " " below " " = Sign (-y)
Bit 3 = " " right to " " = Sing (x-1)
Bit 4 = " " left to " " = Sign (-x)
Bit 5 = " " behind " " = Sign (z-1)
Bit 6 = " " infront " " = Sign (-z)

Recall that sign (a) = 1 if a is poisitive, 0

b) for cannonical perspective view volume

Bit 1 = sign (y-z)
Bit 2 = sign (-z-y)
Bit 3 = sign (x-z)
Bit 4 = Sign (-z-x)
Bit 5 = Sign (z-1)
Bit 6 = sign (z_f - z)

Category (1) visible if both region code are 000000 (2) Not visible if logical AND of the region code is not 000000 (3) logical AND is 000000 → Clipping candidate.

chapter - 10

10.1

Given points P_1 (1, 2, 0), P_2 (3, 6, 20), P_3(2, 4, 6)

and viewpoint C (0, 0, -10)

The projection line from C to P_1 is $x = t$, $y = 2t$

$z = -10 + 10t$

P_2 lies on line ($x = 3$ when $t = 3$)

compare the parameter value P1 (at $t = 1$) is in front of P2 (at $t = 3$). P1 blocks P2 view.

P3 does not lie on the projection line. so, P3 not block or hidden by P1 and P2

10.4

Hidden surface algo are crucial because they determine which parts of objects or scenes should be visible, making images look realistic by hide objects that are behind other images.

10.5

steps required to determine whether any point P1 hides another point P2 view

(i) wheather two points lies on the same projection line

(ii) If they do, which point is in front of the other.

10.6

In parallel projection there are no vanishing points. P (a, b, z) any points with the same x and y coordinates (a, b) project onto the same line. To determine which point is closer, we only need to compare z coordinates

10.7

The z-buffer is like a depth map behind our image. It keeps track of how far away the closest object is for each pixel (x, y) when drawing objects, the z-buffer compares the objects depth to the stored value. If the object is closer, it updates depth & replaces the color with new object. This creates a realistic images where closer objects block the view of distant ones.

Z buffer algorithm (10.2)

(i) Initialize the screen and z-buffer

(ii) For each polygon:
- Calculate depth at each pixel it covers.
- If the depth is less than the z-buffer value, update it & set pixel color.

It keeps track of ~~(?)~~ objects depth for each pixel. Closer object wins, get shown, hides farther ones.

10.8

Display space 2x2 region $0 \le x \le 2$, $0 \le y \le 2$, $0 \le z \le 1$

A is the front of square B coordinate. A_1 (1/2, 1/2, 0), A_2 (2, 2, 0) in B z buffer depth (1/2), B_1 (0, 0, 1/2), B_2 (2, 0, 1/2) B_3 (2, 2, 1/2), B_4 (0, 2, 1/2)

	0	1
1	y	a
0	a	y

a = A color, y = background color. The range of B display.

	0	1
1	b	b
0	b	b

b = B color. we apply z buffer algorithm object A & B. z buffer depth z = 1, background color = y.

Frame buffer

	0	1
1	y	y
0	y	y

z buffer

	0	1
1	1	1
0	1	1

Apply the algo to object B

a) Depth value for B pixel (0,0) is 1/2, z buff (0,0) = 0 pixel (0,0) is unchanged

	0	1
1	y	a
0	a	y

Frame buffer

	0	1
1	1	0
0	0	1

z buffer

b) Depth value of B pixel (1, 0) is 1/2.
present z value is 1. So z buff set to
1/2. Present z value is 1. so z buff ~~at~~
to 1/2 at (1,0) pixel

	0	1
1	y	a
0	a	b

Frame buffer

	0	1
1	1	0
0	0	1/2

z buffer

c) Depth value of B at pixel (0,1) is
1/2. Present z buff value is 1. So pixel
(0,1) is set to B.

	0	1
1	b	a
0	a	b

Frame buffer

	0	1
1	1	0
0	0	1/2

z buffer

d) Depth value of B at pixel (1,1) is
1/2. Present z buff value is 0. so color
pixel unchanged.

	0	1
1	b	a
0	a	b

Frame buffer

	0	1
1	1/2	0
0	0	1/2

z buffer

Apply the algo to object A

a) Depth of A pixel (0,0) is z = 0, z buff (0,0) is changed to 0, pixel (0,0) → Color of A

Frame buff

	0	1
1	y	y
0	a	y

z buffer

	0	1
1	1	1
0	0	1

b) object A not seen from pixel (1,0), z buffer

z buff → unchanged

Frame buff

	0	1
1	y	y
0	a	y

	0	1
1	1	1
0	0	1

c) object A not seen from pixel (0,1) → unchange

Frame buff

	0	1
1	y	y
0	a	y

z buffer

	0	1
1	1	1
0	0	1

d) Depth of A pixel (1,1) is 0 z buff value 1 > 0 pixel has the color of A (1,1)

Frame buff

	0	1
1	y	a
0	a	y

z buffer

	0	1
1	1	0
0	0	1