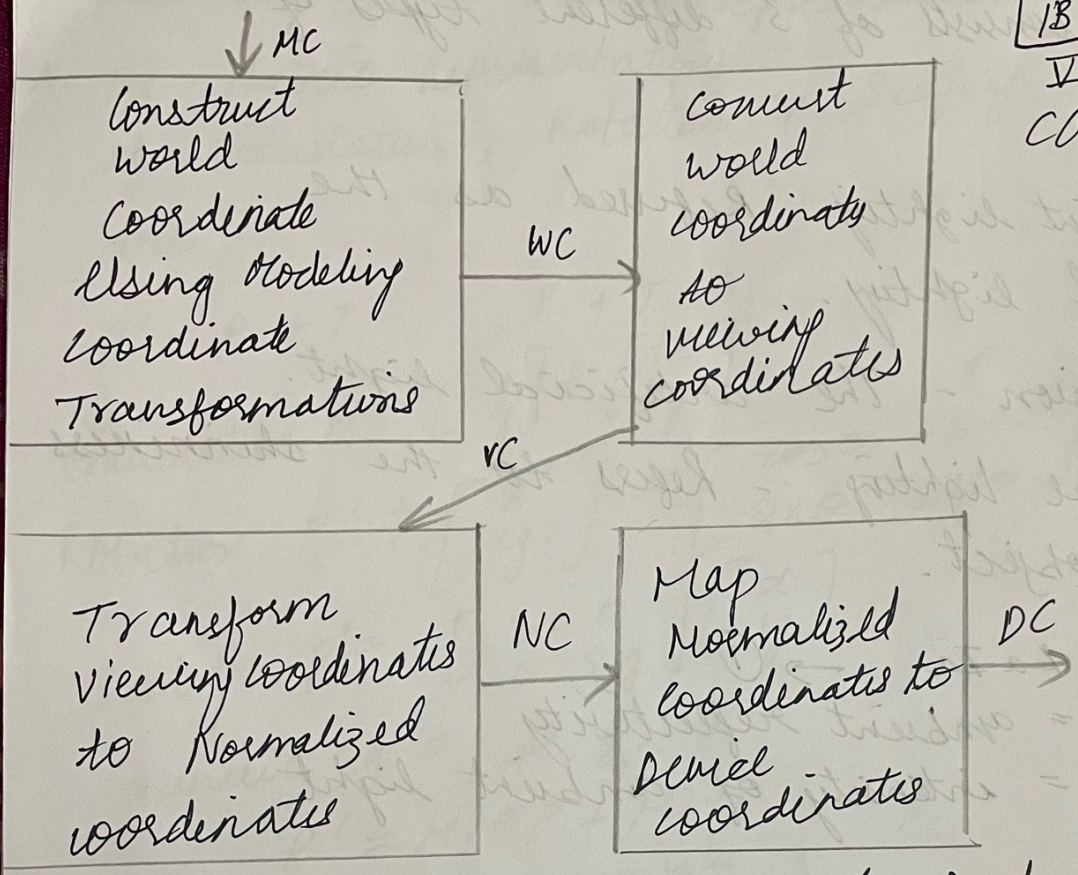
① 2D Pipeline

KRISHA . M

KRISHA . M
1BY20CS089
VI 'B'
CGV ASSIGNMENT



- We could set up a seperate 2-d, viewing coordinate reference frame for specifying clipping window.
- Systems use normalized coordinates in the range from 0 to 1, others used a normalized range from -1 to 1.
- Clipping is usually performed in the normalized coordinates.

② Build Phony lighting Model with Equations

light consists of 3 different types of light.

⇒ Ambient lighting Refered as the natural lighting.

⇒ Diffusion - The artificial light.

⇒ Specular lighting - Refers to the shininess of the object.

$I_{amb} = K_a I_a \rightarrow (1)$

K_a = ambient reflectivity

I_a = intensity of ambient light

Similarly

$I_{diff} = K_d I_p \cos(\theta) \rightarrow (2)$

$= K_d I_p (N \cdot L)$

$I_{spec} = K_s I_l \cos^n \phi$

∴ The Phony Model gives us the equation of all combined

∴ Total intensity $I = K_a I_a + K_d I_p \cos\theta + K_s I_l \cos^n \phi$

Apply homogeneous coordinates for translation, rotation and scaling via matrix representation.

A. The Matrix Representations of Translation, Rotation & Scaling are

$P' = P + T$ $\quad P + T$

↓ Translation $P' = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} tx \\ ty \end{bmatrix}$

Rotation $P' \Rightarrow \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$ $\quad R \cdot P$

Scaling $P' = \begin{bmatrix} sx & 0 \\ 0 & sy \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$

Generic Eq $= S \cdot P$

$P' = M_1 * P + M_2$

But $x = \frac{xh}{h}$, $y = \frac{yh}{h}$

$h = 1$

$\therefore$ consider $(h*x, h*y, h)$

Translation $\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & tx \\ 0 & 1 & ty \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$

Rotation $\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} sx & 0 & 0 \\ 0 & sy & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$

Scaling $\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$

(4.) Differences between Raster and Random scan displays.

Raster Scan	Random Scan
1) Produces jagged lines that are plotted as a discrete point sets.	Random system produces smooth lines drawing.
• less expensive	• More expensive
• Modification difficult	• Modification easy
• Resolution low	• Resolution high
• solid pattern is easy to fill.	• Solid pattern is difficult to fill.

(5) Demonstrate OpenGL functions for window management using GLUT.

- glutCreateWindow - Used to create a new window.
- glutCreateSubWindow - Used to create another window within same window.
- glutSetWindow - Used to set an particular id for the window.

- glutGetWindow - Used to get the Window ID.
- glutDestroyWindow - To delete the window that was created.
- glutPostRedisplay - To display the window again & again, continously until forcibly closed.
- glutReshapeWindow - Used for transformation of world coordinate ~~to~~ view coordinates and displaying it.
- glutFullScreen - To represent Window in Full screen mode.
- glutPopWindow / glutPushWindow - Works just like a matrix on window.
- glutHideWindow - To hide the window from being displayed on screen.
- glutDisplayFunc - To display
- glutMainLoop
- init()

6. OpenGL visibility Detection Functions

(a) openGL polygon culling Functions
Remove backface, frontface of an object.

glCullFace(mode);
glEnable(GL_CULL_FACE);
glDisable(GL_CULL_FACE);

(b) Depth-Buffer Functions
glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB | GLUT_DEPTH)

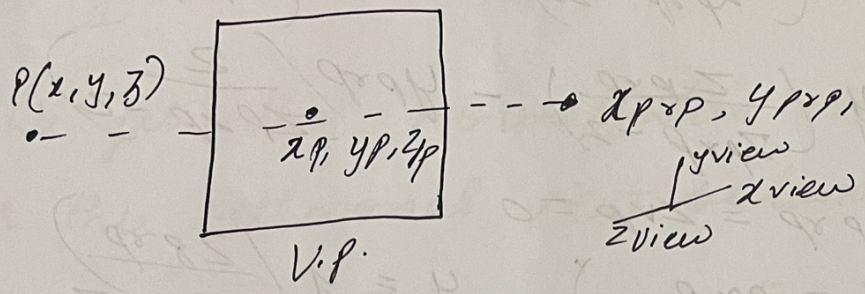
glClear(GL_DEPTH_BUFFER_BIT)
This works as initialization function for depth buffer and refresh buffer.
glDepthRange(nearNormDepth, farNormDepth)
glClear(GL_DEPTH_BUFFER_BIT)
glClearDepth(MaxDepth)
glEnable(GL_DEPTH_TEST)
glDisable(GL_DEPTH_TEST)

(c) OpenGL Wireframe surface visibility Methods
glPolygonMode(GL_FRONT_AND_BACK, GL_LINE), Visible & hidden edges displayed.

(d) OpenGL Depth Cueing Fns.
glFogi(GL_FOG_MODE, GL_LINEAR)
glEnable(GL_FOG)

To increase or decrease the brightness.

With special cases discussed with Perspective Projections.



Consider

$$x' = x - (x - x_{p'p})u$$
$$y' = y - (y - y_{p'p})u$$
$$z' = z - (z - z_{p'p})u$$

$$u = \frac{z_{vp} - z}{z_{p'p} - z}$$

$$x_p = x\left(\frac{z_{p'p} - z_{vp}}{z_{p'p} - z}\right) + x_{p'p}\left(\frac{z_{vp} - z}{z_{p'p} - z}\right)$$

$$y_p = y\left(\frac{z_{p'p} - z_{vp}}{z_{p'p} - z}\right) + y_{p'p}\left(\frac{z_{vp} - z}{z_{p'p} - z}\right)$$

Special cases:

① $x_{p'p}, y_{p'p} = 0$

$$x_p = x\left(\frac{z_{p'p} - z_{vp}}{z_{p'p} - z}\right)$$

$$y_p = y\left(\frac{z_{p'p} - z_{vp}}{z_{p'p} - z}\right)$$

② $x_{p'p}, y_{p'p}, z_{p'p} = (0,0,0)$;

$$x_p = x\left(\frac{z_{vp}}{z}\right)$$

$$y_p = y\left(\frac{z_{vp}}{z}\right)$$

3) $z_{vp} = 0$

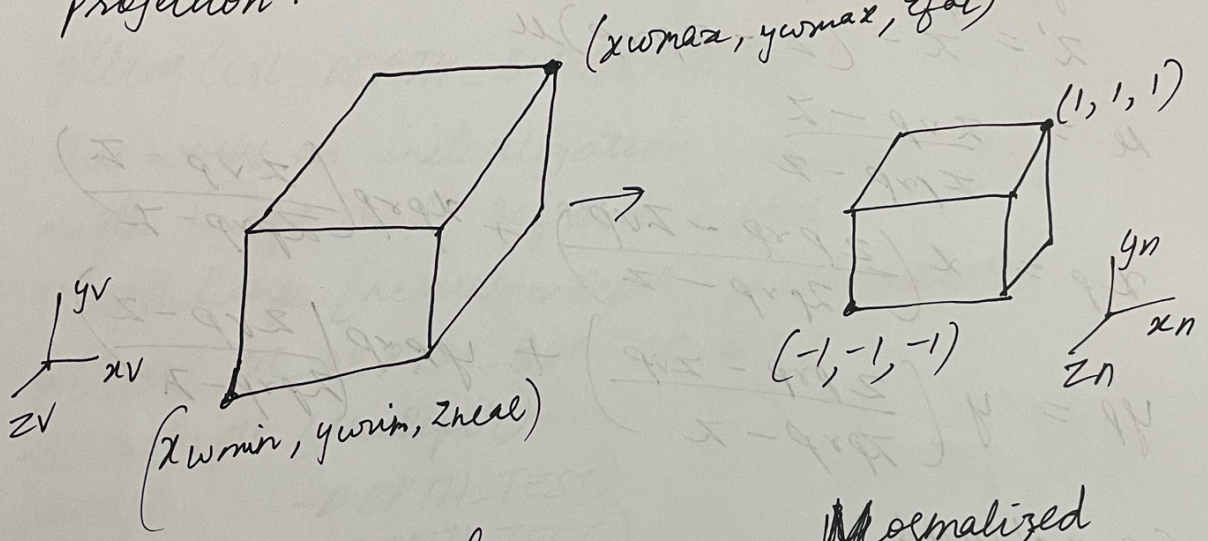
$$x_p = x\left(\frac{z_{prp}}{z_{prp} - z}\right) - x_{prp}\left(\frac{z}{z_{prp} - z}\right)$$

$$y_p = y\left(\frac{z_{prp}}{z_{prp} - z}\right) - y_{prp}\left(\frac{z}{z_{prp} - z}\right)$$

4) $x_{prp} = y_{prp} = z_{vp} = 0$

$$x_p = x\left(\frac{z_{prp}}{z_{prp} - z}\right), \quad y_p = y\left(\frac{z_{prp}}{z_{prp} - z}\right)$$

(8) Explain normalization for an orthogonal projection.



Orthogonal Projection View Volume

Normalized View Volume

- We consider a unit cube for this normalized view volume with each x, y, z coordinates normalized in the range 0 to 1

- Another normalization transformation approach is to use symmetric cube with coordinates -1 to 1 ⑧

∴ We get the normalization transformation for the orthogonal view volume

$$M_{ortho,norm} = \begin{bmatrix} \frac{2}{x_{wmax}-x_{wmin}} & 0 & 0 & -\frac{x_{wmax}+x_{wmin}}{x_{wmax}-x_{wmin}} \\ 0 & \frac{2}{y_{wmax}-y_{wmin}} & 0 & -\frac{y_{wmax}+y_{wmin}}{y_{wmax}-y_{wmin}} \\ 0 & 0 & \frac{-2}{z_{near}-z_{far}} & \frac{z_{near}+z_{far}}{z_{near}-z_{far}} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

⑨ Explain Bezier curve and its properties with equations.

- Bezier curves are parametric curves that are generated with the help of control points. It is widely used in graphics and other related industry.