CG Assignment

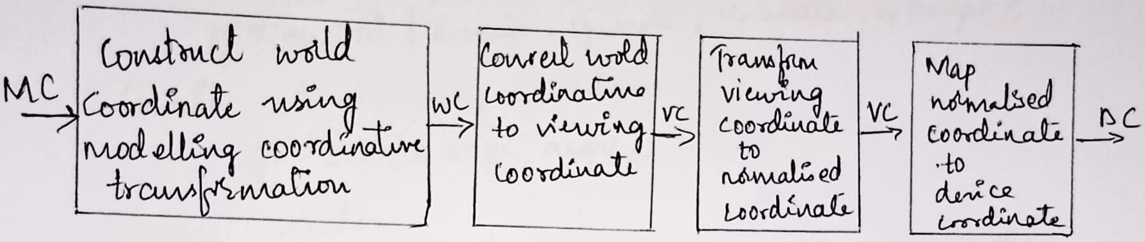
JHANSI PATIL B.S

CSE 6th B

1BY21CS408

① Build a 2D transformation pipeline and also explain OpenGL viewing functions.

→


MC
Construct world coordinate using modelling coordinative transformation
WC
Convert world coordinatino to viewing coordinate
VC
Transform viewing coordinate to normalised coordinate
VC
Map normalised coordinate to device coordinate
DC

→ A section of 2-D scene that is selected for display called clipping window and all parts outside are clipped off.

→ Mapping 2D world coordinate scene description to device coordinate is called 2D viewing transformation or window to viewport transformation.

→ Once the world-coordinate scene has been constructed, we could set 2D viewing coordinate reference frame for specifying clipping window.

→ To make view process independent of requirement of output device, system convert description to normalixd coordinate from 0 to 1 and others use -1 to 1.

→ Finally, normalised coordinate map to device coordinate.

Viewing functions:

OpenGL projection mode:

glMatrixMode (GL_PROJECTION);

This designates the projection matrix a current matrix, which is originally set to identity matrix.

Clipping Window:

To define 2D clipping window gluOrtho2D (xwmin, ywmin, ywmax, ywmax)

OpenGL viewport functions:

glViewPort (xvmin, yvmin, vp width, vp Height);

Create GLutDisplay window

glutInit (& argc, argv);

Display window

glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB);

glut InitWindowSize (500, 500);

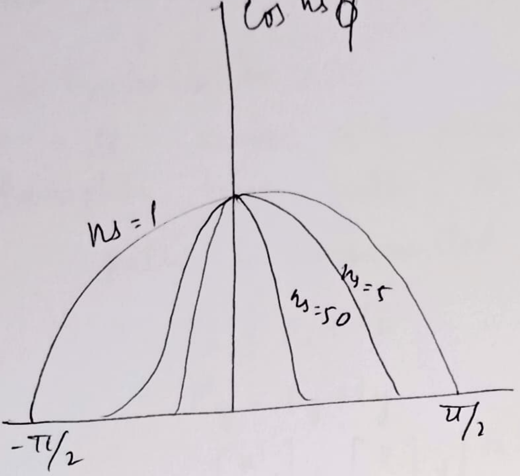
glutInit windowPosition (0, 0);

glClearColor (r, g, b, α);

glClear Index (index);

② Build Phong Lighting mode with equations.

→ Phong reflection is an emperical model of local illumination. It describes the way a surface reflects light as combination of diffuse reflection of rough surface with specular of shiny surface. It is based on phong's informal observation that shiny surface have small intense specular highlights, while dull surface have larger highlights That off gradually.

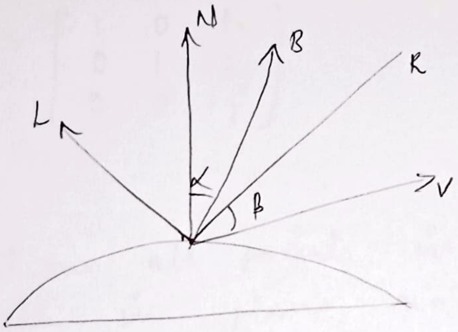


- Phong model sets the intensity of specular reflection of $\cos^{ns}\phi$.

$I_{specular} = w(\theta) I_l \cos^{ns}\phi$

$0 \leq w(\theta) \leq 1$ is specular reflection coefficient.

For most opaque materials specular reflection coefficient is nearly k_s.



$$I_{specular} = \begin{cases} k_s I_l (V, R)^{ns} & V \cdot R > 0 \text{ and } N \cdot L > 0 \\ 0 & \text{Otherwise} \end{cases}$$

$R = (2 N L) N - L$

Normal N vary at each point.

$$H = \frac{L+V}{|L+V|}$$

If the distance b/w light source and viewer are relatively for then α is constant.

H is direction yeilding maximum specular reflection in viewing direction V if surface normal N would coincide with H. If v is coplanar with R and L then $\alpha = \phi/2$

③ Apply homogenous coordinate for translation, rotation and scaling via matrix representation.

→ Translation:

It moves all points in a object along some straight line path to new position.

path is represented by vector.

$P'_x = P_x + t_x$

$P'_y = P_y + t_y$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

In homogenous coordinates.

$$= \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

Rotation:

It repositions all points in an object along a circular path in plane centeredent pivot center.

$\cos\phi = x/r \qquad \sin\theta = y/x$

$x = r\cos\phi \qquad y = r\sin\phi$

$x' = x\cos\theta - y\sin\theta$

$y' = x\sin\theta + y\cos\theta$

$P' = R.P$

$$R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

In homognous coordinate:

$$R = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Scaling:

It is used to change the size of an

an object.

$$p'_n = S_x \cdot p_x$$

$$p'_y = S_y \cdot p_y$$

$$p' = S \cdot P$$

$$S = \begin{bmatrix} S'x & 0 \\ 0 & Sy \end{bmatrix}$$

Homogenous coordinate $= \begin{bmatrix} Sx & 0 & 0 \\ 0 & sy & 0 \\ 0 & 0 & 1 \end{bmatrix}$

(4) Outline difference between restar scan display and random scan displays.

→ Raster scan display:

- Electron beam is swept across one row data from top to bottom.
- As it moves across each row row, beam intensity is turned on & of to create a pattern of illuminated spots.
- Scanning process called refreshing. Complete scanning of screen is normally called frame.
- Refreshing rate is frame rate it is 60 Hz to 80 Hz.
- Picture definition is stored in a memory area called frame buffer. This stores the intensity values for all screen points called pixel.

Random scan display:

- When operated as random scan display unit

CRT has election beam distributed only to those pairs of sceen where picture has to be displayed.

- Pictures are generated as line drawings with electron beam distributed tracing out the component lines one after the other.
- Also known as vector displays. Component line of picture can be drawn and refreshed by a random scan system in any specified order.
- Refresh rate depends on number lines to be displayed on that system.

⑤ Display window management using GLUT.

→ step 1: Initialisation of GLUT.

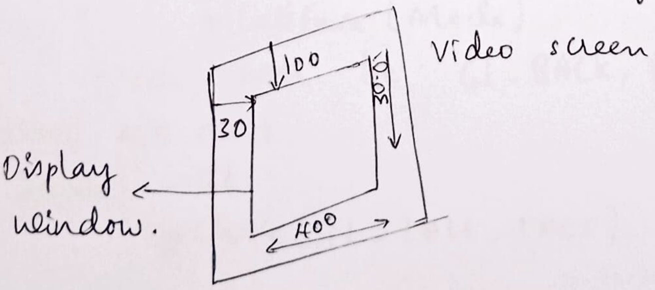
- We use GLUT.
- We perfrm initialisation

```
glutInit(& argc, argv);
```

step 2: Title.

```
glutCreateWindow ("An example");
```

There is single argument for this function can be any, character string that we want character string that we want to display for window.



```
glutDisplayFunc (line segment):
```

It used to call the display Function repeatedly. Name of the function to be called line segment.

glutMainLoop():

It is last Line. It displays the initial graphic and put program to infinite loop.

glut InitWindirecPosition:

It is used to specify upper left corner of the display window.

glutInitWindowsize:

It is used to set initial pixel with height & width of display window.

glutInitDisplaymode(GLUT SINGLE|GLUT_RGB):

It is used for and color mode like red, green and blue component to select to select color value.

(b) Explain openGL visibility detection functions.

(a). OpenGL polygon filling functions:

Back face removal with functions

glEnable(GL_CULL_FACE);

glCullFace(Mode)

mode can be GL_BACK, GL_FRONT, GL_FRONT AND_BACK.

Disable with

glDisable(GL_CULL_FACE);

(b) OpenGL Depth Buffer functions:

To use openGL depth buffer visibility detection function, we need to modify GLUT initialization function.

glutInit DiplayMode (GLUT_SINGLE | GLUT_RGB | GLUT_DEPTH)
glu Bar (GL_DEPTH_BUFFER_BIT);

We can set status of depth buffer.

glDepthMark (write status);

ⓒ openGL wireframe surface visibility method :

A wire frame display can be obtained in openGL by requesting that only its edges are generated.

glPolygon Model (GL.FRONT_END_BACK, GL_LINE):

ⓓ OpenGL_DEPTH_CURING_function :

It is used to valy the brightness of an object as a function of its distance from viewing position with.

glEnable (GL_FOG);
gl Fogi (GL_FOG_MODE_GL_LINEA);

(7) Write special cases that we discussed with respect to perspective projection transformation Coordinates.

$$x_p = x\left[\frac{2prp - 2vp}{2prp - 2}\right] + x_{prp}\left[\frac{2vp - 2}{2prp - 2}\right]$$

$$y_p = y\left[\frac{2prp - 2vp}{2prp - 2}\right] + y_{prp}\left[\frac{2vp - 2}{2prp - 2}\right]$$

Cases:

0 Projection reference point is limited along zview axis,

$z_{prp} = y_{prp} = 0$

$x_p = x\left[\frac{z_{prp} - z_{vp}}{z_{prp} - z}\right]$ $y_p = y\left[\frac{z_{prp} - z_{vp}}{z_{prp} - z}\right]$

① when plojection reference point is at cooldinate origine $O(x_{prp} \cdot y_{prp} = z_{prp}) = (0,0,0)$

$x_p = x\left[\frac{z_{vp}}{z}\right] \cdot y_p = y\left(\frac{z_{vp}}{z}\right)$

③ If view plane is uv plane and no restriction on placement of projection reference point.

$z_{vp} = 0$

$x_p = x\left[\frac{z_{prp}}{z_{prp} - z}\right] - x_{prp}\left[\frac{z}{z_{prp} - z}\right]$

$y_p = y\left[\frac{z_{prp}}{z_{prp} - z}\right] - y_{prp}\left[\frac{z}{z_{prp} - z}\right]$

④ If uv plane is uv projection references point on zview axis,

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

$x_p = x\left[\frac{z_{prp}}{z_{prp} - z}\right]$

$y_p = y\left[\frac{z_{prp}}{z_{prp} - z}\right]$

⑥ Explain Bezier curve equation along with equation along with properties.

→ Deneloped by French engineer pierre Beizer for use in Renault automobile bodies.

→ It has number of properties that make them high of useful for curve and surface design.

* It can be filled to any number of control points.

Equations:

$P_x = (x_k, y_k, z_k)$ P_k = genetal (init) control point positions.

P_u = position vector that describes path.

$$p(u) = \sum_{k=0}^{n} p_k \, BEZ_{k,n}(u)$$

$BEZ_{k,n}(u) \; C(n,k), \; u^k (1-u)^{n-k}$ is Bernstein polynomial

$$C(n,k) = \frac{n!}{k!(n-k)!}$$

properties:

→ Basic function are real.

→ Degree of polynomial is one less than number of control point.

→ Curve generally follows shape of defining polygon.

→ It connects first and last Control points.

$p(0) = p_0$

$p(1) = p_n$

→ Curve lies within convex null of control points.

(9) Explain normalisation transformation for orthognal projection.

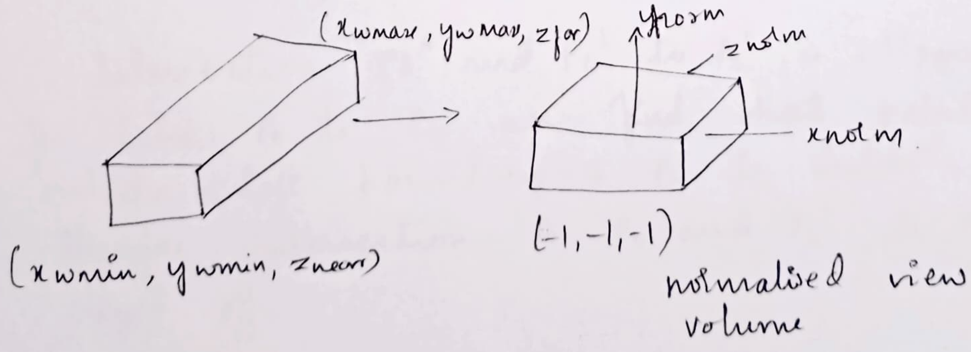
→ We assume that orthogonal projection view volume to mapped into symmetric normalization cube within left-handed reference frame. Also x-coordinate for near and far position is denoted as znear & zfar respectively this position $(x_{min}, y_{min}, z_{near})$ is mapped to normalised position $(-1,-1,-1)$ & position $(x_{max}, y_{max}, z_{far})$ is mapped to $(1,1,1)$.

Transforming rectangular parallel piped view volume to normalised cube is similar to method for converting the clipping method into normalised symmetric square.

Normalization transformation for view volume.

$$= \begin{bmatrix} \frac{-2}{xw_{max} - yw_{min}} & 0 & 0 & \frac{xw_{max} + xw_{min}}{xw_{max} - xw_{min}} \\ 0 & \frac{2}{y_{max} - y_{min}} & 0 & -\frac{yw_{max} + yw_{min}}{yw_{max} - yw_{min}} \\ 0 & 0 & \frac{-2}{z_{near} - z_{far}} & \frac{z_{near} + z_{far}}{z_{near} - z_{far}} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The matrix is multiplied on right by composite viewing transformation R-T to produce complete transformation from world coordinates to normalise orthogonal projection coordinates.



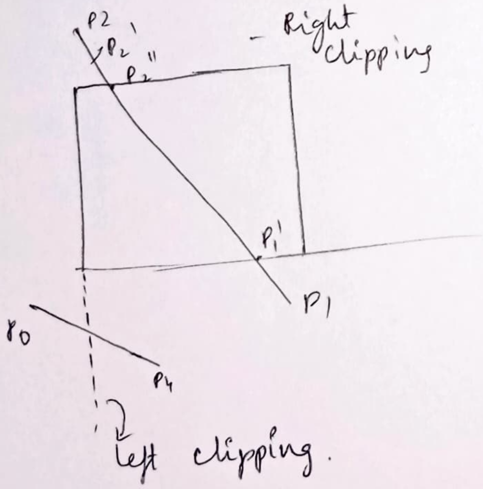
(10) Explain cohen Sutherland line clipping algorithm.

→ Every line endpoint in picture is designed with four digit binary code called region code and each bit it used to indicate where point lies include or outside.

1001	1000	1010
0001	0000	0010
0101	0100	0110

Once we established region code for all line endpoint we determine where they completely insider or not.

when OR operation between 2 endpoints, is false, line inside window AND operation between 2 endpoints, is true, completely outside clipping window if it is not complexity inside, it lies partially outside.



Intersection P_2'' and P_2' to P_2'' is clipped off. for line P_0 to P_4 we find that point P_0 is outside left boundary & P_4 is inside. Therefore, intersection is P_0 and P_3^3 to P_3' is clipped off.

By checking the region code F8 and i4 are find the remainder of line is below clipping Line can be obtained by

$$y = y_0 + m(x - x_0)$$

$$M = y_{end} - y_0 / x_{end} - x_0$$

$$x = x_0 + \left(\frac{y - y_0}{m}\right)$$