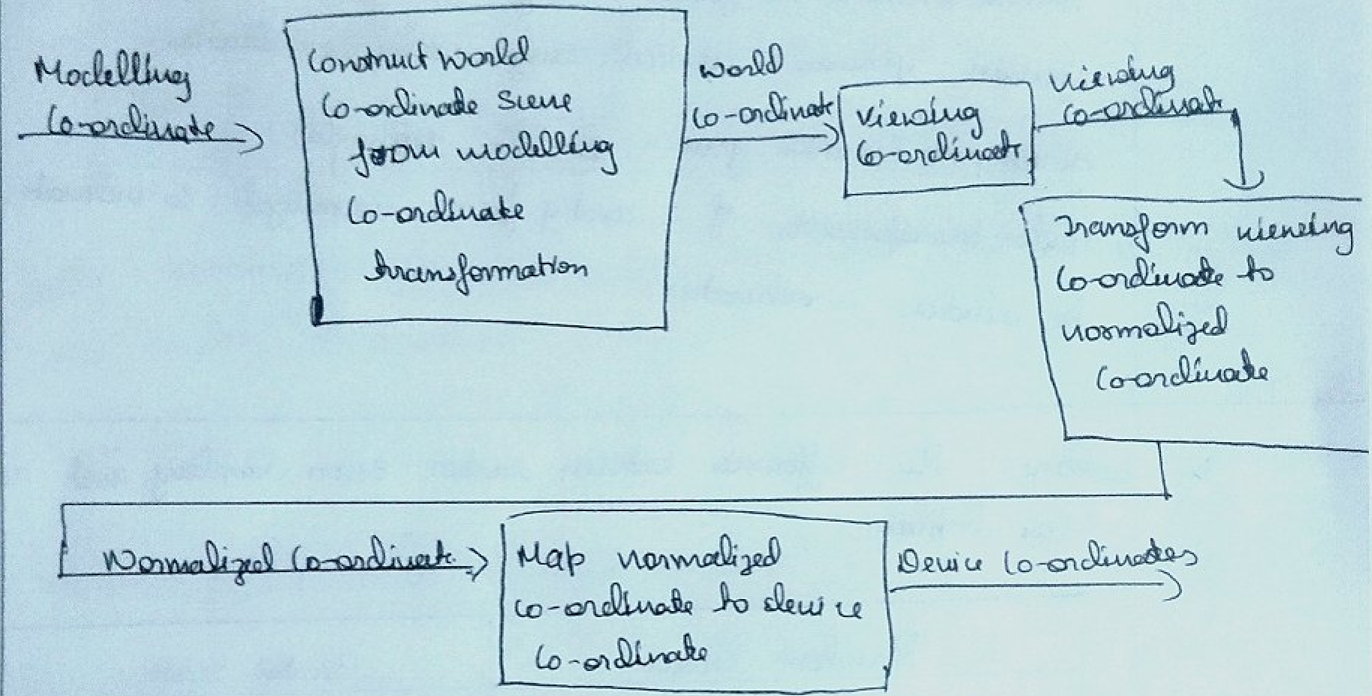
Assignment

1. Build 2D viewing transformation pipelines and also explain open GL 2D viewing functions



change modelling co-ordinate to world co-ordinate by applying modelling transformation. change world co-ordinate to viewing co-ordinate to by determing visible parts. change viewing co-ordinate to normalized co-ordinates and further to device co-ordinate by clipping and determining pixels

OpenGL 2D viewing functions

glt sets the current matrix mode

glt can assume one of the two values

GL Modelview

Applies subsequent matrix operations to modelview matrix stack.

GL - PROJECTION

Applies subsequent matrix operation to projection matrix stack

glOrtho2D (xwmin, xwmax, ywmin, ywmax);

specifies the viewing window

xwmin, xwmax :- horizontal range world co-ordinates

ywmin, ywmax : vertical range world co-ordinates

glViewport (xvmin, yvmin, vpwidth, vpheight)

specifies transformation of x and y from normalized co-ordinate to window co-ordinates

4. Outline the differences between raster scan display and random scan displays.

Random Scan	Raster scan
The resolution of Random scan is higher than raster scan	While the resolution of raster scan is lower than random scan
It is costlier than raster scan	Cost is lesser
raster scan Alteration is easy in comparison of raster scan	Any alteration is not easy

Interweaving is not used	Interweaving is used
It is suitable for applications requiring polygon drawing	It is suitable for creating realistic scenes

3. Apply homogeneous co-ordinate for translation, rotation and scaling with matrix representation

A. Translation $P = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} x \begin{bmatrix} x & x \\ 0 & y \end{bmatrix}$

rotation $P' = \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{bmatrix} \cos\theta , & \sin\theta \\ \sin\theta . & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

Scaling $*P' = \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} Sx & 0 \\ 0 & SY \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

each cartesian co-ordinate (x, y) with homogeneous co-ordinate (x_n, y_n, h) where $x = x_n/h, y = y_n/h$

$(h * x, h * y, h)$

set $h = 1$

$(x, y, 1)$

Homogeneous co-ordinate representation for ~~translating~~ translation scaling and rotation are as follows

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

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

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

→ explain Bezier curve equation along with properties.

A. For N+1 control-point positions, denoted as $p_k = (x_k, y_k, z_k)$ with k varying from 0 to n. These co-ordinate points are blended to produce position vector $P(u)$, which describes the path of an approximating Bezier polynomial function between p_0 and p_n.

$$P(u) = \sum_{k=0}^{n} p_k \, BEZ_{k,n}(u), \quad 0 \leq u \leq 1$$

$$BEZ_{k,n}(u) = C(n,k)\, u^k (1-u)^{n-k}$$

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

eqn $P(u)$ represents a set of three parametric equations for the individual curve co-ordinates

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

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

→ If the view plane is the uv plane and there are no restrictions on the placement of the projection reference point, then we have

$$z_{vp} = 0$$

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

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

→ With uv plane as the view plane and the projection refered point on the z view axis, the perspective equation are

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

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

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

→ explain open GL visibility selection function

glEnable (GL-CULL-face)

$$x(u) = \sum_{k=0}^{n} x_k \, BEZ_{k,n}(u)$$

$$y(u) = \sum_{k=0}^{n} y_k \, BEZ_{k,n}(u)$$

$$z(u) = \sum_{k=0}^{n} z_k \, BEZ_{k,n}(u)$$

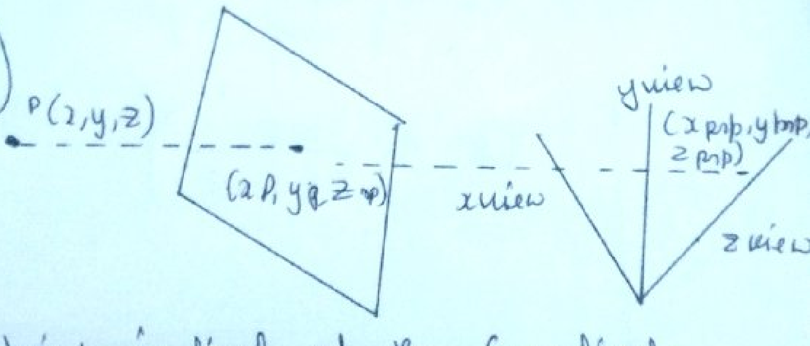
In most cases, a Bezium curve is a polynomial of degree that is one less than designated number of control points. Three points generate a parabola, four points a cubic ~~curve~~ curve and so forth.

→ Write the special cases that we discussed with respects to perspective projection transformation co-ordinate

*. if projection reference point is on z view, means $x_{prp} = y_{prp} = 0$

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

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



2. The projection reference point is fixed at the co-ordinate origin, and

$$(x_{prp}, y_{prp}, z_{prp}) = (0,0,0)$$

It is used for turning culling on

glcullFace (mode)

It specifies what to cull

mode = GL-Back is default

glFrontFace (vertexorder)

It is for order of vertices

Orientation in changes

vertexorder = GL-CW or GL-CCW

GL-CW is for clockwise direction (front)

GL-CCW is for counterclockwise direction (back)

GL-CCW is default

create depth buffer by setting GLUT-DEPTH flag in glutInitDisplay

() or the appropriate flag in the

PIXELFORMATDESCRIPTOR

enable pre-pixel depth testing with glenable (GL-DEPTH-TEST)

Clear depth buffer by setting GL-DEPTH-Buffer-BIT in

gl clear ().

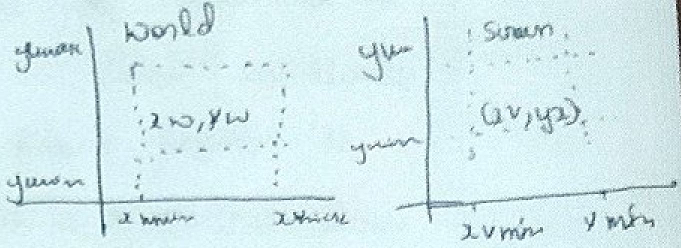
gldepthfunc (condition);

Changes the test used

Condition : GL-LESS [Closer ~~value~~ visible | default]

GL-GREATER [Farther visible)

→ explain normalization transformation for an orthogonal projection



Relative position to send

$$\frac{xv - xvmin}{xvmax - xvmin} = \frac{xw - xwmin}{xwmax - xwmin}$$

$$xv - xmin = (xmax - xmin)\left(\frac{xw - xwmin}{xwmax - xwmin}\right)$$

$$xv - xmin = (xw - xwmin)\left(\frac{xmax - xvmin}{xwmax - xwmin}\right)$$

$$xv = xw\left(\frac{xvmax - xvmin}{xwmax - xwmin}\right) + xmin + \frac{xmin \cdot xmin - xwmin\ xmax}{xwmax - xmin}$$

$$xy = xw\left(\frac{xvmax - xvmin}{xwmax - xwmin}\right) + \left(\frac{xwmax\ xmin - xwmin \cdot xvmax}{xwmax - xwmin}\right)$$

$$xv = xw s_1 + tx$$

where $s_1 = \frac{xvmax - xvmin}{xwmax - xwmin}$

$$tx = \frac{xwmax\ xvmin - xwmin\ xvmax}{xwmax - xvmin}$$

similarly

$$yv = yw\ sy + ty$$

where $sy = \frac{yvmax - yvmin}{ywmax - ywmin}$

$$ty = \frac{yvmax\ ywmin - ywmin\ yvmax}{ywmax - ywmin}$$

$xv = sx\ xw + tx$
$yv = sy\ yw + ty$
} can be written as

$M_{window, normview} = T.S = \begin{bmatrix} sx & 0 & tx \\ 0 & sy & ty \\ 0 & 0 & 1 \end{bmatrix}$

For normalized coordinates,

-1 for xvmin and yvmin

1 for xvmax ~~&~~ and yvmax

$$M_{window, normsquare} = \begin{bmatrix} \frac{2}{xwmax - xwmin} & 0 & -\frac{xwmax + xwmin}{xwmax - xwmin} \\ 0 & \frac{2}{ywmax - ywmin} & -\frac{ywmax + ywmin}{ywmax - ywmin} \\ 0 & 0 & 1 \end{bmatrix}$$

similarly for 3D

$$M_{ortho, norm} = \begin{bmatrix} \frac{2}{xwmax - xwmin} & 0 & 0, & \frac{xwmax + xwmin}{xwmax - xwmin} \\ 0 & \frac{2}{ywmax - ywmin}, & 0, & \frac{ywmax + ywmin}{ywmax - ywmin} \\ 0 & 0 & , \frac{-2}{znear - zfar} & , \frac{znear + zfar}{znear - zfar} \\ 0, & 0, & 0 & , 1 \end{bmatrix}$$

10

→ Demonstrate open GL function for displaying window managment using GLUT

glutInit (&argc, argv)

It is used to initialize GLUT library
glut init window position(xtop left, ytop left)
position of display window on screen

glut init windowsize (dm width, dm height)

size of window
dm width is width of display
dm width is height of display

glut create window ("string")
It is used to create display window with name.
glut display func ()

It sets the display callback for current window
glut init display mode ();

It sets the initial display mode
glut - Reshape func ()

It sets the reshape call back for current window.
glut set cursor ();

It changes the cursor image of current window

10 Explain cohen - sutherland line clipping allgorithm

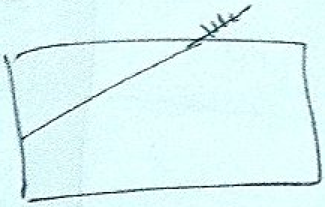
There will be a rectangular window (clipping window)

There will be an object

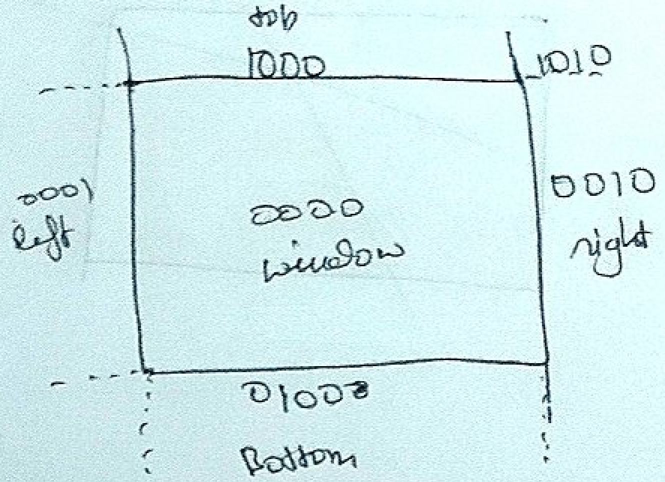
only pencil inside the rectangle must be shown.

pencil outside the rectangle should not be shown

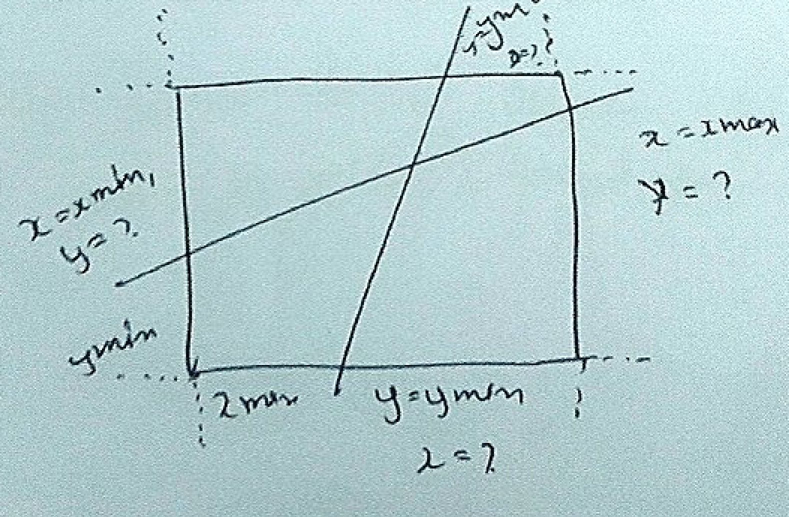
example



Boundaries



Consider

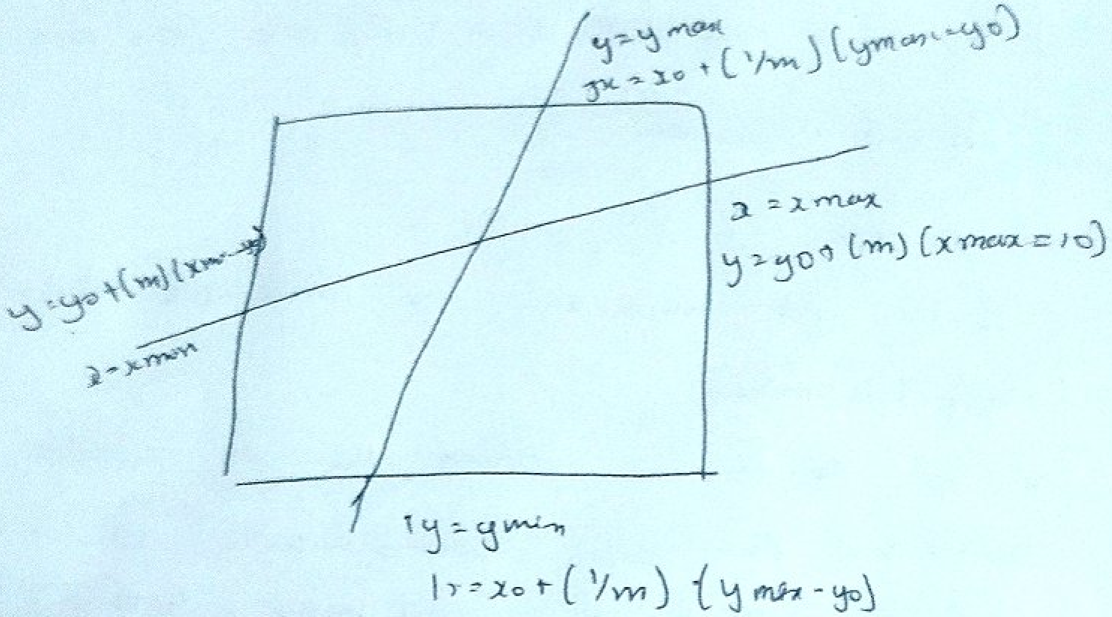


$m = (y - y_0) / (x - x_0)$

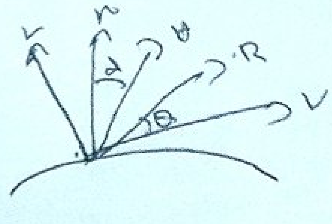
$m \cdot (x - x_0) = (y - y_0)$

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

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



Phong model sets the intensity of specular reflection to $\cos^{n_s}$



n_s = shininess

$I_{specular} = W(\theta) I_l \cos^{n_s} \phi$

I = intensity

0 b/w 0≤1 is called specular reflection coefficient

If light direction L and viewing direction V are on the same side of the normal N or if L is behind the surface, specular effects do not exist

For most opaque materials specular-reflection coefficient is nearly constant ks

$$I_{e,\,specular} = \begin{cases} k_s I_p\, V \cdot R & V \cdot R > 0 \text{ and } N \cdot L > 0 \\ 0.0, & \text{otherwise} \end{cases}$$

R can be calculated from L and N. $R = 2N \cdot L\, N - L$

efficient computation

$H = L + V$

If the light source and the viewer are relatively far from object H is constant

H is the direction yielding maximum specular reflection to viewing direction V if the surface normal N would coincide with H. If V coplanar with R and L (and hence with N too) $\alpha = \phi/2$.