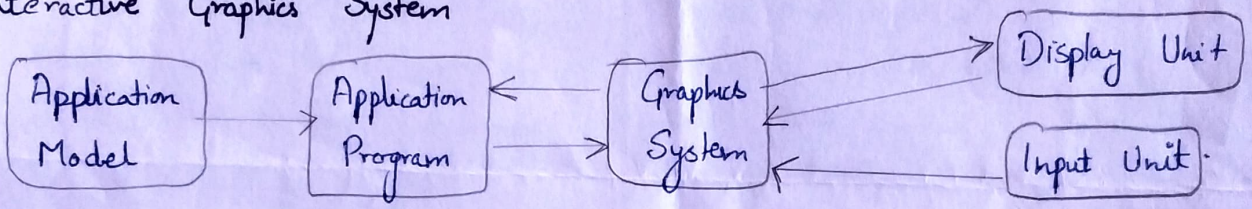
Computer Graphics & Visualization

DIT University - 3rd year 5th semester

Session ⇒ 2021-22

- **Computer Graphics**
  - → Art of drawing pictures or shapes using computers with help of programming.
  - → Made up of pixels
  - → Tasks
    a). Modeling (creating & representing geometry of objects in 3D world)
    b). Rendering (generating 2D images of an object)
    c). Animation (describe how state of object change in time)

- **Interactive Graphics System**



Advantages:-

- High quality graphics display
- Produce animation & control it
- Dynamic updates (prospects of object in view)
- Audio feedback

Applications

1. **Computer Aided Design (CAD)**
   - Design of engineering & architectural systems (Buildings, automobiles, aircrafts, etc.)
2. **Presentations**
   Summarize financial, mathematical & scientific data (economic) → bar, line graphs, pie charts.
3. **Computer Art**
   - Design object shapes & specifying object motion
4. **Education & training**
   - Computer generated models & ~~stim~~ simulators.
5. **Visualization**
   - Represent large amounts of data obtained from scientific, medical, business analysis.
6. **Image Processing**
   - Modify/interpret existing images (improve quality or machine perception of visual info)
7. **Entertainment**
   Animated movies, motion pictures & games.
8. **Medical field**
   Represent parts & processes of human body
9. **GUI**
   Interface of software, communicating with users.

Types of Computer Graphics.

1. Raster (composed of pixels)
2. Random/Vector (composed of paths)

* Raster images → Bitmap images (use a grid of individual pixels where each pixel can be a different color/shade)

* Vector graphics → use mathematical relationships b/w points & paths connecting them to describe an image)

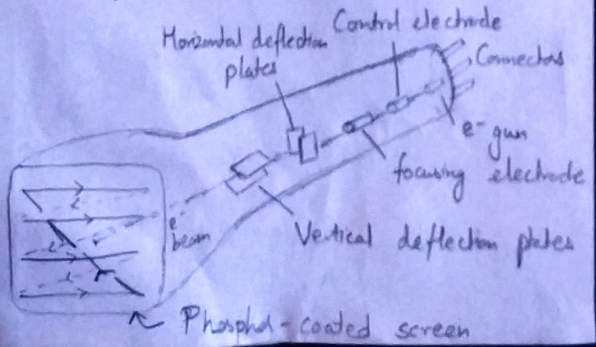
Raster Scan Display

Raster → Rectangular array of points/dots

- An image is sub-divided into a sequence of horizontal strips ("scan lines") which are further divided into discrete pixels for processings.

Working

- The e⁻ beam is swept across the screen, one row at a time from top to bottom
- As e⁻ beam moves across each row, the beam intensity is turned on & off to create a pattern of illuminated spots.
- Return to left of screen, after refreshing each scan line (Horizontal retrace)
- At end of each frame, the e⁻ beam returns to top left corner of screen to begin the next frame (Vertical retrace)



Refresh buffer / Frame buffer: Memory area that holds the set of intensity values for all screen points (picture definition)

These values are retrieved from the buffer & painted on screen, one row (scan line) at a time.

- Quality of raster image
  - → Resolution (no. of pixels)
  - → Color depth (amount of info in each pixel)

- Black & white system
  - → Each screen point is either ON or Off, so only 1 bit/pixel is needed to control intensity of screen positions
  - Such frame buffer is called Bitmap.

- High quality raster graphics system → 24 bits/pixel
- Refresh rate (raster scan display) → 60-80 frames/sec

Applications

- Realistic screen display.
- Home TVs, computers, printers

formats :- BMP (Windows Bitmap), JPEG, GIF, PNG, PSD (Adobe Photoshop)

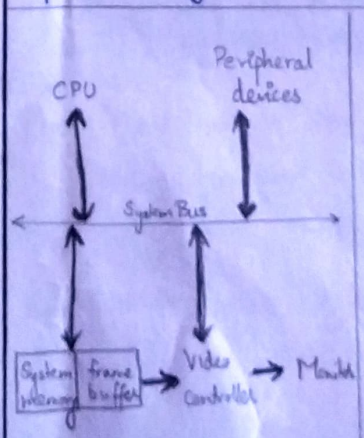
Disadvantages:-

- Pixelation on increasing size (loss of details & clarity)
- Aliasing (jagged staircase-effect plotted as discrete pts.)

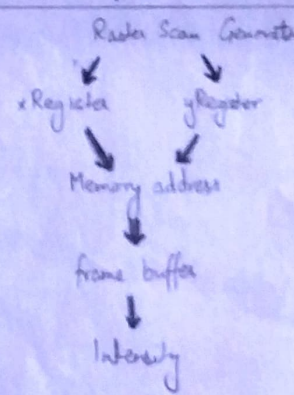
Interlacing / Interleaving

Method of encoding a bitmap image s.t. a person who has partially received it, sees a degraded copy of entire image.

- Used for slower refresh rates; each frame displayed in 2passes using interlaced refresh procedure.



Raster scan architecture



Raster refresh process.

Random Scan Display

- Uses e^- beam which operates like a pencil to create a line image on CRT screen.
- CRT has e^- Beam directed only to parts of screen where picture is to be drawn
- Draw pictures 1 line at a time & thus, is called vector display / stroke-writing / calligraphic display.
- Refresh rate depends on no. of lines to be displayed
- Picture definition (line drawing commands) is stored in refresh display file (display list)
- Designed to draw all component lines of a pic 30-60 per second.
- Produce smooth line drawing
- Can't display realistic shaded scenes

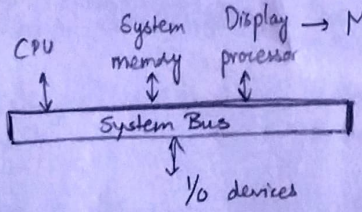
Advantages

- Higher resolution, smooth line drawing.
- Smaller file sizes
- No pixelation
- Parameters can be modified later.

Disadvantages.

- Costlier
- Just does wire frame
- Complex scene cause visible flicker

Random Scan Architecture

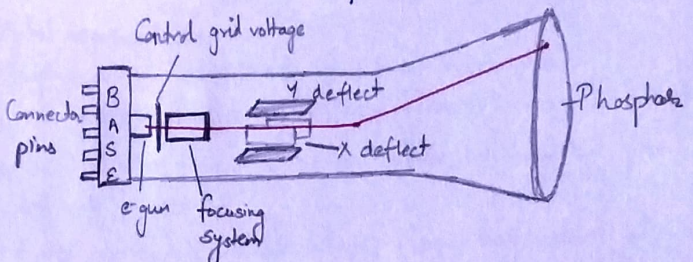


Display devices

VDU → Video/Visual Display Unit
CRT → Cathode Ray Tube
LCD → Liquids Crystal Display
LED → Light Emitting Diode
DVST → Direct View Storage tubes

① CRT

- A vacuum tube that displays images when an e^- beam collides on radiant surface.



e^- gun :- Consists of heating filament (heater) & cathode Source of e^- focused on narrow beam facing CRT

focusing & accelerating anodes :- Produce narrow & sharply focused e^- beam.

Horizontal & vertical deflection plates :- Guide path of e^- beam
Plates produce EM field to deflect beam

Phosphor coated screen :- Produce bright spots when high velocity e^- beam hits it.

② LCD

- Depends on light modulating properties of liquid crystals
- Works on flat panel display technology.
- Uses liquid crystal to turn pixels on or off.
- Liquid crystals → Solid + Liquid. When current flows inside it, its position change into desired color.

Pros :- Produce bright images
Energy efficient
Completely flat screen

Cons:- fixed aspect ratio & resolution
Lower contrast
More expensive

③ LED

- Semiconductor device which emits light when current passes through it.
- Powerful in structure → capable of withstanding mechanical pressure

Pros:- • Control on light intensity
• Low operational voltage
• High temp. handling

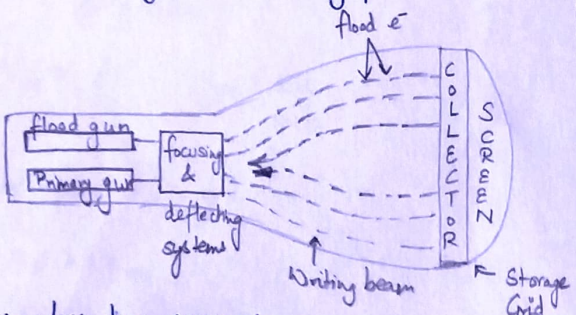
Cons:- • More power consuming

(4) DVST

- Store pic info as a charge distribution behind phosphor-coated screen.
- 2 guns used:-

a). Primary Gun (store pic info)

b). flood/secondary Gun (display pic on screen)



Pros:-
- Less time consuming
- No refreshing required
- High resolution
- Less cost

Cons:-
- Specific part of image can't be erased
- Do not display color

Output primitives

- Each structure is specified with input coordinate data & other info about the way that object is displayed.

Digital representation

- Display screen divided into scan lines & columns.
- Pixel positions are referenced acc. to scan line no & col no.
- Scan lines start from 0 at screen bottom, columns start from 0 at left side of screen
- Pixels referenced with int values; frame buffer stores intensities temporarily; video controller reads from frame buffer & plot the pixels.

Point drawing (Raster System)

- **B/w** :- setting bit value corresponding to specified screen pos within frame buffer to 1.
- **RGB**:- load frame buffer with color codes for intensities that are to be displayed at screen pixel positions.

Line drawing

- Calculate intermediate positions b/w endpoints & directing o/p devices to fill in the calculated positions

Raster Scan	Random Scan
1. Draw image by scanning one row at a time.	Draws image by directing e⁻ beam directly to the desired part of screen.
2. e⁻ beam move all over the screen.	e⁻ beam moves b/w end-points of graphics primitives.
3. Resolution is poor coz of aliasing effect.	Resolution is good
4. Refresh buffer's used in which pic def is stored as set of intensity values.	Display file is used in which pic def is stored as a set of line drawing instructions.
5. Pixels are used to draw images	Mathematical funct" are used to draw images.
6. Designed for realistic display containing shadows & shades.	Designed for line/character drawings.
7. Less expensive	More expensive
8. Refresh rate 60-80 frames/sec	Refresh rate: 30-60 frames/sec.
9. Used in monitors/TVs	Used in CRO, pen plotter
10. Entire screen is refreshed	Only selected portions are redrawn

- Beam penetration technique (Random scan)
- Shadow mask technique (Raster scan)

Generation of characters

- Stroke method
- Vector/Bitmap method
- Star burst method.

> Samples line at unit intervals for one coordinate & determine corresponding int values nearest line path for other coordinate

DDA Algorithm

- Digital Differential Analyzer
- Incremental method of scan conversion of line
- Calculation is performed at each step using results of previous steps.

Pros

- Simple, easy to implement
- faster
- Doesn't use multiplication theorem
- Detect change in values of x & y
- Gives overflow indication when pt. is repositioned.

Cons

- Overhead of round off()
- ↑ time complexity
- Suitable for software but not hardware implementation
- Resulted lines aren't smooth.

DDA

1. Enter x_1, x_2, y_1, y_2
2. $dx = x_2 - x_1$; $dy = y_2 - y_1$
3. If $dy > dx$
   $step = dy$
   else
   $step = dx$
4. $x_{inc} = \frac{dx}{step}$ $\quad y_{inc} = \frac{dy}{step}$
5. Plot pixel (x, y)
6. $x = x + x_{inc}$
   $y = y + y_{inc}$
7. Repeat till $x = x_2$

Bresenham Line Algorithm

- Used for scan converting a line
- Efficient method → only involves integer (addition, subtraction & multiply ops).
- Quick line generation.

Pros

- Simple (only integer arithmetic)
- Avoids duplicate pt. generation.
- Implemented using hardware
- Faster than DDA

Cons

- Only for basic line drawing
- Though better, but still not smooth
- Can't handle diminishing jaggies.

Algorithm.

1. Enter x_1, x_2, y_1, y_2
2. Calculate dx & dy

3a). If $dy > dx$ $\quad P_0 = (2dx - dy)$ $\quad$ always ↑ y.

$P_k \geq 0$ $\quad [x_{i+1}, y_{i+1}]$ $\quad P_{k+1} = P_k + 2(dx - dy)$

$P_k < 0$ $\quad [x_{i+1}, y_{i+1}]$ $\quad P_{k+1} = P_k + 2dx$

3b). If $dx > dy$

$P_0 = 2dy - dx$ $\quad$ (always ↑ x)

$P_k \geq 0$ $\quad [x_{i+1}, y_{i+1}]$ $\quad P_{k+1} = P_k + 2(dy - dx)$

$P_k < 0$ $\quad [x_{i+1}, y_i]$ $\quad P_{k+1} = P_k + 2dy$.

4). Repeat dx/dy times.

Mid-point Circle Algorithm

- Given radius & center coordinates, it attempts to generate the points of one octant.
- Points for other octants are generated by eight-symmetry property.

Pros:-

- Powerful & efficient
- Based on circle eqn
- Easy to implement
- Generate curves on raster displays.

Cons:-

- Accuracy of generating pts. ↓
- Circle is not smooth
- Time consuming.

Algorithm.

1. Enter x_c, y_c, r
2. Initial pt. $(x_0, y_0) = (0, r)$
3. $P_0 = 1 - r$
4. If $P_k < 0$ $\quad (x_{k+1}, y_k)$

   $P_{k+1} = P_k + 2x_{k+1}$

   If $P_k > 0$ $\quad (x_{k+1}, y_k - 1)$

   $P_{k+1} = P_k + 2(x_k - y_k) + 1$.
5. If $(x_c, y_c) \neq (0, 0)$

   $(x_{plot}, y_{plot}) = (x_k + x_c, y_k, y_c)$
6. Repeat until $y_k \geq x_k$
7. Use 8-symmetry property for all coordinates.

Bresenham Circle Algorithm

- Same process as mid-point circle algo

Pros:-
- Easy to implement
- Based on eqn of circle

Cons:-
- Accuracy of pts ↓
- Suffers when used to generate complex & high graphical images
- No significant enhancement w.r.t. performance

Algorithm

1. Enter x_c, y_c, r
2. Starting pt. $(x_0, y_0) = (0, r)$
3. $P_0 = 3 - 2r$
4. If $P_k < 0$ (x_{k+1}, y_k)
   $P_{k+1} = P_k + 4x_k + 6$
   If $P_k \geq 0$ (x_k+1, y_k-1)
   $P_{k+1} = P_k + 4(x_k - y_k) + 10$
5. If $(x_c, y_c) \neq (0,0)$
   $(x_{plot}, y_{plot}) = (x_k + x_c, y_k + y_c)$
6. Repeat until $y_k \geq x_k$
7. Use 8-way symmetry for all points.

Midpoint Ellipse Algorithm

- Mid-point b/w 2 pixels is calculated which helps in calculating decision parameter.
- The value of decision parameter decides whether mid-point lies inside, outside or on ellipse boundary & then position of mid-point helps drawing ellipse

Pros:- • Simple, easy implementation (only addition)

Cons:- • Time consuming.

Algorithm:-

1. Input x_c, y_c, r_x, r_y.
2. $dx = 2r_y^2 x$; $dy = 2r_x^2 y$
3. Start point $(x_0, y_0) = (0, r_y)$
4. Region 1

$$P_0 = r_y^2 - r_x^2 r_y + \frac{1}{4} r_x^2$$

If $P_k < 0 \rightarrow (x_{k+1}, y_k)$

$$P_{k+1} = P_k + dx + r_y^2$$

Update dx ; $dy = dy$

$P_k \geq 0 \rightarrow (x_{k+1}, y_k - 1)$

$$P_{k+1} = P_k + dx - dy + r_y^2$$

Update dx, dy

Until $dx < dy$

5. Region 2

$$P_0 = r_y^2 \left(\frac{x+1}{2}\right)^2 + r_x^2 (y-1)^2 - r_x^2 r_y^2$$

If $P_k > 0 \rightarrow (x_k, y_k - 1)$

$$P_{k+1} = P_k - dy + r_x^2$$

Update dy ; $dx = dx$

$P_k \leq 0 \rightarrow (x_{k+1}, y_k - 1)$

$$P_{k+1} = P_k + dx - dy + r_x^2$$

Update dx, dy

Until $dx \geq dy$

6. Plot symmetry points in other 3 quadrants.
7. If $(x_c, y_c) \neq (0,0)$

$$(x_{plot}, y_{plot}) = (x_k + x_c, y_k + y_c)$$

1. Scaling $\begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$

2. Rotation (clockwise) $\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$

3. Rotation (counter-clockwise) $\begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$

4. Translation $\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$

5. Reflection (about x-axis) $\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

6. Reflection (about y-axis) $\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

7. Reflection (about origin) $\begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

8. Reflection (about $y=x$) $\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

9. Reflection (about $y=-x$) $\begin{bmatrix} 0 & -1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

10. Shearing (Y direction) $\begin{bmatrix} 1 & 0 & 0 \\ sh_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

11. Shearing (X direction) $\begin{bmatrix} 1 & sh_x & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

12. Shearing (X & Y both) $\begin{bmatrix} 1 & sh_x & 0 \\ sh_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

DDA	Bresenham
1. Less efficient	More efficient
2. Less calculating speed	More calculating speed
3. Costlier	Cheaper
4. Less precision/accuracy	More accurate
5. Complex calculations	Rather simple calculations
6. Optimization not provided	Optimization provided

Mid-point	Bresenham
1. Not optimized	Optimized form of mid-point circle
2. Prevent trigonometric calculations & only use adopting integers	Deals with integers only
3. More time consuming	Less time consuming
4. Needs floating point calculations	Prevents calculations of floating point
5. Less efficient & less accurate	Though accuracy is not much greater, it's better than mid-point circle

Homogeneous coordinates

→ Many graphics application involve sequences of geometric transformations. In order to combine sequence of transformations, we have to eliminate matrix addition

→ To achieve this, we add an additional dummy coordinate h, matrix becomes $(n+1, n+1)$

→ Allows to express transformation equation.

Representation (x, y, h)

Actual Coordinates $(x/h, y/h)$

X shear with y reference line

$$T = \begin{bmatrix} 1 & sh_x & -sh_x \cdot y_{ref} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Y shear with x reference line

$$T = \begin{bmatrix} 1 & 0 & 0 \\ sh_y & 1 & -sh_y \cdot x_{ref} \\ 0 & 0 & 1 \end{bmatrix}$$

Rotation abt arbitrary point

$$T = \begin{bmatrix} \cos\theta & -\sin\theta & x_r(1-\cos\theta) + y_r\sin\theta \\ \sin\theta & \cos\theta & y_r(1-\cos\theta) - x_r\sin\theta \\ 0 & 0 & 1 \end{bmatrix}$$

3D Transformations

① Translation

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

② Scaling

$$\begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

③ Rotation

i) about x-axis

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

ii) about y-axis

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

iii) about z-axis

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

④ Shearing

i) about X-axis

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ sh_y & 1 & 0 & 0 \\ sh_z & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

ii) about y-axis

$$\begin{bmatrix} 1 & sh_x & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & sh_z & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

iii) about z-axis

$$\begin{bmatrix} 1 & 0 & sh_x & 0 \\ 0 & 1 & sh_y & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

⑤ Reflection

ii) Relative to XY plane

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

ii) Relative to YZ plane

$$\begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

iii) Relative to ZX plane

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

* Window to Viewport Transformation.

$$X_v = X_{v_{min}} + (X_w - X_{w_{min}})\, S_x$$

$$S_x = \frac{\text{Width}(v)}{\text{Width}(w)} = \frac{X_{v_{max}} - X_{v_{min}}}{X_{w_{max}} - X_{w_{min}}}$$

$$Y_v = Y_{v_{min}} + (Y_w - Y_{w_{min}})\, S_y$$

$$S_y = \frac{\text{Height}(v)}{\text{Height}(w)}$$

* Point Clipping

→ $X_{w_{min}} \leq X \leq X_{w_{max}}$

→ $Y_{w_{min}} \leq Y \leq Y_{w_{max}}$

* Cohen - Sutherland Line Clipping

① Assign 4-bit code to each end point of line segment

② if (both codes == 0000)
line completely visible
else
do bitwise ANDING
if (result == 0000)
partially visible
↳ perform clipping

③ Clipping
if (line cuts $X_{w_{min}}$ / $X_{w_{max}}$)

$$Y = Y_1 + m(X - X_1)$$

else

$$X = X_1 + \frac{(Y - Y_1)}{m}$$

* Sutherland Hodgeman Polygon Clipping

① Completely inside → Preserved

② Inside - to - outside → Intersection point

③ Completely outside → No output

④ Outside to - inside → Intersection point + next vertex.

Projection

→ Mapping / transformation of 3D object into view / projection plane

Types -

Perspective	Parallel
· Realistic looking	· Used in architecture drawings

Description

· View plane
· Center of projection
· Projectors.

Perspective projection

· Vanishing Point (all line converge to meet)
· Perspective foreshortening. (create illusion of object receding strongly into the distance)

Parallel projection

· View plane · Viewing direction · Projectors.

i). Isometric :- All projectors make equal angles (usually 30°)

ii). Dimetric :- Two projectors have equal angle wrt principal axes.

iii). Trimetric :- Direction of projection makes unequal angle with principal axis

iv). Cavalier :- All lines ⊥ to projection plane are projected (45°) with no change in length

v). Cabinet :- All lines ⊥ to Projection plane are projected (63.4°) one half of their length. Realistic appearance

Orthographic
- Projectors are ⊥ to view plane

Oblique
- DOP not ⊥ to view plane

Scan conversion
→ Process of representing continuous graphics objects as a collection of discrete points.

Inside-outside test
- Even-odd test
- Determine whether a point is inside polygon or not
- Construct line segment from point p to known distant outside point of polygon.
- Count no. of intersection of line segment with polygon boundary
- If odd no → point p is inside
  otherwise → outside.

Winding Number Method
- Determine whether a point is inside polygon or not
- Method counts no. of times polygon edges wind around point P in anti-clockwise direction
- Thus count is called winding no., interior points ⇒ have non-zero value from winding no.
- Draw line from P to exterior point at infinity without passing through any vertex
- Count no. of edges crossing line in each direction.
- Right to left +1; left to right -1.