Z-buffer

Adv

1. Easy to implement
2. No sorting object.
3. Unlimited scene complexity.
4. Need buffer.

Disadv

1. Requires lots of memory.
2. Requires recalculation.
3. Waste time drawing hidden objects.

Painters Algorithm → (Depth sort)

Object space algorithm

Draw surface from back to front:

→ Sort surface by depth.

→ Draw objects in order

→ Comparing overlapping surfaces

→ It repeats for all surface.

→ However, the overlapping is observed, there need further tests.

Frame, ~~Image buffer(x,y)~~
Z buffer(x,y)

Z buffering

① Initially each pixel of the Z-buffer is set to the maximum depth value.

② Image buffer is set to background color.

③ Surface are rendered one at a time.

④ For first surface, depth value each pixel is calculated.

⑤ If the depth value smaller than the corresponding depth value. Then replace the depth value and the color value calculated at the pixel position.

⑥ Repeat 4 and 5 for remaining surface.

⑦ After all surface have been processed, each image buffer represents color of visible surface at the pixel.

Hidden Chp-15

Visible surface → It refers to the portion of an object that is visible to the viewer from a particular viewpoint.

→ (efficient, realistic, clean and interactive visualization while allowing designer to control over composition and focus.)

Visible surface

Object space method	Image space method
(1) Operates 3D object entities.	(1) Operates on 2D image.
(2) Compare object each other within scene definition.	(2) Visibility decided pixel position on the projection plane.
(3) Object	(3) Pixel

3D Shearing

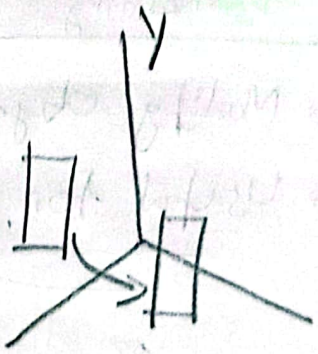
→ Modify object shape.

→ Useful for perspective projection.

$$M_{3Shear} = \begin{bmatrix} 1 & 0 & Sh_{zx} & -Sh_{zx} Z_{ref} \\ 0 & 1 & Sh_{zy} & -Sh_{zy} Z_{ref} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Y-axis

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



$$P' = R_y(\theta)\, P$$

3D scaling

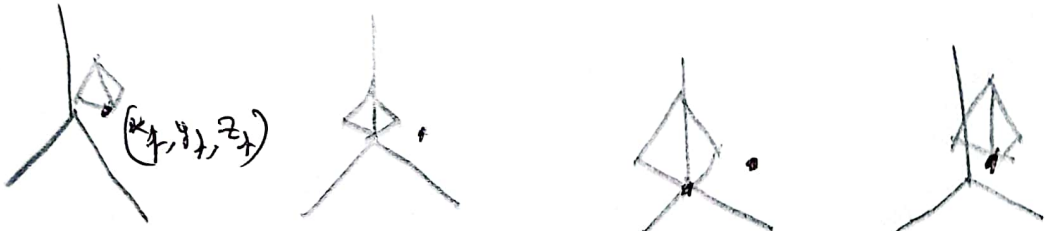
Changes the size of the object and reposition the object.

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$P' = S \cdot P$$

Fixed point:

$$T(x_f, y_f, z_f) \cdot S(S_x, S_y, S_z) \cdot T(-x_f, -y_f, -z_f) = \begin{bmatrix} S_x & 0 & 0 & (1-S_x)x_f \\ 0 & S_y & 0 & (1-S_y)y_f \\ 0 & 0 & S_z & (1-S_z)z_f \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



original position translate scaling inverse translate

(Ch 9)

Homogenous Coordinates

1. It is easier to compose translation and rotation.
2. Matrix multiplication.
3. 4×4 matrix.

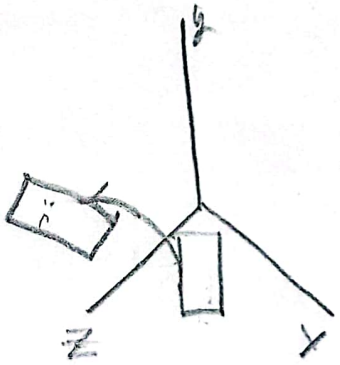
3D Rotation

Specified by a rotation axis and angle.

Positive rotation angles is counter clockwise.

Z-axis

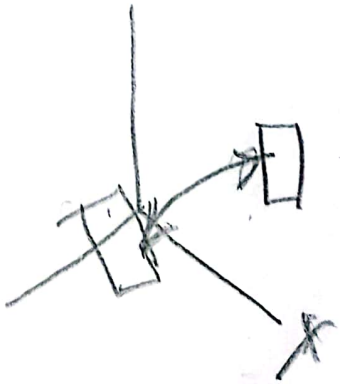
$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$



$$P' = R_z(\theta)\, P$$

X-axix

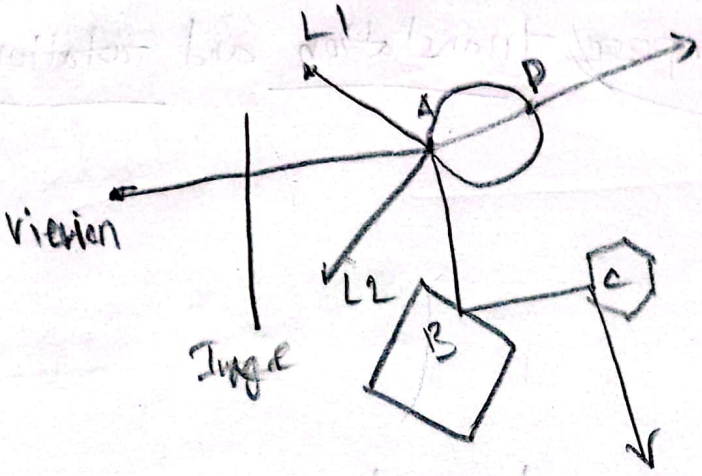
$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$



$$P' = R_x(\theta)\, P$$

Recursive Ray tracing

Consider specular reflection.

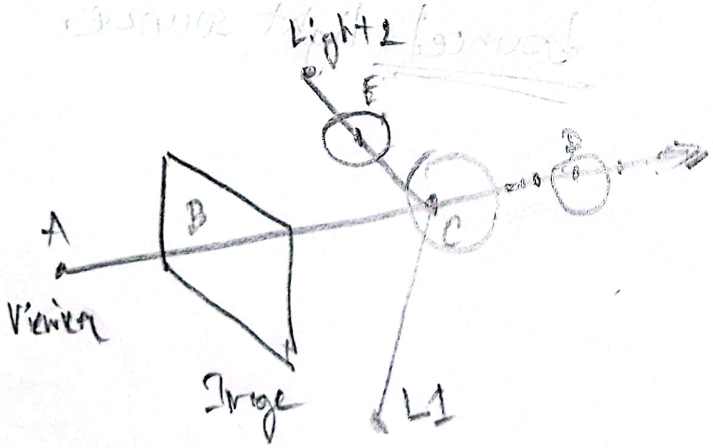


Illumination: An observable property and effect of light.

Specular	Diffuse
① Reflection from a polished surface.	② Reflection from a rough surface.
② Parallel rays remain after reflection	② Parallel rays do not remain parallel after reflection.

Ray tracing

Behaviour of the light as it interacts with the objects in 3D scene by tracing the path of light rays from a virtual camera.



Chapter-12 Illumination

Local → Objects are only illuminated directly by the light source.

Global → objects are do not only reflect light to the viewer but also other objects.

Local	Global
① Calculation simple, point based.	① complex, scene-wide.
② Lighting accuracy Limited	② High
③ Shadows hard edged and unrealistic.	③ Soft edged and realistic shadow.
④ Only considered direct light source.	④ considered indirected and bounced light source.

NDCS → a unit square (1 × 1) defines the display area of a virtual display device.

WCS → Objects are placed into a scene by modeling transformation to a master coordinate system.

VCS → Introduced to simulate the effect of moving the camera.

Viewing transformation → Two coordinate mapping operations.

Clipping → the portions of a picture that are either inside or outside of a specified region of space.

(b) If line pass through the bottom

$$x' = x + (y_{min} - y)/m$$

$$y = y_{min}$$

(c) If line pass through the left

$$y' = y + (x_{min} - x) * m$$

$$x = x_{min}$$

(d) If line pass through the right

$$y' = y + (x_{max} - x) * m$$

$$x' = x_{max}$$

(6) Overwrite end points with new one and update.

(7) Repeat step 4 till the line get completely clipped

Algo

① Assign region code.

② Perform OR operation on endpoint.

③ If the result of OR is 0000

then line is completely visible.

else perform AND operation on the end points.

(i) If result of AND is not 0000

the line invisible and not inside the window.

not considered for clipping.

(ii) else the line is a clipping candidate.

④ After confirming the ~~clipping candidate~~ line partially inside the window. we will find the intersection.

(i) If bit 1 is 1, intercept with the line $y = y_{max}$

(ii) " " 2 " 1, " " " $y = y_{min}$

(iii) " " 3 " 1, " " " $x = x_{max}$

(iv) " " 4 " 1, " " " $x = x_{min}$

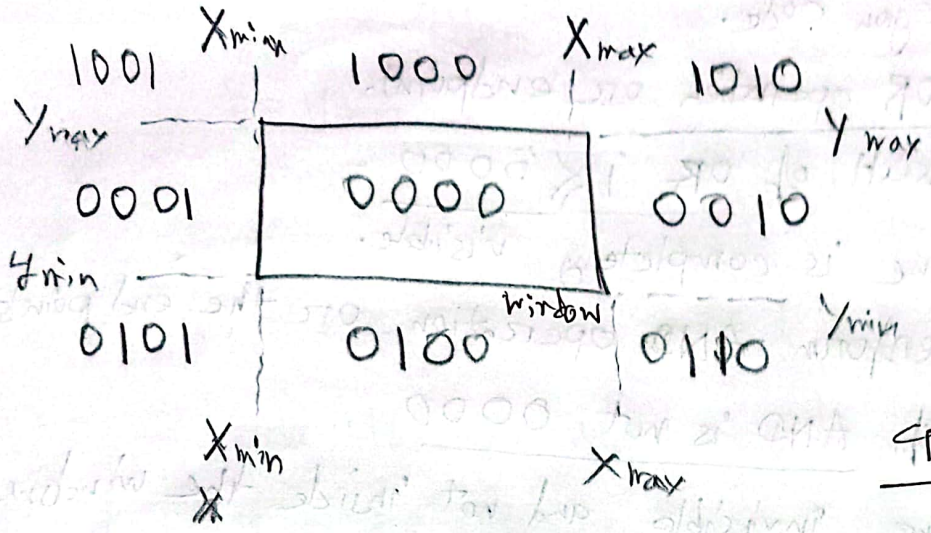
⑤ (a) If line pass through top line intercepts with the top boundary,

$$x' = x + (y_{max} - y)/m$$

$$y' = y_{max}$$

Cohen Sutherland

TBRL



4 bit code

If,

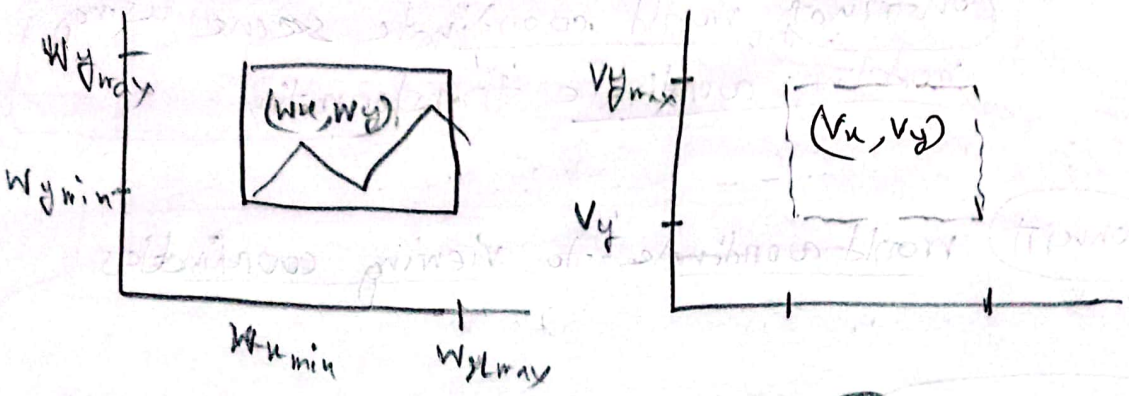
$x < x_{min}$ → left of window (বাইরে থাকে নাকি / কোন side এ হবে)

$x > x_{max}$ → right " "

$y < y_{min}$ → bottom " "

$y > y_{max}$ → top " "

Window-to-viewport coordinate transformation



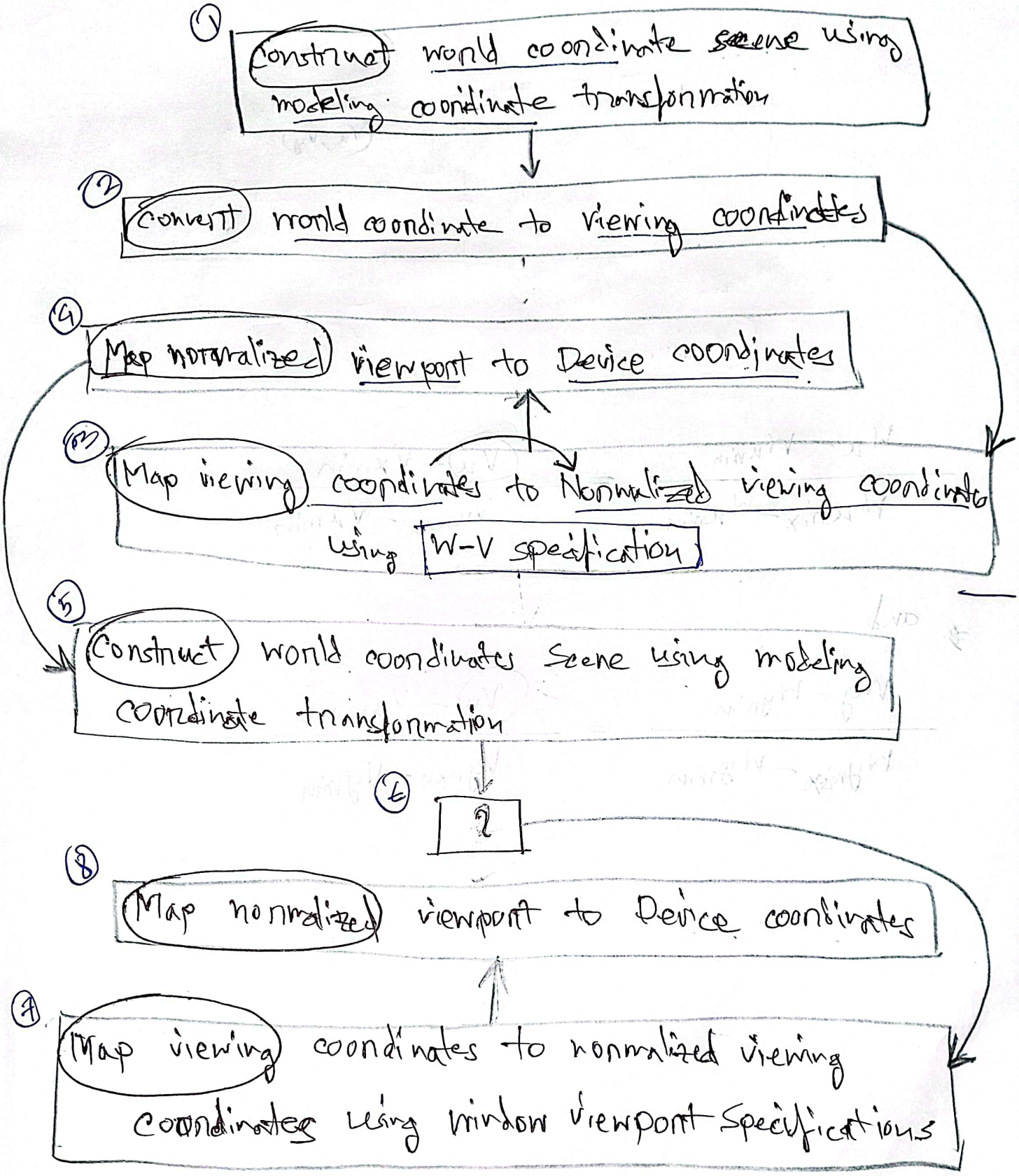
(W) window (V) viewport

$$\frac{W_x - W_{x_{min}}}{W_{x_{max}} - W_{x_{min}}} = \frac{V_x - V_{x_{min}}}{V_{x_{max}} - V_{x_{min}}}$$

* and

$$\frac{W_y - W_{y_{min}}}{W_{y_{max}} - W_{y_{min}}} = \frac{V_y - V_{y_{min}}}{V_{y_{max}} - V_{y_{min}}}$$

Two dimensional viewing transformation pipeline:

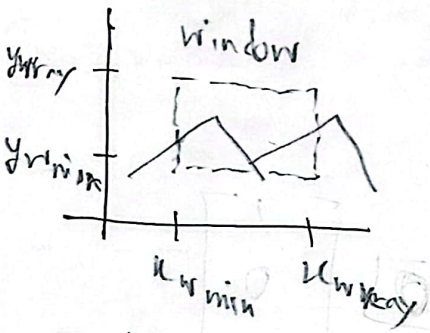


Chapter – 11

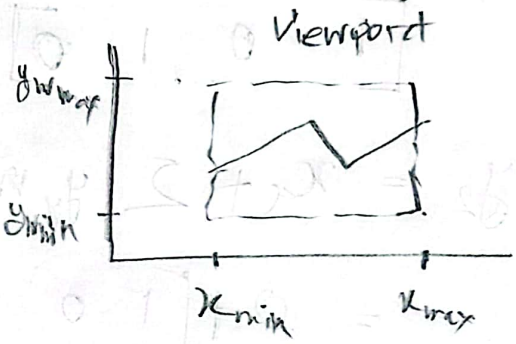
World coordinate → Coordinate system.

Device coordinates → Device এর coordinate system,

Viewing Pipeline → (viewing transformation in several steps)



World coordinates



Device

Window → what is to be viewed.

Device → where is to be displayed.

① World coordinate area selected for display called window.

② Viewport Display device to which a window is mapped is called viewport.

Perspective

Object

① Dimension are not same

② Not perpendicular Projection.

③ COP is finite.

Parallel

① Dimension same.

② Perpendicular projection.

③ DOP is infinite.

Orthographic → parallel projection where the projecting lines emerge parallely.

Orthographic

Adv

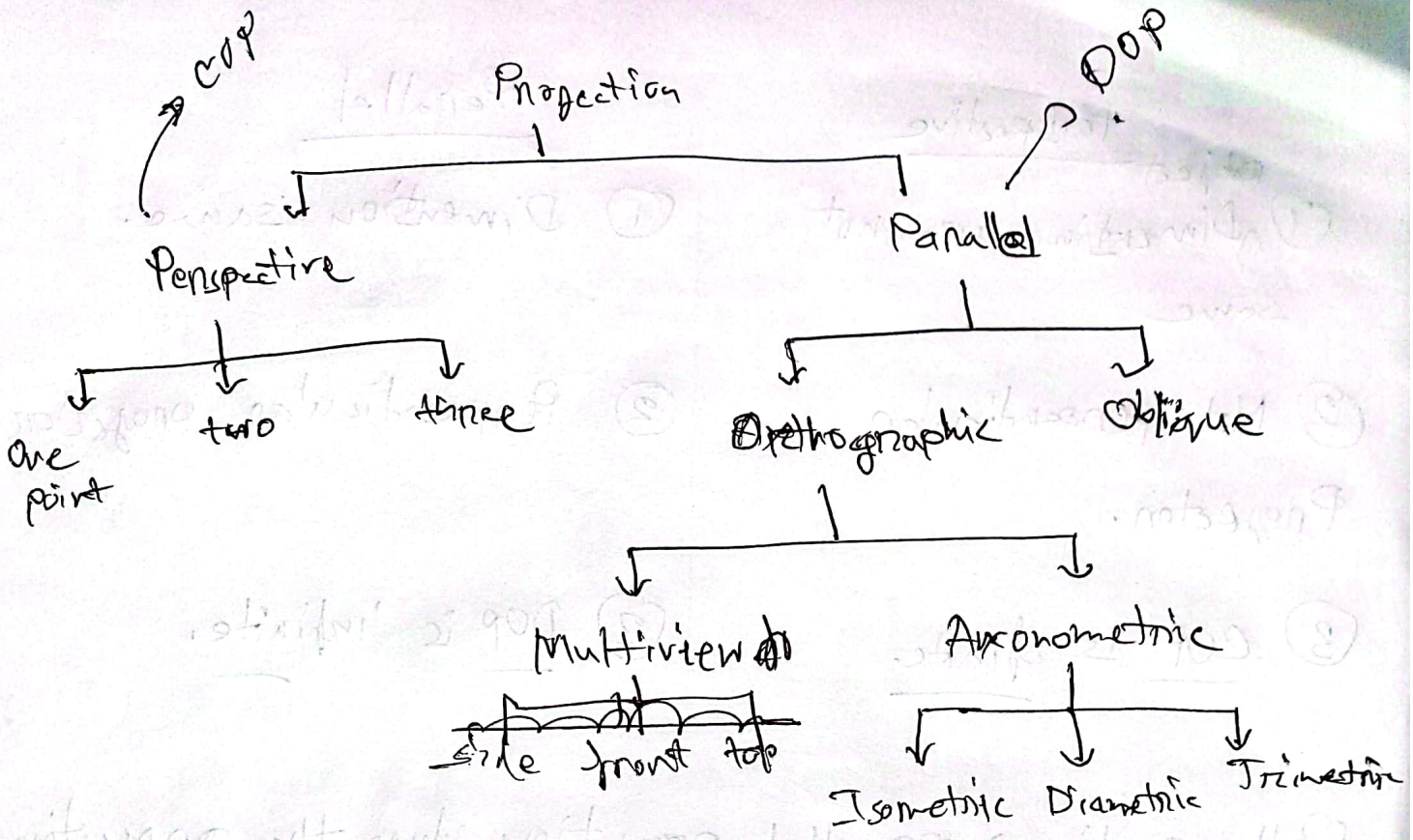
① Shapes preserved.

② Used for measurement

③ Perpendicular projection.

Disadv

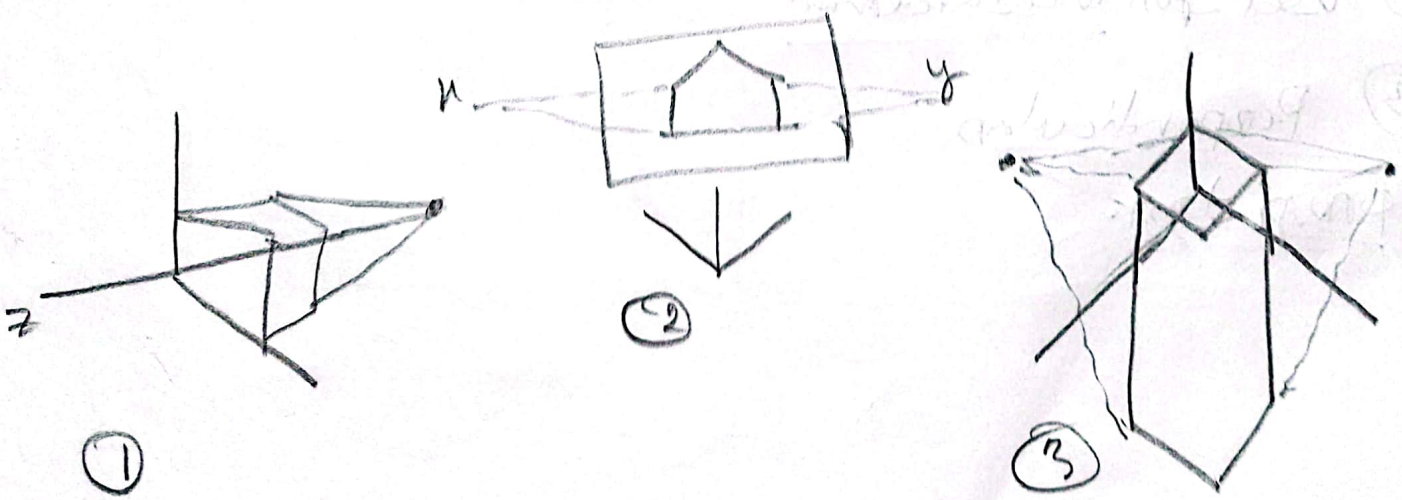
① Surface hidden from the view.



Projection → Defines a mapping of the point onto its image in the projection plane.

Perspective foreshortening → converge at COP.

Vanishing point: converge at a single point in the projection



Dimetric's Visualized object appears to have only 2 adjacent sides and angles are equal.

Trimetric's All the adjacent sides and angle unequal.

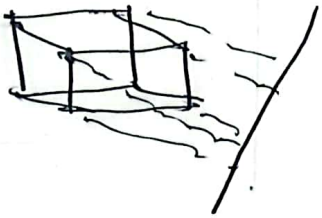
Adv/Dis

→ Lines are scaled.
→ || preserved but angles are not.
→

Oblique

Arbitrary relationship between projectors and projection plane.

Front view: Rays emerge front view of polygon surface observed.



Axonometric

→ Orthographic Projection.
→ Perpendicular projection line.
→ Rotated around one or more axis.

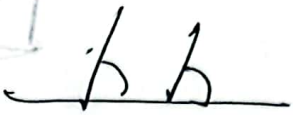
Isometric : Visually representing 3 dimensional objects in two dimensional display.

Orthographic

Projection is perpendicular to the projection plane.

Oblique

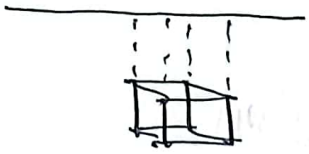
Projection is not perpendicular to the projection plane.



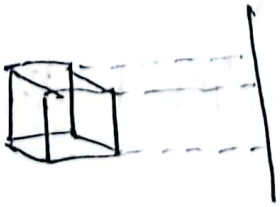
Multiview

Projection face parallel to the principle face.

Top view: Rays emerge from top of the polygon surface are observed.

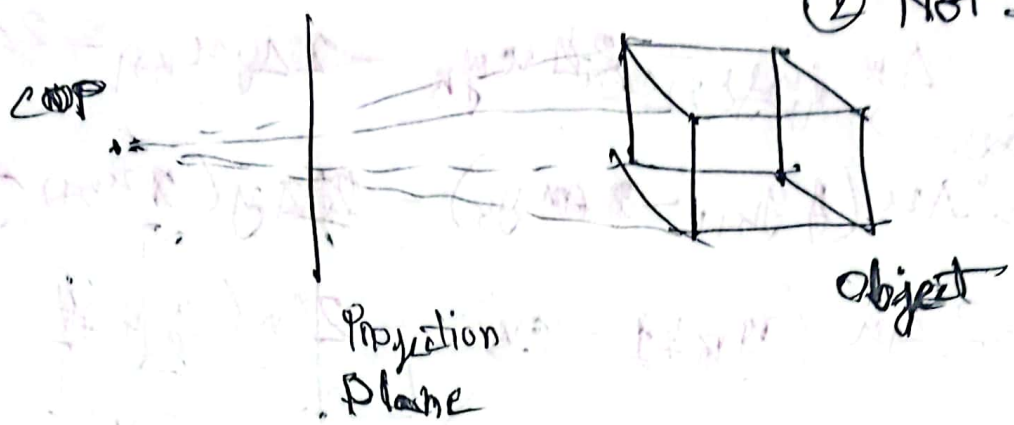


Side view: Rays emerge from side view of the polygon surface are observed.



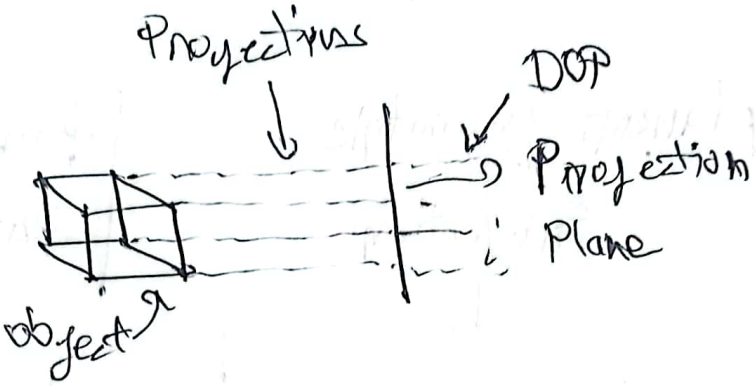
Perspective

① COP
② Not same ↓ dimensio



COP is located at a finite point in 3D space.

Parallel



The COP is located at Infinity and all projection line are parallel.

① DOP same 2 ,
②