UNIT 4

PART A

modeling transform-ations	**Viewing transformations**
world coordinate system.	re-position the scene in front of a camera
view orient-ation transf-ormation	DC to WC; camera at origin looking along -z axis
parallel projection	**perspective projection**
preserves true measur-ements; less realistic	realistic (depth); does not preserve true measur-ements

Center of Projection (COP) or **Projection Reference Point** (PRP) is the single point where all projectors converge

Viewport Clipping - after projection in standard viewing pipeline (before the perspective projection division; avoids division by zero)

Back-Face Detection condition	V = N • D
Scan-Line Method	determines the closest object along the line of sight (depth) and its corres-ponding color

UNIT 4 (cont)

Painter's Algorithm. - Depth--Sorting Method	**Primary criterion:** sorted by minimum z-depth Ambiguity tests: Bounding rectangle overlap test, depth overlap test, surface inters-ection test, and final "complete behind" test.

composite transformation matrix is not commutative
limitation of Back-Face Detection : multiple front faces overlap or obscure each other (hard to detect)

Parallel vs. Perspective projection

Parallel projection	**Perspective projection**
Orthographic: perpendicular rays Oblique: angled rays	converge at a point (foreshortening)
No depth cues, looks artificial, parallel lines stay parallel	Realistic, depth visible, lines converge (vanishing points)
Technical Drawings	Realistic Visual-ization

Parallel → accuracy & measurement
Perspective → realism & depth

Back-Face Detection algorithm

Removes polygons whose normal faces away from viewer

N·V≤0⇒discard

Sufficient for:	Single, convex, closed objects (e.g., cube)

Limitations

Back-Face Detection algorithm (cont)

1. Concave Objects:	May remove faces that are actually visible
2. Multiple Objects:	Cannot decide visibility between different objects; No depth comparison
3. Transparent Surfaces:	Incorrectly removes visible back faces
4. Open Surfaces:	Back-face concept may not apply

5. Does not resolve cyclic overlap cases

6. Object-space only; doesn't compare between objects	does not work for scene-level visibility

It is a pre-processing culling step that reduces the polygon count sent to the rendering pipeline (by ~50%).
followed by a true **Hidden Surface Removal** (HSR) algorithm (e.g., Depth-Buffer/Z--Buffer, Painter's, BSP Trees)

3D clipping

Steps

1. Transform vertices → Normalized Device Coordinates (NDC) using projection

2. Clip against 6 planes of canonical cube:	x=±1, y=±1, z=0/1

3. Test vertices (inside/outside)

4. Compute intersection points if edges cross planes

5. Discard outside parts, pass clipped primitives for viewport mapping

Standardized Cube

Uniform planes (inside/out and intersection math faster); hardware optimized; simplified depth (0 to 1)

Clipping in normalized space: consistency & simplicity (axis-aligned with plane equations than skewed, perspective dependent frustum); device independence; efficiency

3D projection

Perspective Projection	
1. find parameter t	$t=z_p-z_{CoP}/z-z_{CoP}$
2. projected x' and y'	$x'=x_{CoP}+t(x-x_{CoP})$ $y'=y_{CoP}+t(y-y_{CoP})$
3. projected z'	$z'=z_p$
Orthographic Projection	
1. Keep x and y unchanged	$x'=x$, $y'=y$
2. Set z to projection plane	$z'=z_p$

Multi-Viewport Viewing Pipeline for CAD

1. Divergence of Single Modeling Stream	One modeling transformation pipeline (world/model matrix) feeds all viewports.
2. Viewing & Projection Management	Each viewport has its own: **View Matrix** (camera orientation) **Projection Matrix** Orthographic (Top, Front, Side) Perspective (3D view)
3. Clipping per Viewport	Clipping done independently for each viewport (frustum)

4. Overall architecture

Model Transform → Stream Duplication → Parallel Per-Viewport (View+Proj Transform → Clipping → Viewport Transform).

Multi-Viewport Viewing Pipeline for CAD

1. Divergence of Single Modeling Stream	One modeling transformation pipeline (world/model matrix) feeds all viewports.
2. Viewing & Projection Management	Each viewport has its own: **View Matrix** (camera orientation) **Projection Matrix** Orthographic (Top, Front, Side) Perspective (3D view)