UNIT 2

PART A

homogeneous coordinates - represent all transformations as a uniform matrix multiplication; easy composition

viewing pipeline - mapping world coordinate system to viewport (display device)

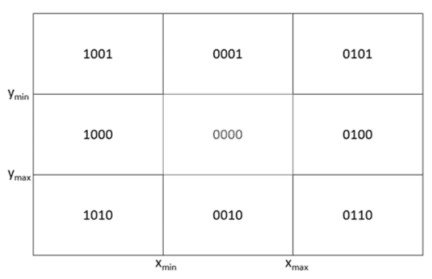
modeling coordinates are local to the object and **world coordinates** are the common reference system

normalization transformation - maps the world coordinate window to a standard, device-independent coordinate system (simplify **clipping** and **viewport mapping**)

Sutherland-Hodgman:
P1 inside, P2 inside → Output P2
P1 inside, P2 outside → Output Intersection
P1 outside, P2 inside → Output I, then P2
P1 outside, P2 outside → Output nothing

Line clipping algorithms: Cohen-Sutherland, Liang-Barsky (or Nicholl–Lee–Nicholl (NLN), Cyrus-Beck

fig. 1



Cohen Sutherland

Transformations

$$Translation \quad \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \qquad Scaling \quad \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$Rotation \quad \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad Shearing \quad \begin{bmatrix} 1 & sh_x & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Homogeneous Coordinates

Extension of 2D Cartesian coordinates (x, y) to 3D (x, y, w).

Point represented as a column vector [x, y, 1]^T.

Without homogeneous coordinates, translation is vector addition (T + v), while others are matrix multiplications (M * v).

1. Unified Representation: 3x3 matrix multiplication

2. Efficient Concatenation: multiple transformations

3. Elegant Handling of Translation: translation requires addition

4. Ability to Represent Points at Infinity: advanced graphical concepts

5. Simplified Inverse Transform: inverse of the single composite matrix,

Trade-off with Cartesian system:
1. Conceptual Complexity - understanding an extended coordinate system
2. Computational Complexity - slight increase (3x3 instead of 2x2) per operation; though there is a computational gain through matrix concatenation (composite matrix)
⇒ composite matrix transformations happen right to left

Polygon tables

Vertex + Polygon Tables	Single Integrated Polygon Table	Three--Table Model
Vertex table (8 coords) + Polygon table (6 faces)	Each face stores full vertex coordinates (6 faces, 3 coordinates)	Vertex table + Edge table + Polygon table
24 floats + 24 ints	72 floats	24 floats + 48 ints
efficient, no duplication	simple	full topology, adjacency queries

Polygon tables (cont)

no explicit edge info	high memory	more memory + complex

Data Efficiency

best	worst	moderate

Integrated = simple but wasteful | Three--table = powerful but heavy | Vertex+Polygon = best balance

2D viewing pipeline process

1. World Window Definition	a rectangular region in the World Coordinate System (clipping boundary)
2. Viewport Definition	a rectangular region or the Device Coordinate System
3. Window-to-Viewport Mapping Process	linear transformation (mapping) from window to viewport

Steps

1. Translate window → origin
2. Scale to viewport size
3. Translate to viewport position

(Additionally) $P_{screen}=M_{screen} \times M_{viewport} \times M_{zoom} \times M_{pan} \times P_{world}$	**Pan:** M_{pan_i}=Translate by ($-T_{xi}$, $-T_{yi}$) **Zoom:** M_zoom_i = Scale by (S_i, S_i)

Flexibility: Separating pan & zoom matrices allows independent control.

Efficiency: GPU-friendly single matrix multiplication

Combined Importance
1. Separation of Concerns (decouples scene design from display specs)
2. Flexibility & Reuse
3. Device Independence

By **racheleva**
cheatography.com/racheleva/

Not published yet.
Last updated 27th April, 2026.

Sutherland–Hodgman

Inside test	$x_{min} \leq x \leq x_{max}$, $y_{min} \leq y \leq y_{max}$

computationally cheap

Inters-ection Checks for Lines	point where a polygon edge crosses a clip boundary.

Use line equations to calculate intersection coordinates

intersection becomes a new vertex; closed and correctly shaped

Iterative Vertex Generation	processes the polygon against each clip boundary (left, right, bottom, top) successively.

Modular and systematic - 4 simple clipping steps
Builds final result incrementally

Cohen–Sutherland

Assign Region Codes	For both end points P_1 and P_2

Perform Trivial Tests

Trivial Acceptance: (code1|code2)=0000

Trivial Rejection: (code1&code2)≠0000

Select Outside Point	Choose endpoint with non-zero region code
Find Inters-ection	Left/Right boundary (x = constant): $y=y_1+m(x-x_1)$ Top/Bottom boundary (y = constant): $x=x_1+ (y-y_1)/m$
Repeat	Continue until: Accepted (both codes = 0000), or Rejected (AND ≠ 0)