

CACS305 — Computer Graphics and Animation

Exam Cheat Sheet — All 5 Units — Quick Review

UNIT 1 — Introduction

Advantages & Applications

- Enables visualization of data, design & simulation not possible with text alone.
- Apps: CAD, GUI, presentation graphics, entertainment/movies, education, medical imaging, simulation & training, art, business & scientific visualization.

Hardware / Display Tech

Category	Examples
Hard-copy devices	Printer, Plotter
Display tech	CRT, LCD, Plasma, LED, DLP
Video Controller	Interfaces frame buffer to display; refreshes screen from memory

Random Scan vs Raster Scan

Random Scan	Raster Scan
Draws lines directly (vector) using a Display Processor	Scans pixel grid row by row from a frame buffer
High-res lines, flicker on complex images	Realistic shaded images; standard for modern displays
Uses refresh buffer of line-draw commands	Uses bitmap/frame buffer of pixel intensities

Scan Conversion — Key Formulas

DDA Line Algorithm

$m = (y_2 - y_1) / (x_2 - x_1)$; step x, compute $y += m$ (or vice-versa if $|m| > 1$)

Simple but uses float rounding — slower & less accurate than Bresenham

Bresenham Line Algorithm

Decision parameter: $p_k = 2\Delta y - \Delta x$ (initial)

If $p_k < 0$: next pixel $(x+1, y)$; $p_{k+1} = p_k + 2\Delta y$

Else: next pixel $(x+1, y+1)$; $p_{k+1} = p_k + 2\Delta y - 2\Delta x$

Bresenham Circle / Midpoint Ellipse

Circle: uses 8-way symmetry; decision $p_k = 1 - r$ (initial, at $(0, r)$)

Ellipse: midpoint method split into 2 regions by slope = -1 (dy/dx)

Area Filling & Clipping (recap)

- Flood Fill — replaces target color, works outward via 4/8-connectivity.
- Boundary Fill — fills until a specified boundary color is hit.
- Scan-Line Fill — fills polygon interior pixel-row by pixel-row using edge intersections.
- Clipping types: point, line, polygon, curve, text — full detail in Unit 3.

UNIT 2 — 2D and 3D Transformation

2D Transformations (Homogeneous, 3×3)

Translation / Scaling / Rotation

Translate: $x' = x + t_x$, $y' = y + t_y$

Scale: $x' = x \cdot s_x$, $y' = y \cdot s_y$ (about origin)

Rotate (θ , about origin): $x' = x \cos \theta - y \sin \theta$

$y' = x \sin \theta + y \cos \theta$

Reflection & Shear (2D)

Reflect about x-axis: $(x, y) \rightarrow (x, -y)$

Reflect about y-axis: $(x, y) \rightarrow (-x, y)$

Reflect about origin: $(x, y) \rightarrow (-x, -y)$

Shear-x: $x' = x + sh_x \cdot y$, $y' = y$ | Shear-y: $y' = y + sh_y \cdot x$, $x' = x$

Homogeneous coordinates add a 3rd coordinate $(x, y, 1)$ so translation can be combined with scaling/rotation into a single 3×3 matrix multiplication, enabling composite transformations via matrix multiplication order.

3D Transformations (Homogeneous, 4×4)

Translation / Scaling (3D)

Translate: $x' = x + t_x$, $y' = y + t_y$, $z' = z + t_z$

Scale: $x' = x \cdot s_x$, $y' = y \cdot s_y$, $z' = z \cdot s_z$

3D Rotation (about each axis)

About Z: $x' = x \cos \theta - y \sin \theta$, $y' = x \sin \theta + y \cos \theta$, $z' = z$

About X: $y' = y \cos \theta - z \sin \theta$, $z' = y \sin \theta + z \cos \theta$, $x' = x$

About Y: $z' = z \cos \theta - x \sin \theta$, $x' = z \sin \theta + x \cos \theta$, $y' = y$

3D Reflection & Shear

Reflect about xy-plane: $z \rightarrow -z$ (similarly for yz-, xz-planes)

Shear distorts an object along one axis proportional to another (e.g., $x' = x + sh_x \cdot z$), used for oblique/cavalier projections

Note: Composite transforms = multiply matrices in the order operations are applied (rightmost matrix applied first to the point vector).

UNIT 3 — Clipping

Window-to-Viewport

Mapping Equations

$$x_v = x_{vmin} + (x_w - x_{wmin}) \cdot S_x, S_x = (x_{vmax} - x_{vmin}) / (x_{wmax} - x_{wmin})$$
$$y_v = y_{vmin} + (y_w - y_{wmin}) \cdot S_y, S_y = (y_{vmax} - y_{vmin}) / (y_{wmax} - y_{wmin})$$

Cohen-Sutherland Line Clipping

- 4-bit region code [Top Bottom Right Left]; 0000 = inside window.
- OR of both codes = 0 → Trivial Accept (draw as-is).
- AND of both codes ≠ 0 → Trivial Reject (discard line).

- Else → clip against a violated boundary, recompute code, repeat.

Intersection formulas

vs Left/Right ($x=x_b$): $y = y_1 + m(x_b - x_1)$

vs Top/Bottom ($y=y_b$): $x = x_1 + (y_b - y_1)/m$

Sutherland–Hodgman Polygon Clipping

Clip polygon against each boundary (L,R,B,T) in turn; output list of one boundary feeds input of the next.

Case (V1→V2)	Output
Both inside	V2
In → Out	Intersection point
Both outside	Nothing
Out → In	Intersection, then V2

UNIT 4 — Visible Surface Determination & Algorithms

Object-Space vs Image-Space

Object Space	Image Space
Compares objects/surfaces in world coords (geometric)	Determines visibility per-pixel on screen
e.g. Back-face removal, Painter's Algorithm	e.g. Z-Buffer, Scan-Line Algorithm

Z-Buffer (Depth-Buffer) Algorithm

- Maintain a depth buffer (init = ∞ /far) and frame buffer (init = background).
- For each pixel of each polygon: if new $z < \text{stored depth}[x,y]$, update both depth buffer and frame buffer (color); else discard.
- Simple, works on unsorted polygons; memory-intensive.

Back-Face Removal

Test Condition

Surface is a back face (hidden, facing away from viewer) if:

$N \cdot V \geq 0$ (dot product of normal N and view vector V)

Quick test using only the surface's normal — no depth compare needed

Painter's Algorithm

- Sort polygons by depth (farthest first).
- Render/paint back-to-front so nearer polygons overwrite farther ones.
- Fails for cyclic or interpenetrating polygons without splitting.

Scan-Line Algorithm

- Processes the image one scan-line (row) at a time.
- Uses edge tables & active-edge lists to find polygon spans intersecting the line.
- Resolves visibility per span using depth at each intersecting edge.

Color Models

Model	Basis / Use
RGB	Red-Green-Blue additive model; used for displays/monitors
CMY	Cyan-Magenta-Yellow subtractive model; used for printing (CMY = 1- RGB)
YIQ	Luminance (Y) + Chrominance (I,Q); used in NTSC TV broadcast, backward B/W compatible

UNIT 5 — Animation and Virtual Reality

Types of Animation

- Traditional/Cel, Frame-by-Frame, 2D (tween-based), 3D, Stop-Motion, Motion Graphics.

12 Basic Principles (condensed)

- Squash & Stretch • Anticipation • Staging • Straight-Ahead / Pose-to-Pose
- Follow-Through & Overlap • Slow In/Slow Out • Arcs • Secondary Action
- Timing • Exaggeration • Solid Drawing • Appeal

Flash/Animate Interface

Component	Role
Stage	Visible canvas area of the movie
Timeline	Layers + frames controlling content over time
Toolbox	Drawing/selection tools (Line, Oval, Pen, Brush...)
Panels	Properties, Library, Color, Align, Actions

Stage/Panel Setup

- Stage props: width×height (px), background color, frame rate (fps, default ~24), ruler units.
- Panel layouts can be docked, tabbed, floated, or saved as custom workspace.

Layers & Views

Layer Type	Use
Normal	Standard artwork/animation
Guide / Motion Guide	Reference / path for motion (not rendered / defines path)
Mask	Reveals only the portion of layer(s) below within its shape
Folder	Groups layers for organization

- Views: zoom, outline mode, onion skinning (see faint neighboring frames).

Shapes, Bitmaps & Sounds

- Shape = Stroke (outline) + Fill (solid/gradient/bitmap).
- Tools: Line, Oval, Rectangle, Pencil, Pen, Brush; modify via Selection, Subselection, Free Transform, Envelope.
- Bitmaps: import → library; can fill shapes, be traced to vectors, or broken apart.
- Sound sync: Event (independent), Start, Stop, Streaming (locked to timeline).

Frame-by-Frame vs Tweening vs Masks

Technique	Key Idea
Frame-by-Frame	Every frame hand-drawn individually — full control, more work
Motion/Classic Tween	Animates position/scale/rotation/color between 2 keyframes
Shape Tween	Morphs one vector shape into another
Mask	Top layer shape reveals only overlapping area of layer(s) below

Virtual Reality

- Components: HMD (stereo display), tracking sensors, hand controllers, haptics, processing unit.
- VR = fully replaces real world | AR = overlays on real world | MR = blends & interacts real+virtual.
- Apps: gaming, training/education, architecture walkthroughs, therapy, virtual tourism.

Exam-Day Quick Recall

- ✓ Bresenham uses integer decision parameters — faster than DDA.
- ✓ Homogeneous coords (x,y,1) let translation combine with scale/rotate as one matrix.
- ✓ Cohen-Sutherland: $OR=0 \rightarrow \text{accept}$; $AND \neq 0 \rightarrow \text{reject}$; else clip.
- ✓ Z-Buffer compares per-pixel depth; Painter's sorts whole polygons back-to-front.
- ✓ $N \cdot V \geq 0 \rightarrow \text{back face (cull it)}$.
- ✓ Tweening needs only start/end keyframes; frame-by-frame needs every frame.
- ✓ Mask layer shape defines the visible 'window' onto the layer(s) beneath it.