1. Basics of Computer Graphics

1. Define computer graphics and its applications.

Computer Graphics (CG) is the field of visual computing where images and animations are created, manipulated, and displayed using computers. It involves the generation and representation of pictorial data to convey information or create visual effects. CG combines concepts from mathematics, physics, and computer science to render images, both 2D and 3D, that can be static or dynamic.

Applications of computer graphics are vast and impact many domains:

- **Entertainment:** Movies, video games, virtual reality (VR), and augmented reality (AR) use CG for realistic or stylized visuals.
- **Design and Engineering:** CAD (Computer-Aided Design) software relies on CG for product design, architectural visualization, and simulations.
- **Education and Training:** Simulations and visualizations help in teaching complex concepts, like anatomy or aerodynamics.
- **Scientific Visualization:** Data such as weather patterns, molecular structures, or astronomical data is rendered graphically for analysis.
- **User Interfaces:** Graphical User Interfaces (GUIs) make software accessible and interactive.

Thus, CG bridges the gap between raw data and human visual perception, enhancing understanding and communication.

2. Explain raster and vector graphics.

Raster and vector graphics are two fundamental methods for representing digital images.

- **Raster Graphics** represent images as a grid of pixels, where each pixel stores color information. Common formats include JPEG, PNG, and BMP. Raster images are resolution-dependent; zooming in causes pixelation and loss of quality. Raster graphics are ideal for complex images like photographs with continuous tone variations. Each pixel's color is stored independently, making editing specific image parts detailed but potentially resource-heavy.
- **Vector Graphics** represent images using mathematical equations and geometric primitives like points, lines, curves, and polygons. They are resolution-independent, meaning they can be scaled infinitely without losing quality. Common vector formats include SVG, EPS, and AI. Vector graphics are preferred for logos, fonts, and illustrations requiring scalability and precise shapes. Editing involves changing the properties of objects rather than pixels.

Choosing between raster and vector depends on the use case: raster for detailed images, vector for scalable designs.

3. What is resolution and aspect ratio?

Resolution in computer graphics refers to the number of pixels displayed on the screen, typically given as width × height (e.g., 1920×1080 pixels). It determines the image's clarity and detail. Higher resolution means more pixels per unit area, resulting in sharper and more detailed images. For digital displays, resolution also affects how crisp text and graphics appear.

Aspect Ratio is the proportional relationship between the width and height of an image or screen, expressed as a ratio (e.g., 16:9, 4:3). It dictates the shape of the displayed image. Maintaining correct aspect ratios ensures images and videos are not stretched or squished, preserving the original proportions.

Both resolution and aspect ratio are critical for designing graphics for various devices and media, ensuring visual consistency and quality.

4. Describe the working of a graphics pipeline.

A graphics pipeline is a conceptual model describing the steps involved in rendering a 3D scene into a 2D image on the screen. It is a sequence of stages that transforms 3D models into pixels.

The key stages include:

- **Application Stage:** Prepares data and issues drawing commands.
- **Geometry Processing:** Applies transformations (translation, rotation, scaling) to vertices; performs lighting calculations.
- **Rasterization:** Converts vector information (primitives like triangles) into fragments (potential pixels).
- **Fragment Processing:** Colors fragments based on lighting, textures, and shading.
- **Output Merging:** Combines fragments into the final framebuffer, managing depth and transparency.

Modern graphics pipelines are implemented in hardware (GPU) for speed and can be programmable (e.g., using shaders). This pipeline abstracts complex processes, allowing efficient rendering of detailed 3D scenes.

2. 2D and 3D Transformations

1. Explain 2D transformations: translation, rotation, scaling.

2D transformations alter the position, orientation, or size of objects in a 2D plane:

- **Translation:** Moves an object by shifting its coordinates by a fixed amount (dx, dy). Formula: (x', y') = (x + dx, y + dy).
- **Rotation:** Rotates the object around an origin by an angle θ. New coordinates are calculated using trigonometric functions:

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

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

- **Scaling:** Changes the size of the object by multiplying coordinates by scale factors (sx, sy): $(x', y') = (sx \times x, sy \times y)$.

These transformations allow manipulation of shapes for positioning, orientation, and resizing in graphics applications.

2. Define homogeneous coordinates.

Homogeneous coordinates extend the traditional Cartesian coordinates by adding an extra dimension, typically represented as (x, y, w) in 2D and (x, y, z, w) in 3D. This allows representation of translation as a matrix multiplication, which cannot be done using only 2x2 or 3x3 matrices for scaling and rotation.

For 2D:
$(x, y) \rightarrow (xw, yw, w)$
Usually, $w = 1$ for points. Using homogeneous coordinates enables all affine transformations (translation, rotation, scaling, shearing) to be expressed as matrix multiplications, simplifying the graphics pipeline and concatenation of transformations.

3. What are 3D transformations?

3D transformations manipulate objects in three-dimensional space and include translation, scaling, and rotation extended to the z-axis:

- **Translation:** Moves the object along x, y, and z axes.
- **Scaling:** Enlarges or shrinks objects along each axis.
- **Rotation:** Can be about x, y, or z-axis or an arbitrary axis using rotation matrices.

Like 2D, these transformations are expressed using 4×4 matrices in homogeneous coordinates, facilitating efficient computation and chaining.

3D transformations enable positioning, orienting, and resizing objects in a virtual scene before projection to 2D screens.

4. Describe viewing and projection transformations.

Viewing Transformation: Converts the world coordinates into a camera or eye coordinate system. It defines the viewpoint, orientation, and position of the virtual camera, enabling the scene to be viewed from different angles.

Projection Transformation: Maps 3D coordinates onto a 2D plane (screen). There are two main types:

- **Orthographic projection:** Parallel projection preserving object size regardless of depth.
- **Perspective projection:** Mimics human vision; objects farther away appear smaller, adding depth realism.

Together, these transformations allow realistic rendering of 3D scenes onto 2D displays.

3. Algorithms and Techniques

1. Explain Bresenham's line drawing algorithm.

Bresenham's algorithm efficiently draws a straight line between two points on a raster display using only integer arithmetic, avoiding floating-point operations for speed.

It determines the pixels closest to the ideal line by incrementally deciding whether to move vertically or diagonally based on an error term. For a line with slope between 0 and 1, the algorithm:

- Starts at the first pixel.
- For each x-coordinate, it chooses either the current y or y+1 pixel based on the error.
- Updates the error and moves to the next x.

This method minimizes computation and produces visually smooth lines, making it widely used in graphics hardware and software.

2. What is the midpoint circle algorithm?

The midpoint circle algorithm draws a circle by determining the next pixel closest to the ideal circle boundary using only integer addition and subtraction.

Starting at the top of the circle, it calculates a decision parameter (midpoint between two possible pixels). Based on this parameter, it chooses the next pixel moving horizontally or diagonally. It uses the circle's symmetry to plot points in all eight octants simultaneously, optimizing performance.

It avoids floating-point arithmetic and square roots, making it efficient for raster graphics.

3. Define clipping. Explain Cohen-Sutherland algorithm.

Clipping is the process of confining rendering to a specified viewport or window, removing portions of graphics outside the visible area to improve efficiency.

The Cohen-Sutherland algorithm is a line clipping algorithm that uses region codes (4-bit codes) for endpoints of a line segment to quickly decide if a line is completely inside, outside, or partially inside the clipping window.

The algorithm:

- Assigns region codes based on location relative to the clip window.

- Uses bitwise operations on these codes to trivial accept or reject lines.
- Iteratively clips lines crossing boundaries until fully inside or discarded.

This algorithm efficiently handles rectangular clipping windows and is fundamental in computer graphics.

4. What is scan-line polygon filling?

Scan-line polygon filling is a technique for coloring the interior of polygons on raster displays by processing one horizontal line (scan-line) at a time.

It involves:

- Identifying intersections of the polygon edges with the current scan-line.
- Sorting intersections from left to right.
- Filling pixels between pairs of intersections.

This method is efficient for convex and concave polygons and handles complex shapes better than simple algorithms. It is commonly used in raster graphics rendering to fill polygons with color or patterns.

4. Animation Basics

1. Define animation and its types.

Animation is the process of creating the illusion of motion by displaying a sequence of images or frames. These frames, shown rapidly in succession, trick the eye into perceiving continuous movement.

Types of animation include:

- **2D Animation:** Traditional frame-by-frame drawings or computer-generated flat images.

- **3D Animation:** Uses 3D models and environments to create lifelike or stylized motion.
- **Stop Motion:** Physical objects are moved incrementally and photographed frame-by-frame.
- **Motion Graphics:** Animated graphic design elements for presentations or UI.

Each type varies in complexity, tools, and applications.

2. Explain keyframe, tweening, and morphing.

Keyframe Animation: Animators set critical frames defining start and end positions. These "key" frames specify major points in the motion.

Tweening (Inbetweening): Intermediate frames are automatically generated between keyframes to create smooth transitions. Tweening reduces manual work by interpolating positions, colors, or other attributes.

Morphing: Smoothly transforms one shape into another by gradually changing vertex positions or pixel values, creating a fluid transformation effect.

These techniques combine to produce realistic and efficient animations.

3. What are the principles of animation?

The 12 principles of animation, established by Disney animators, provide guidelines for creating believable and appealing animations:

- Squash and stretch
- Anticipation
- Staging
- Straight ahead action and pose to pose
- Follow-through and overlapping action

- Slow in and slow out
- Arc
- Secondary action
- Timing
- Exaggeration
- Solid drawing
- Appeal

These principles enhance the illusion of life, weight, and emotion in animated characters and objects.

4. Difference between 2D and 3D animation.

2D Animation involves flat images and characters, created by drawing or vector graphics. Movement is simulated by altering shapes frame-by-frame on a two-dimensional plane. It is simpler but limited in depth perception.

3D Animation uses computer-generated 3D models with depth, lighting, and textures, allowing more realistic and complex movements. Objects are manipulated in a 3D virtual environment, enabling camera rotations and detailed visual effects.

3D animations generally require more computational resources but offer greater realism and flexibility.

5. Advanced Topics and Applications

1. What is ray tracing and how does it work?

Ray tracing is a rendering technique that simulates the path of light rays interacting with objects in a scene to produce highly realistic images.

It works by:

- Tracing rays from the eye (camera) into the scene.
- Calculating intersections with objects.
- Computing light interactions such as reflection, refraction, and shadows by tracing secondary rays.
- Accumulating color and light intensity at each pixel.

Ray tracing produces accurate shadows, reflections, and global illumination but is computationally intensive, used mostly in offline rendering and increasingly in real-time graphics with powerful GPUs.

2. Explain the role of OpenGL or DirectX.

OpenGL and DirectX are graphics APIs that provide standardized interfaces for programmers to interact with GPUs and create graphics applications.

- **OpenGL:** Cross-platform API widely used in scientific visualization, CAD, and gaming. It supports rendering 2D and 3D graphics, shader programming, and hardware acceleration.
- **DirectX:** A Microsoft Windows-exclusive API suite, with Direct3D for graphics. It integrates tightly with Windows OS and is extensively used in gaming.

Both APIs abstract low-level hardware details, enabling efficient graphics programming, real-time rendering, and multimedia development.

3. Describe shading models: flat, Gouraud, Phong.

Flat Shading: Calculates lighting once per polygon, giving a uniform color. It is fast but results in faceted appearances.

Gouraud Shading: Computes lighting at polygon vertices and interpolates colors across surfaces. It smooths shading but can miss specular highlights.

Phong Shading: Interpolates surface normals across polygons and computes lighting per pixel, resulting in more accurate and smoother highlights.

Phong shading is more computationally expensive but produces realistic results and is widely used in modern graphics.

4. Applications of computer graphics in real life

Computer graphics permeate many real-world applications:

- **Entertainment:** Video games, movies, virtual reality.
- **Medicine:** Visualizing medical scans, surgical simulations.
- **Education:** Interactive learning, scientific simulations.
- **Engineering:** CAD, 3D modeling, and prototyping.
- **Architecture:** Virtual walkthroughs and urban planning.
- **Advertising:** Digital marketing and visual effects.
- **User Interfaces:** Enhanced, intuitive GUIs.

Graphics technologies improve visualization, understanding, and communication across industries.