SNS COLLEGE OF ENGINEERING

Kurumbapalayam(Po), Coimbatore – 641 107
Accredited by NAAC-UGC with 'A' Grade
Approved by AICTE, Recognized by UGC & Affiliated to Anna University, Chennai

Department of Information Technology

Computer Graphics

Unit 1 : INTRODUCTION TO COMPUTER GRAPHICS

Topic :OPENGL Basics Primitives

Prepared By
R.Vaishnavi.,AP/IT
SNSCE.

OPENGL

- A low-level graphics library specification.
- OpenGL (Open Graphics Library) is a widely used graphics API (Application Programming Interface) that allows developers to create 2D and 3D graphics in various applications, including video games, simulations, and graphical user interfaces.
- A small set of geometric primitives
  - Points
  - Lines
  - Polygons

  Geometric primitives
  - Images
  - Bitmaps

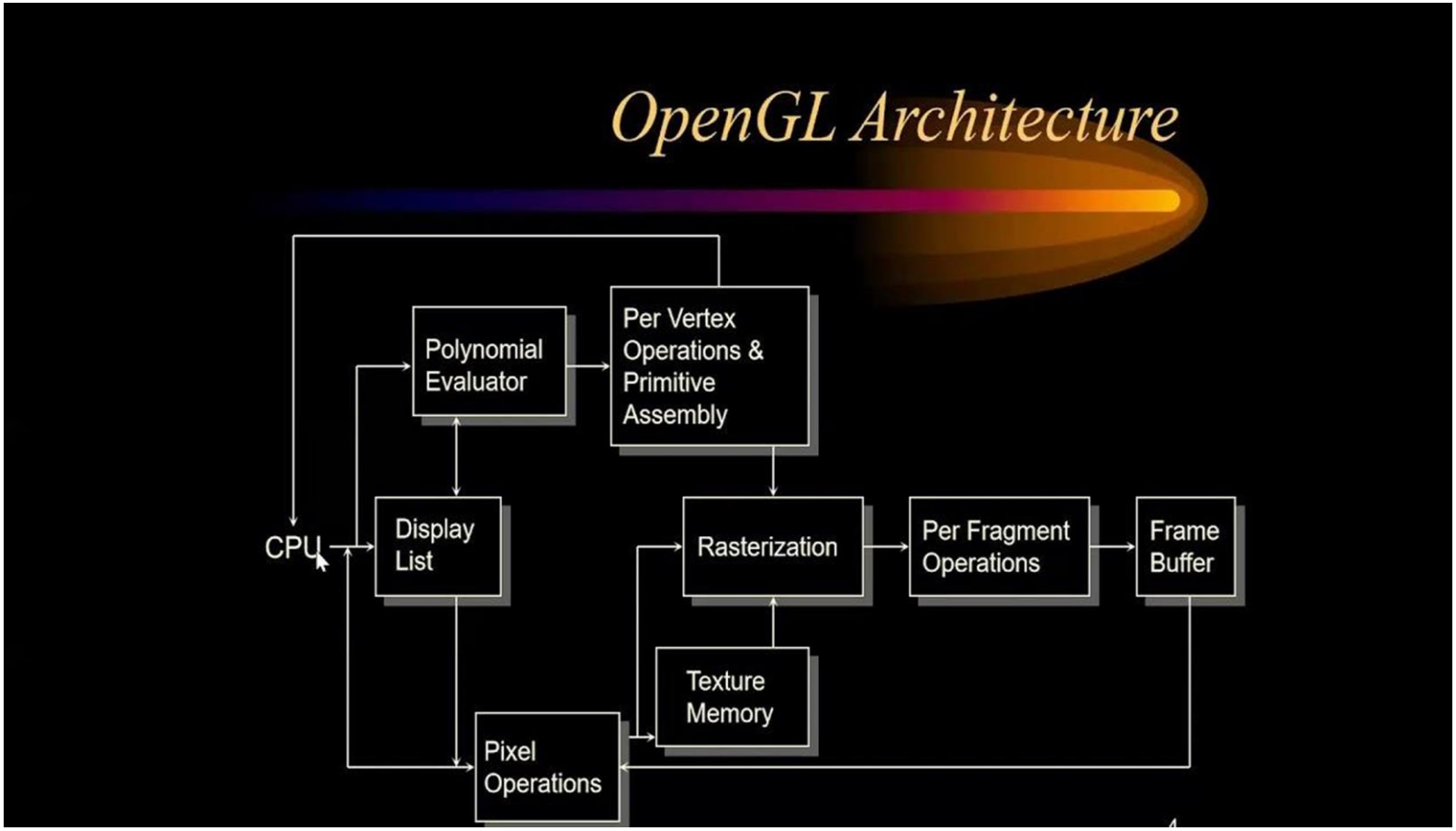
  Image primitives

Abstractions

GLUT
- Windowing toolkit (key, mouse handler, window events)

GLU
- Viewing –perspective/orthographic
- Image scaling, polygon tessellation
- Sphere, cylinders, quadratic surfaces

GL
- Primitives - points, line, polygons
- Shading and Colour
- Translation, rotation, scaling
- Viewing, Clipping, Texture
- Hidden surface removal


OpenGL Architecture
Polynomial Evaluator
Per Vertex Operations & Primitive Assembly
CPU
Display List
Rasterization
Per Fragment Operations
Frame Buffer
Texture Memory
Pixel Operations

TYPES OF OPENGL FUNCTIONS

- **Setting Functions**
  - Enable/disable functionality
  - Control OpenGL state
  - **Example**: alpha, transforms
- glEnable(capability);
- glDisable(capability);
- glLightfv(light, pName, pValue);
- glTranslate(x, y, z);

- **Data Handling Functions**
  - Create persistent structures
  - Involves memory allocation
  - **Example**: Texture loading
- glVertexPointer(...);
- glGenTextures(size, names);
- glDeleteTextures(size, names);
- glTexImage2D(target, level,...);

- **Rendering Functions**
  - Draw and texture primitives
  - **Example**: triangles, quads
- glBegin()/glEnd()
- glVertex3f(x,y,z);
- glDrawElements(...);

OpenGL Primitives

1. GL_POINTS:

- Treats each vertex as a single point.
- Vertex n defines a point n.
- N points are drawn.

```
glBegin(GL_POINTS);
glVertex2f(x1,y1);
glEnd();
```

2. GL_LINES:

- Treats each pair of vertices as an independent line segment.
- Vertices 2n-1 and 2n define a line n.
- N/2 lines are drawn.

```
glBegin(GL_LINES);
glVertex2f(x1,y1);
glVertex2f(x2,y2);
glEnd();
```

3. GL_LINE_STRIP:

- Draws a connected group of line segments from the first vertex to the last.
- Vertices n and n+1 define line n.
- N-1 lines are drawn.

```
glBegin(GL_LINE_STRIP);
    glVertex2f(x1, y1);
    glVertex2f(x2, y2);
    glVertex2f(x3, y3);
        glEnd();
```

4. GL_LINE_LOOP:

- Draws a connected group of line segments from the first vertex to the last, then back to the first.
- Vertices n and n+1 define line n.
- N lines are drawn.

```
glBegin(GL_LINE_LOOP);
    glVertex2f(x1, y1);
    glVertex2f(x2, y2);
    glVertex2f(x3, y3);
        glEnd();
```

5.GL_TRIANGLES:

- Treats each triplet of vertices as an independent triangle.
- Vertices 3n-2, 3n-1, and 3n define triangle n.
- N/3 triangles are drawn.

```
glBegin(GL_TRIANGLES);
    glVertex2f(x1, y1);
    glVertex2f(x2, y2);
    glVertex2f(x3, y3);
    glEnd();
```

6.GL_QUADS:

- Treats each group of four vertices as an independent quadrilateral.
- Vertices 4n-3, 4n-2, 4n-1, and 4n define quadrilateral n.
- N/4 quadrilaterals are drawn.

```
glBegin(GL_QUADS);
    glVertex2f(x1, y1);
    glVertex2f(x2, y2);
    glVertex2f(x3, y3);
    glVertex2f(x4, y4);
    glEnd();
```

7. GL_TRIANGLE_STRIP:

- Draws a connected group of triangles.
- One triangle is defined for each vertex presented after the first two vertices.
- For odd n, vertices n, n+1, and n+2 define triangle n.
- For even n, vertices n+1, n, and n+2 define triangle n.
- N-2 triangles are drawn.

$$glBegin(GL_LINE_STRIP);$$
$$glVertex2f(x1, y1);$$
$$glVertex2f(x2, y2);$$
$$glVertex2f(x3, y3);$$
$$glEnd();$$

8.GL_TRIANGLE_FAN:

- Draws a connected group of triangles that fan around a central point.
- One triangle is defined for each vertex presented after the first two vertices.
- Vertices 1, n+1, and n+2 define triangle n.
- N-2 triangles are drawn.

```
glBegin(GL_TRIANGLE_FAN);
    glVertex2f(x1, y1);
    glVertex2f(x2, y2);
    glVertex2f(x3, y3);
    glVertex2f(x4, y4);
    glEnd();
```

9.GL_QUAD_STRIP:

- Draws a connected group of quadrilaterals.
- One quadrilateral is defined for each pair of vertices presented after the first pair.
- Vertices 2n-1, 2n, 2n+2, and 2n+1 define quadrilateral n.
- N/2-1 quadrilaterals are drawn.

```
glBegin(GL_QUADS_TRIP);
    glVertex2f(x1, y1);
    glVertex2f(x2, y2);
    glVertex2f(x3, y3);
    glVertex2f(x4, y4);
    glVertex2f(x5, y5);
    glVertex2f(x6, y6);
        glEnd();
```

10.GL_POLYGON:

- Draws a single and convex polygon.
- Vertices 1 through N define this polygon.
- A polygon is convex if all points on the line segment between any two points in the polygon or at the boundary of the polygon lie inside the polygon.

```
glBegin(GL_POLYGON);
    glVertex2f(x1, y1);
    glVertex2f(x2, y2);
    glVertex2f(x3, y3);
    glVertex2f(x4, y4);
    glVertex2f(x5, y5);
    glEnd();
```