Computer Graphics

Dr. Keshav Sinha

Lab 2: Learn about the Primitives

• **Line Generation** • **Point Generation** • **Polygon Generation** • **GL_TRIANGLES** • **GL_TRIANGLE_STRIP**	• **GL_QUADS** • **GL_QUAD_STRIP** • **GL_LINE_STRIP** • **GL_LINE_LOOP** • **GL_TRIANGLE_FAN**

LAB Performance (In Lab Only)

1. **Create the Word file and paste all the output images into that file.**
2. **Write the answer to the given question in the notebook.**

1. Run all the primitive code and take the screenshot. Make the **Call Back Function with your name**. **(4M)**
2. What is the primary purpose of the `GL_TRIANGLE_FAN` primitive in OpenGL? **(0.5M)**
3. In the `GL_TRIANGLE_FAN` mode, how many vertices are required to form a complete fan shape? **(0.5M)**
4. What would be the visual result if you specify only two vertices in `GL_TRIANGLE_FAN` mode? **(0.5M)**
5. How does `GL_LINE_STRIP` differ from `GL_LINE_LOOP`? **(0.5M)**
6. If you specify the following vertices for `GL_LINE_STRIP`: (10,10), (20,20), (30,30), what will be the result? **(0.5M)**
7. What happens if you specify only one vertex in `GL_LINE_STRIP` mode? **(0.5M)**
8. How are quadrilaterals defined in `GL_QUAD_STRIP` mode? **(0.5M)**
9. What effect does specifying more vertices than needed for `GL_QUAD_STRIP` have? **(0.5M)**
10. If you specify the following vertices for `GL_QUAD_STRIP`: (10,10), (20,10), (10,20), (20,20), (30,30), (40,30), how many quads will be drawn? **(0.5M)**
11. What is the role of `glColor3f` in OpenGL drawing functions? **(0.5M)**
12. How does `glFlush()` differ from `glFinish()` in OpenGL? **(0.5M)**
13. Why might you use `gluOrtho2D` in an OpenGL program? **(0.5M)**

Line Generation

```
#include <GL/glut.h>

// Function to initialize OpenGL settings
void init() {
   glClearColor(0.0, 0.0, 0.0, 0.0); // Set background color to black
   glColor3f(1.0, 1.0, 1.0);         // Set the drawing color to white
   gluOrtho2D(0.0, 100.0, 0.0, 100.0); // Set the coordinate system
}

// Display callback function to draw lines
void kesh() {
   glClear(GL_COLOR_BUFFER_BIT); // Clear the screen

   glBegin(GL_LINES); // Begin drawing lines

   // Specify pairs of points to draw lines between
   glVertex2i(10, 10);
   glVertex2i(90, 10);

   glVertex2i(10, 20);
   glVertex2i(90, 20);

   glVertex2i(10, 30);
   glVertex2i(90, 30);

   glVertex2i(10, 40);
   glVertex2i(90, 40);

   glVertex2i(10, 50);
   glVertex2i(90, 50);

   glVertex2i(10, 60);
   glVertex2i(90, 60);

   glVertex2i(10, 70);
   glVertex2i(90, 70);

   glVertex2i(10, 80);
   glVertex2i(90, 80);

   glVertex2i(10, 90);
   glVertex2i(90, 90);
```

```
    glEnd(); // End drawing

    glFlush(); // Flush the OpenGL commands
}

// Main function
int main(int argc, char** argv) {
    glutInit(&argc, argv); // Initialize GLUT
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB); // Set display mode
    glutInitWindowSize(500, 500); // Set the window size
    glutInitWindowPosition(100, 100); // Set the window position
    glutCreateWindow("OpenGL Line Drawing"); // Create the window
    init(); // Initialize OpenGL settings
    glutDisplayFunc(kesh); // Set the display callback function
    glutMainLoop(); // Enter the GLUT event processing loop
    return 0;
}
```

Point Generation

```
#include <GL/glut.h>

void init() {
    glClearColor(0.0, 0.0, 0.0, 0.0);
    glColor3f(1.0, 1.0, 1.0);
    gluOrtho2D(0.0, 100.0, 0.0, 100.0);
}

void kesh() {
    glClear(GL_COLOR_BUFFER_BIT);
    glBegin(GL_POINTS);

    glVertex2i(10, 10);
    glVertex2i(20, 20);
    glVertex2i(30, 30);
    glVertex2i(40, 40);
    glVertex2i(50, 50);
    glVertex2i(60, 60);
    glVertex2i(70, 70);
    glVertex2i(80, 80);
    glVertex2i(90, 90);
    glVertex2i(50, 90);
```

```
    glEnd();
    glFlush();
}

int main(int argc, char** argv) {
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(500, 500);
    glutInitWindowPosition(100, 100);
    glutCreateWindow("OpenGL 10 Points Generation");
    init();
    glutDisplayFunc(kesh);
    glutMainLoop();
    return 0;
}
```

Polygon Generation

```
#include <GL/glut.h>

// Function to initialize OpenGL settings
void init() {
    glClearColor(0.0, 0.0, 0.0, 0.0); // Set background color to black
    glColor3f(1.0, 1.0, 1.0);         // Set the drawing color to white
    gluOrtho2D(0.0, 100.0, 0.0, 100.0); // Set the coordinate system
}

// Display callback function to draw a polygon
void kesh() {
    glClear(GL_COLOR_BUFFER_BIT); // Clear the screen

    glBegin(GL_POLYGON); // Begin drawing the polygon

    // Specify the vertices of the polygon
    glVertex2i(20, 20);
    glVertex2i(80, 20);
    glVertex2i(90, 50);
    glVertex2i(50, 80);
    glVertex2i(10, 50);

    glEnd(); // End drawing

    glFlush(); // Flush the OpenGL commands
}
```

```
// Main function
int main(int argc, char** argv) {
    glutInit(&argc, argv); // Initialize GLUT
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB); // Set display mode
    glutInitWindowSize(500, 500); // Set the window size
    glutInitWindowPosition(100, 100); // Set the window position
    glutCreateWindow("OpenGL Polygon Drawing"); // Create the window
    init(); // Initialize OpenGL settings
    glutDisplayFunc(kesh); // Set the display callback function
    glutMainLoop(); // Enter the GLUT event processing loop
    return 0;
}
```

GL_TRIANGLES

```
#include <GL/glut.h>

// Function to initialize OpenGL settings
void init() {
    glClearColor(0.0, 0.0, 0.0, 0.0); // Set background color to black
    glColor3f(1.0, 1.0, 1.0);         // Set the drawing color to white
    gluOrtho2D(0.0, 100.0, 0.0, 100.0); // Set the coordinate system
}

// Display callback function to draw a shape using triangles
void kesh() {
    glClear(GL_COLOR_BUFFER_BIT); // Clear the screen

    glBegin(GL_TRIANGLES); // Begin drawing triangles

    // Define the first triangle
    glVertex2i(50, 70);
    glVertex2i(30, 40);
    glVertex2i(70, 40);

    glEnd(); // End drawing

    glFlush(); // Flush the OpenGL commands
}

int main(int argc, char** argv) {
    glutInit(&argc, argv); // Initialize GLUT
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB); // Set display mode
    glutInitWindowSize(500, 500); // Set the window size
    glutInitWindowPosition(100, 100); // Set the window position
```

```
    glutCreateWindow("OpenGL Triangle Drawing"); // Create the window
    init(); // Initialize OpenGL settings
    glutDisplayFunc(kesh); // Set the display callback function
    glutMainLoop(); // Enter the GLUT event processing loop
    return 0;
}
```

GL_TRIANGLE_STRIP

```
#include <GL/glut.h>

// Function to initialize OpenGL settings
void init() {
    glClearColor(0.0, 0.0, 0.0, 0.0); // Set background color to black
    gluOrtho2D(0.0, 100.0, 0.0, 100.0); // Set the coordinate system
}

// Display callback function to draw a shape using triangle strips with different
vertex colors
void kesh() {
    glClear(GL_COLOR_BUFFER_BIT); // Clear the screen

    glBegin(GL_TRIANGLE_STRIP); // Begin drawing with triangle strips

    // Define the vertices with different colors
    glColor3f(1.0, 0.0, 0.0); // Red color
    glVertex2i(30, 30); // Vertex 1

    glColor3f(0.0, 1.0, 0.0); // Green color
    glVertex2i(70, 30); // Vertex 2

    glColor3f(0.0, 0.0, 1.0); // Blue color
    glVertex2i(50, 70); // Vertex 3

    glColor3f(1.0, 1.0, 0.0); // Yellow color
    glVertex2i(30, 70); // Vertex 4

    glColor3f(1.0, 0.0, 1.0); // Magenta color
    glVertex2i(70, 70); // Vertex 5

    glEnd(); // End drawing

    glFlush(); // Flush the OpenGL commands
}
```

```
// Main function
int main(int argc, char** argv) {
    glutInit(&argc, argv); // Initialize GLUT
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB); // Set display mode
    glutInitWindowSize(500, 500); // Set the window size
    glutInitWindowPosition(100, 100); // Set the window position
    glutCreateWindow("OpenGL Triangle Strip with Vertex Colors"); // Create the
window
    init(); // Initialize OpenGL settings
    glutDisplayFunc(kesh); // Set the display callback function
    glutMainLoop(); // Enter the GLUT event processing loop
    return 0;
}
```

GL_QUADS

```
#include <GL/glut.h>

// Function to initialize OpenGL settings
void init() {
    glClearColor(0.0, 0.0, 0.0, 0.0); // Set background color to black
    gluOrtho2D(0.0, 100.0, 0.0, 100.0); // Set the coordinate system
}

// Display callback function to draw a quadrilateral with vertex colors
void kesh() {
    glClear(GL_COLOR_BUFFER_BIT); // Clear the screen

    glBegin(GL_QUADS); // Begin drawing quadrilaterals

       // Define the vertices with different colors
    glColor3f(1.0, 0.0, 0.0); // Red color
    glVertex2i(5, 5); // Bottom-left

    glColor3f(0.0, 1.0, 0.0); // Green color
    glVertex2i(10, 5); // Bottom-right

    glColor3f(0.0, 0.0, 1.0); // Blue color
    glVertex2i(10, 10); // Top-right

    glColor3f(1.0, 1.0, 0.0); // Yellow color
    glVertex2i(5, 10); // Top-left



    // Define the vertices with different colors
```

```
    glColor3f(1.0, 0.0, 0.0); // Red color
    glVertex2i(30, 30); // Bottom-left

    glColor3f(0.0, 1.0, 0.0); // Green color
    glVertex2i(70, 30); // Bottom-right

    glColor3f(0.0, 0.0, 1.0); // Blue color
    glVertex2i(70, 70); // Top-right

    glColor3f(1.0, 1.0, 0.0); // Yellow color
    glVertex2i(30, 70); // Top-left

    glEnd(); // End drawing

    glFlush(); // Flush the OpenGL commands
}

// Main function
int main(int argc, char** argv) {
    glutInit(&argc, argv); // Initialize GLUT
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB); // Set display mode
    glutInitWindowSize(500, 500); // Set the window size
    glutInitWindowPosition(100, 100); // Set the window position
    glutCreateWindow("OpenGL Quadrilateral with Vertex Colors"); // Create the
window
    init(); // Initialize OpenGL settings
    glutDisplayFunc(kesh); // Set the display callback function
    glutMainLoop(); // Enter the GLUT event processing loop
    return 0;
}
```

GL_LINE_STRIP

```
#include <GL/glut.h>

// Function to initialize OpenGL settings
void init() {
    glClearColor(0.0, 0.0, 0.0, 0.0); // Set background color to black
    glColor3f(1.0, 1.0, 1.0);         // Set the default drawing color to white
    gluOrtho2D(0.0, 100.0, 0.0, 100.0); // Set the coordinate system
}

// Display callback function to draw lines using line strips with different colors
void kesh() {
    glClear(GL_COLOR_BUFFER_BIT); // Clear the screen
```

```
    // Draw the first line segment
    glBegin(GL_LINE_STRIP);

    // Define colors and vertices
    glColor3f(1.0, 0.0, 0.0); // Red color
    glVertex2i(10, 10); // Vertex 1

    glColor3f(0.0, 1.0, 0.0); // Green color
    glVertex2i(30, 30); // Vertex 2

    glColor3f(0.0, 0.0, 1.0); // Blue color
    glVertex2i(50, 10); // Vertex 3

    glColor3f(1.0, 1.0, 0.0); // Yellow color
    glVertex2i(70, 30); // Vertex 4

    glColor3f(1.0, 0.0, 1.0); // Magenta color
    glVertex2i(90, 10); // Vertex 5

    glEnd(); // End drawing

    glFlush(); // Flush the OpenGL commands
}

// Main function
int main(int argc, char** argv) {
    glutInit(&argc, argv); // Initialize GLUT
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB); // Set display mode
    glutInitWindowSize(500, 500); // Set the window size
    glutInitWindowPosition(100, 100); // Set the window position
    glutCreateWindow("OpenGL Line Strip with Vertex Colors"); // Create the
window
    init(); // Initialize OpenGL settings
    glutDisplayFunc(kesh); // Set the display callback function
    glutMainLoop(); // Enter the GLUT event processing loop
    return 0;
}
```

GL_LINE_LOOP

```
#include <GL/glut.h>

// Function to initialize OpenGL settings
void init() {
```

```
    glClearColor(0.0, 0.0, 0.0, 0.0); // Set background color to black
    glColor3f(1.0, 1.0, 1.0);         // Set the default drawing color to white
    gluOrtho2D(0.0, 100.0, 0.0, 100.0); // Set the coordinate system
}

// Display callback function to draw a closed loop using line loop with different
colors
void kesh() {
    glClear(GL_COLOR_BUFFER_BIT); // Clear the screen

    glBegin(GL_LINE_LOOP); // Begin drawing with line loop

    // Define colors and vertices
    glColor3f(1.0, 0.0, 0.0); // Red color
    glVertex2i(30, 30); // Vertex 1

    glColor3f(0.0, 1.0, 0.0); // Green color
    glVertex2i(70, 30); // Vertex 2

    glColor3f(0.0, 0.0, 1.0); // Blue color
    glVertex2i(70, 70); // Vertex 3

    glColor3f(1.0, 1.0, 0.0); // Yellow color
    glVertex2i(30, 70); // Vertex 4

    glEnd(); // End drawing

    glFlush(); // Flush the OpenGL commands
}

// Main function
int main(int argc, char** argv) {
    glutInit(&argc, argv); // Initialize GLUT
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB); // Set display mode
    glutInitWindowSize(500, 500); // Set the window size
    glutInitWindowPosition(100, 100); // Set the window position
    glutCreateWindow("OpenGL Line Loop with Vertex Colors"); // Create the
window
    init(); // Initialize OpenGL settings
    glutDisplayFunc(kesh); // Set the display callback function
    glutMainLoop(); // Enter the GLUT event processing loop
    return 0;
}
```

GL_TRIANGLE_FAN

```
#include <GL/glut.h>

// Function to initialize OpenGL settings
void init() {
   glClearColor(0.0, 0.0, 0.0, 0.0); // Set background color to black
   gluOrtho2D(0.0, 100.0, 0.0, 100.0); // Set the coordinate system
}

// Display callback function to draw a fan shape using triangle fan with different
colors
void kesh() {
   glClear(GL_COLOR_BUFFER_BIT); // Clear the screen

   glBegin(GL_TRIANGLE_FAN); // Begin drawing with triangle fan

   // Define the central vertex
   glColor3f(1.0, 1.0, 1.0); // White color for the center
   glVertex2i(50, 50); // Central vertex

   // Define the outer vertices with different colors
   glColor3f(1.0, 0.0, 0.0); // Red color
   glVertex2i(30, 30); // Vertex 1

   glColor3f(0.0, 1.0, 0.0); // Green color
   glVertex2i(70, 30); // Vertex 2

   glColor3f(0.0, 0.0, 1.0); // Blue color
   glVertex2i(70, 70); // Vertex 3

   glColor3f(1.0, 1.0, 0.0); // Yellow color
   glVertex2i(30, 70); // Vertex 4

   glEnd(); // End drawing

   glFlush(); // Flush the OpenGL commands
}

// Main function
int main(int argc, char** argv) {
   glutInit(&argc, argv); // Initialize GLUT
   glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB); // Set display mode
   glutInitWindowSize(500, 500); // Set the window size
```

```
    glutInitWindowPosition(100, 100); // Set the window position
    glutCreateWindow("OpenGL Triangle Fan with Vertex Colors"); // Create the
window
    init(); // Initialize OpenGL settings
    glutDisplayFunc(kesh); // Set the display callback function
    glutMainLoop(); // Enter the GLUT event processing loop
    return 0;
}
```

GL_QUAD_STRIP

```
#include <GL/glut.h>

// Function to initialize OpenGL settings
void init() {
    glClearColor(0.0, 0.0, 0.0, 0.0); // Set background color to black
    gluOrtho2D(0.0, 100.0, 0.0, 100.0); // Set the coordinate system
}

// Display callback function to draw a series of quads using quad strip with different
colors
void kesh() {
    glClear(GL_COLOR_BUFFER_BIT); // Clear the screen

    glBegin(GL_QUAD_STRIP); // Begin drawing with quad strip

    // Define the vertices with different colors
    // Each quad will be defined by two vertices from each of the next two strips

    glColor3f(1.0, 0.0, 0.0); // Red color
    glVertex2i(30, 30); // Vertex 1
    glVertex2i(40, 30); // Vertex 2

    glColor3f(0.0, 1.0, 0.0); // Green color
    glVertex2i(30, 40); // Vertex 3
    glVertex2i(40, 40); // Vertex 4

    glColor3f(0.0, 0.0, 1.0); // Blue color
    glVertex2i(40, 30); // Vertex 5
    glVertex2i(50, 30); // Vertex 6

    glColor3f(1.0, 1.0, 0.0); // Yellow color
    glVertex2i(40, 40); // Vertex 7
    glVertex2i(50, 40); // Vertex 8
```

```
    glColor3f(1.0, 0.0, 1.0); // Magenta color
    glVertex2i(50, 30); // Vertex 9
    glVertex2i(60, 30); // Vertex 10

    glColor3f(0.0, 1.0, 1.0); // Cyan color
    glVertex2i(50, 40); // Vertex 11
    glVertex2i(60, 40); // Vertex 12

    glEnd(); // End drawing

    glFlush(); // Flush the OpenGL commands
}

// Main function
int main(int argc, char** argv) {
    glutInit(&argc, argv); // Initialize GLUT
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB); // Set display mode
    glutInitWindowSize(500, 500); // Set the window size
    glutInitWindowPosition(100, 100); // Set the window position
    glutCreateWindow("OpenGL Quad Strip with Vertex Colors"); // Create the
window
    init(); // Initialize OpenGL settings
    glutDisplayFunc(kesh); // Set the display callback function
    glutMainLoop(); // Enter the GLUT event processing loop
    return 0;
}
```