OpenGL Basics

> **Complete guide for Section 8 + Lab 3 coding**
> Includes: GLUT Setup, Primitives, Colors, Depth Test, 3D Transforms, Pipeline, Meshes
> **Tips**: Always `glEnable(GL_DEPTH_TEST)` in 3D. Use `glPushMatrix()/glPopMatrix()` for hierarchy. Draw order doesn't matter with depth test. Use `glDrawElements` for indexed meshes.

1. GLUT Setup & Main Loop

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

void init() {
    glClearColor(0.0, 0.0, 0.0, 1.0);     // Black background
    glEnable(GL_DEPTH_TEST);              // Critical for 3D
}

void display() {
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    // Draw scene here
    glutSwapBuffers();                    // Double buffering
}

void reshape(int w, int h) {
    glViewport(0, 0, w, h);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluPerspective(60, (float)w/h, 0.1, 100);  // Perspective
    glMatrixMode(GL_MODELVIEW);
}

int main(int argc, char** argv) {
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);
    glutInitWindowSize(800, 600);
    glutCreateWindow("OpenGL 3D Scene");

    init();
    glutDisplayFunc(display);
    glutReshapeFunc(reshape);
    glutMainLoop();
    return 0;
}
```

Key Flags:

`GLUT_DOUBLE` → Double buffering (no flicker)

`GLUT_DEPTH` → Enable depth buffer

`GLUT_RGB` → Color mode

Tip: Always clear both buffers in 3D:

```
glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
```

---

2. Primitives & Drawing

2.1 Basic Primitives

```
glBegin(GL_TRIANGLES);
    glVertex3f(-1, -1, 0);
    glVertex3f( 1, -1, 0);
    glVertex3f( 0,  1, 0);
glEnd();
```

Primitive	Use
GL_POINTS	Single pixels
GL_LINES	Disconnected segments (2 verts each)
GL_LINE_STRIP	Connected line
GL_LINE_LOOP	Closed line
GL_TRIANGLES	Separate triangles (3 verts each)
GL_TRIANGLE_STRIP	Shared vertices (efficient)
GL_TRIANGLE_FAN	First vertex shared

2.2 Line/Point Size

```
glLineWidth(3.0);
glPointSize(5.0);
glEnable(GL_POINT_SMOOTH);  // Round points
```

2.3 Circle Approximation

```
void drawCircle(float cx, float cy, float r, int segments) {
    glBegin(GL_LINE_LOOP);
    for (int i = 0; i < segments; i++) {
        float angle = 2.0 * M_PI * i / segments;
        glVertex2f(cx + r * cos(angle), cy + r * sin(angle));
    }
    glEnd();
}
```

---

3. Colors

3.1 Set Current Color

```
glColor3f(1.0, 0.0, 0.0);     // Red
glColor3ub(255, 0, 0);        // Red (byte)
glColor4f(1.0, 0.0, 0.0, 0.5); // Transparent red
```

Per-vertex coloring:

```
glBegin(GL_TRIANGLES);
    glColor3f(1,0,0); glVertex3f(-1,-1,0);
    glColor3f(0,1,0); glVertex3f( 1,-1,0);
    glColor3f(0,0,1); glVertex3f( 0, 1,0);
glEnd();
```

---

4. Occlusion & Depth Test

4.1 Painter's Algorithm (Order-Dependent)

```
// WRONG in 3D if objects intersect or cycle
drawBackObject();
drawFrontObject();  // Overwrites back
```

4.2 Depth Test (Order-Independent!)

```
glEnable(GL_DEPTH_TEST);  // Must enable!
glClear(GL_DEPTH_BUFFER_BIT);  // Clear every frame
```

How it works:
Each pixel stores closest z-value.
New pixel drawn only if closer → fixes hidden surfaces automatically.

4.3 Z-Fighting Fix

```
// 1. Better near/far
gluPerspective(60, aspect, 0.1, 100);  // Avoid 0.001 and 10000

// 2. Polygon offset (for coplanar surfaces)
```

```
glEnable(GL_POLYGON_OFFSET_FILL);
glPolygonOffset(1.0, 1.0);
```

---

5. 3D Transformations

5.1 Basic Transforms

```
glTranslatef(dx, dy, dz);
glRotatef(angle, ax, ay, az);  // Degrees, axis
glScalef(sx, sy, sz);
```

Transform	Example	Effect
glScalef(2,2,2)	Double size	
glScalef(-1,1,1)	Mirror X	
glTranslatef(5,0,0)	Move +X	
glRotatef(90,0,1,0)	90° around Y	

5.2 Hierarchy with Matrix Stack

```
void drawCube(float size) {
    glPushMatrix();
    glScalef(size, size, size);
    drawUnitCube();
    glPopMatrix();
}

void drawRobot() {
    drawTorso();

    glPushMatrix();
    glTranslatef(1, 0, 0);
    glRotatef(armAngle, 0, 0, 1);
    drawArm();
    glPopMatrix();
}
```

Critical: `glPushMatrix()` before child, `glPopMatrix()` after.
Order in code = reverse of application.

---

6. Transformation Pipeline

```
Object → [Model] → World → [View] → Eye → [Proj] → Clip → [Viewport] → Screen
```

6.1 ModelView Matrix

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
gluLookAt(eyex, eyey, eyez,
          centerx, centery, centerz,
          upx, upy, upz);
```

6.2 Projection Matrix

```
glMatrixMode(GL_PROJECTION);
glLoadIdentity();

// Perspective
glFrustum(left, right, bottom, top, near, far);
// OR
gluPerspective(fovY, aspect, near, far);

// Orthographic
glOrtho(left, right, bottom, top, near, far);
```

---

7. Meshes & Indexed Drawing

7.1 Indexed Face Set (IFS)

```
GLfloat vertices[] = {
    1,1,1,  1,1,-1,  1,-1,-1,  1,-1,1,
   -1,1,1, -1,1,-1, -1,-1,-1, -1,-1,1
};

GLfloat colors[] = { /* 8 colors */ };

GLuint indices[] = {
    0,1,2,3,  0,3,7,4,  0,4,5,1,
    6,2,1,5,  6,5,4,7,  6,7,3,2
};
```

7.2 glDrawElements (Efficient!)

```
glEnableClientState(GL_VERTEX_ARRAY);
glEnableClientState(GL_COLOR_ARRAY);

glVertexPointer(3, GL_FLOAT, 0, vertices);
glColorPointer(3, GL_FLOAT, 0, colors);

glDrawElements(GL_QUADS, 24, GL_UNSIGNED_INT, indices);

glDisableClientState(GL_VERTEX_ARRAY);
glDisableClientState(GL_COLOR_ARRAY);
```

Advantages:

- Reuse vertices (e.g., cube: 8 verts → 24 indices)
- Faster than `glBegin/glEnd`

---

8. Complete 3D Cube Example

```
void display() {
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glLoadIdentity();
    gluLookAt(3,4,5, 0,0,0, 0,1,0);  // Camera

    glRotatef(angle, 0,1,0);  // Animate

    GLfloat verts[] = { /* 8 vertices */ };
    GLfloat cols[] = { /* 8 colors */ };
    GLuint indices[] = { /* 24 indices */ };

    glEnableClientState(GL_VERTEX_ARRAY);
    glEnableClientState(GL_COLOR_ARRAY);
    glVertexPointer(3, GL_FLOAT, 0, verts);
    glColorPointer(3, GL_FLOAT, 0, cols);
    glDrawElements(GL_QUADS, 24, GL_UNSIGNED_INT, indices);
    glDisableClientState(GL_VERTEX_ARRAY);
    glDisableClientState(GL_COLOR_ARRAY);

    glutSwapBuffers();
}
```

---

9. Exam Tips & Common Mistakes

Issue	Fix
Flickering	Use GLUT_DOUBLE + glutSwapBuffers()
Objects invisible	Enable GL_DEPTH_TEST + clear depth buffer

Issue	Fix
Wrong order	Depth test fixes this
Z-fighting	Adjust near/far, use glPolygonOffset
Hierarchy broken	Mismatched Push/Pop
Flat shading	No lighting yet (next section)

10. Lab 3 Style Questions

- Draw a 3D robot with moving arms (use glPushMatrix)
- Orbiting lights around cylinder
- Hierarchical car with rotating wheels
- Indexed mesh (cube, house)
- Camera controls (gluLookAt in idle/keyboard)