3D Computer Lab

CS352:ComputerGraphics&Visualization Lab

Project Code

Course Instructor:

Dr. Somnath Dey

Submitted By:

Allu Mrudula– 200001002

Padamata Kanishka Sai–200001058

Rayala Navya Harshitha– 200003064


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

```
#include<bits/stdc++.h>
#include<SOIL/SOIL.h>
using namespace std;

GLfloat angle = 0;
GLint mouseX = 0;
GLint mouseY = 0;
int speed=0;
int d=1;
float screen_color[3]= {0, 0, 0};
double Txval=0,Tyval=0,Tzval=0;
const float windowHeight=1000, windowWidth=800;
GLfloat theta = 0.0, axis_x=0.0, axis_y=0.0;
GLboolean bRotate = false, uRotate = false;
GLfloat eyeX=0;
GLfloat eyeY=30;
GLfloat eyeZ=30;
GLfloat lookX = 0;
GLfloat lookY = 0;
GLfloat lookZ = 0;
float rotation = 0, fan_rt = 0;

bool light_switch_0=false;
bool light_switch_1=false;
bool spot_light_switch=false;
bool computer_on= false;

const float aspect_ratio = 1.0 * windowWidth / windowHeight;

GLint LoadGLTexture(const char *filename)
{
    GLuint textureID = SOIL_load_OGL_texture(
        filename,
        SOIL_LOAD_AUTO,
        SOIL_CREATE_NEW_ID,
        SOIL_FLAG_INVERT_Y
    );

    return textureID;
}
```

```
static void resize(int windowWidth, int windowHeight)
{

    //glViewport(0, 0, windowWidth, windowHeight/aspect_ratio);
    const float ar = (float) windowWidth / (float) windowHeight;

    glViewport(0, 0, windowWidth, windowHeight);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    glFrustum(-ar, ar, -1.0, 1.0, 2.0, 100.0);

    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity() ;

}

static GLfloat v_cube[8][3] =
{
    {0,0,0},
    {0,0,1},
    {0,1,0},
    {0,1,1},
    {1,0,0},
    {1,0,1},
    {1,1,0},
    {1,1,1}
};

static GLubyte quadIndices[6][4] =
{
    {3, 1, 5, 7},
    {2, 0, 4, 6},
    {2, 0, 1, 3},
    {6, 4, 5, 7},
    {2, 3, 7, 6},
    {0, 1, 5, 4}
};
```

```
static void getNormal3p(GLfloat x1, GLfloat y1, GLfloat z1, GLfloat x2,
GLfloat y2, GLfloat z2, GLfloat x3, GLfloat y3, GLfloat z3)
{
    GLfloat Ux, Uy, Uz, Vx, Vy, Vz, Nx, Ny, Nz;

    Ux = x2-x1;
    Uy = y2-y1;
    Uz = z2-z1;

    Vx = x3-x1;
    Vy = y3-y1;
    Vz = z3-z1;

    Nx = Uy*Vz - Uz*Vy;
    Ny = Uz*Vx - Ux*Vz;
    Nz = Ux*Vy - Uy*Vx;

    glNormal3f(Nx,Ny,Nz);
}

void cube(float color_red = 0.5, float color_green = 0.5, float color_blue
= 0.5)
{

    GLfloat no_mat[] = { 0.0, 0.0, 0.0, 1.0 };
    GLfloat mat_ambient[] = { color_red, color_green, color_blue, 1.0 };
    GLfloat mat_diffuse[] = { color_red, color_green, color_blue, 1.0 };
    GLfloat mat_specular[] = { 1.0, 1.0, 1.0, 1.0 };
    GLfloat mat_shininess[] = {10};
    glMaterialfv( GL_FRONT, GL_AMBIENT, mat_ambient);
    glMaterialfv( GL_FRONT, GL_DIFFUSE, mat_diffuse);
    glMaterialfv( GL_FRONT, GL_SPECULAR, mat_specular);
    glMaterialfv( GL_FRONT, GL_SHININESS, mat_shininess);

    glBegin(GL_QUADS);
    for(GLint i = 0; i<6; i++)
    {
        getNormal3p(v_cube[quadIndices[i][0]][0],
v_cube[quadIndices[i][0]][1], v_cube[quadIndices[i][0]][2],
```

```
                    v_cube[quadIndices[i][1]][0],
v_cube[quadIndices[i][1]][1], v_cube[quadIndices[i][1]][2],
                    v_cube[quadIndices[i][2]][0],
v_cube[quadIndices[i][2]][1], v_cube[quadIndices[i][2]][2]);

        for(GLint j=0; j<4; j++)
        {
            glVertex3fv(&v_cube[quadIndices[i][j]][0]);
        }
    }
    glEnd();
}

void wall_light()
{
    float length=80;
    float width=1;

    //right light
    glPushMatrix();
    glTranslatef(49,30,0);
    glScalef(1,1,10);
    glTranslatef(-0.5,-0.5,-0.5);
    cube(0.8,0.8,0.8);
    glPopMatrix();


    // left light
    glPushMatrix();
    glTranslatef(-49,30,0);
    glScalef(1,1,10);
    glTranslatef(-0.5,-0.5,-0.5);
    cube(0.8,0.8,0.8);
    glPopMatrix();

}
void wall_floor()
{
    float length1 = 100;
    float length2 = 90;
```

```
    float height= 40;
    float width = 1;

    //floor
    glPushMatrix();
    glScalef(length1,width,length2);
    glTranslatef(-0.5,-1,-0.5);
    cube(0.9,0.9,0.9);
    glPopMatrix();

    //left
    glPushMatrix();
    glTranslatef(length1/2,0,0);
    glScalef(width,height,length2);
    glTranslatef(0,0,-0.5);
    cube(0.9,0.9,0.9);
    glPopMatrix();

    // right
    glPushMatrix();
    glTranslatef(-length1/2,0,0);
    glScalef(width,height,length2);
    glTranslatef(0,0,-0.5);
    cube(0.9,0.9,0.9);
    glPopMatrix();

    // Up
    glPushMatrix();
    glTranslatef(0,height,0);
    glScalef(length1,width,length2);
    glTranslatef(-0.5,0,-0.5);
    cube(0.9,0.9,0.9);
    glPopMatrix();

    // Front
    glPushMatrix();
    glTranslatef(0,0,-length2/2);
    glScalef(length1,height,width);
    glTranslatef(-0.5,0,0);
    cube(0.9,0.9,0.9);
```

```
    glPopMatrix();

    //back
    glPushMatrix();
    glTranslatef(0,0,length2/2);
    glScalef(length1,height,width);
    glTranslatef(-0.5,0,0);
    cube(0.9,0.9,0.9);
    glPopMatrix();

    // Black board
    glPushMatrix();
    glTranslatef(0,height/3,-(length2/2 -1));
    glScalef(length1/3,height/3,width);
    glTranslatef(-0.5,0,0);
    cube(0.0,0.0,0.0);
    glPopMatrix();

    // GLuint texture = LoadGLTexture("Windows.png");
    // glEnable(GL_TEXTURE_2D);
    // glBindTexture( GL_TEXTURE_2D, texture );
    // glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T,
GL_CLAMP_TO_EDGE);
    // glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
    // glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);

    // glBegin(GL_QUADS);
    //     glTexCoord2f(0.0f, 0.0f);glVertex3f(-16.66, height/3,
-(length2/2 -1)+10);
    //     glTexCoord2f(1.0f, 0.0f);glVertex3f(16.66, height/3,
-(length2/2 -1)+10);
    //     glTexCoord2f(1.0f, 1.0f);glVertex3f(16.66, 2*height/3,
-(length2/2 -1)+10);
    //     glTexCoord2f(0.0f, 1.0f);glVertex3f(-16.66, 2*height/3,
-(length2/2 -1)+10);
    // glEnd();

    // glDisable(GL_TEXTURE_2D);
}
```

```
void chair()
{

    float length=20;
    float width=1;

    //base seat
    glPushMatrix();
    glTranslatef(0,length/2,0);
    glScalef(length,width,length);
    glTranslatef(-0.5,-0.5,-0.5);
    cube(0,0,1);
    glPopMatrix();

    // leg base 1
    glPushMatrix();
    glTranslatef(length/2 -width/2,0,length/2-width/2);
    glScalef(width,length,width);
    glTranslatef(-0.5,-0.5,-0.5);
    cube(0,0,0);
    glPopMatrix();
    // leg base 2
    glPushMatrix();
    glTranslatef(length/2 -width/2,0,- length/2 +width/2);
    glScalef(width,length,width);
    glTranslatef(-0.5,-0.5,-0.5);
    cube(0,0,0);
    glPopMatrix();
    // leg base 3
    glPushMatrix();
    glTranslatef(-length/2 +width/2,0,+ length/2 -width/2);
    glScalef(width,length,width);
    glTranslatef(-0.5,-0.5,-0.5);
    cube(0,0,0);
    glPopMatrix();
    // leg  base 4
    glPushMatrix();
    glTranslatef(-length/2 +width/2,0,- length/2 +width/2);
    glScalef(width,length,width);
```

```
    glTranslatef(-0.5,-0.5,-0.5);
    cube(0,0,0);
    glPopMatrix();

    // upper  1
    glPushMatrix();
    glTranslatef(length/2 -width/2,length,length/2-width/2);
    glScalef(width,length,width);
    glTranslatef(-0.5,-0.5,-0.5);
    cube(0,0,0);
    glPopMatrix();
    // upper 2
    glPushMatrix();
    glTranslatef(-length/2 -width/2,length,length/2+width/2);
    glScalef(width,length,width);
    glTranslatef(-0.5,-0.5,-0.5);
    cube(0,0,0);
    glPopMatrix();

    //support
    glPushMatrix();
    glTranslatef(0,length+5,length/2);
    glScalef(length,length/2,0);
    glTranslatef(-0.5,-0.5,-0.5);
    cube(0,0,1);
    glPopMatrix();

}

void computer(){


    // GLuint texture = LoadGLTexture("Windows.jpg");
    // glEnable(GL_TEXTURE_2D);
    // glBindTexture( GL_TEXTURE_2D, texture );
    // glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T,
GL_CLAMP_TO_EDGE);
    // glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
    // glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
```

```
    // glBegin(GL_QUADS);
    //     glTexCoord2f(0.0f, 0.0f);glVertex3f(-7, 14.5, -3);
    //     glTexCoord2f(1.0f, 0.0f);glVertex3f(7, 14.5, -3);
    //     glTexCoord2f(1.0f, 1.0f);glVertex3f(7, 23.5, -3);
    //     glTexCoord2f(0.0f, 1.0f);glVertex3f(-7, 23.5, -3);
    // glEnd();

    // glDisable(GL_TEXTURE_2D);

    //desktop
    glPushMatrix();
    glTranslatef(0,19,-5);
    glScalef(15,10,1);
    glTranslatef(-0.5,-0.5,-0.5);
    cube();
    glPopMatrix();

    //screen
    glPushMatrix();
    glTranslatef(0,19,-4.495);
    glScalef(14,9,0.01);
    glTranslatef(-0.5,-0.5,-0.5);
    cube(screen_color[0], screen_color[1], screen_color[2]);
    glPopMatrix();

    //screen edge left
    glPushMatrix();
    glTranslatef(-7.25,19,-4.4);
    glScalef(0.5,9,0.2);
    glTranslatef(-0.5,-0.5,-0.5);
    cube();
    glPopMatrix();

    //screen edge right
    glPushMatrix();
    glTranslatef(7.25,19,-4.4);
    glScalef(0.5,9,0.2);
    glTranslatef(-0.5,-0.5,-0.5);
    cube();
    glPopMatrix();
```

```
//screen edge up
glPushMatrix();
glTranslatef(0,23.75,-4.4);
glScalef(15,0.5,0.2);
glTranslatef(-0.5,-0.5,-0.5);
cube();
glPopMatrix();

//screen edge down
glPushMatrix();
glTranslatef(0,14.25,-4.4);
glScalef(15,0.5,0.2);
glTranslatef(-0.5,-0.5,-0.5);
cube();
glPopMatrix();

//stand
glPushMatrix();
glTranslatef(0,12.6,-5);
glScalef(2,2.8,0.5);
glTranslatef(-0.5,-0.5,-0.5);
cube();
glPopMatrix();

//stand base
glPushMatrix();
glTranslatef(0,11.1,-5);
glScalef(4,0.2,4);
glTranslatef(-0.5,-0.5,-0.5);
cube();
glPopMatrix();

//keyboard
glPushMatrix();
glTranslatef(-1,11.25,3);
glScalef(12,0.5,5);
glTranslatef(-0.5,-0.5,-0.5);
cube();
glPopMatrix();
```

```
for(int i=0; i<18; i++){
    for(int j=0; j<6; j++){
        glPushMatrix();
        glTranslatef(-6.1+ 0.6*i,12.1,5- 0.6*j);
        glScalef(0.5,0.2,0.5);
        glTranslatef(-0.5,-0.5,-0.5);
        cube();
        glPopMatrix();
    }
}

//CPU
glPushMatrix();
glTranslatef(5.5,-5,5);
glScalef(5,10,8);
glTranslatef(-0.5,-0.5,-0.5);
cube();
glPopMatrix();

//mouse base
glPushMatrix();
glTranslatef(7,11.3,3);
glScalef(2,0.6,3);
glTranslatef(-0.5,-0.5,-0.5);
cube();
glPopMatrix();

//mouse upper base
glPushMatrix();
glTranslatef(7,11.8,3.6);
glScalef(2,0.4,1.8);
glTranslatef(-0.5,-0.5,-0.5);
cube();
glPopMatrix();

//mouse left click
glPushMatrix();
glTranslatef(6.45,11.8,2.05);
glScalef(0.9,0.4,1.1);
```

```
    glTranslatef(-0.5,-0.5,-0.5);
    cube();
    glPopMatrix();

    //mouse right click
    glPushMatrix();
    glTranslatef(7.55,11.8,2.05);
    glScalef(0.9,0.4,1.1);
    glTranslatef(-0.5,-0.5,-0.5);
    cube();
    glPopMatrix();

    //mouse scroll
    glPushMatrix();
    glTranslatef(7,11.85,2);
    glScalef(0.2,0.5,0.4);
    glTranslatef(-0.5,-0.5,-0.5);
    cube();
    glPopMatrix();

    //mouse cable
    glPushMatrix();
    glTranslatef(7,11.1,1.5);
    glRotatef(75,0,1,0);
    glScalef(11,0.2,0.2);
    cube();
    glPopMatrix();


    //CPU cable
    glPushMatrix();
    glTranslatef(4.5,-2,1);
    glRotatef(75,1,0,0);
    glRotatef(90,0,1,0);
    glScalef(13,0.2,0.2);
    cube();
    glPopMatrix();

    //keyboard cable
    glPushMatrix();
```

```
    glTranslatef(3,11.1,0.5);
    glRotatef(60,0,1,0);
    glScalef(11,0.2,0.2);
    cube();
    glPopMatrix();

    // left switch cable vertical
    glPushMatrix();
    glTranslatef(5.66, 11, -9.1);
    glScalef(0.2, 2, 0.2);
    cube();
    glPopMatrix();

    //desktop to plug cable
    glPushMatrix();
    glTranslatef(5.66, 11, -9.1);
    glRotatef(-50, 0, 1, 0);
    glScalef(0.2, 0.2, 5);
    cube();
    glPopMatrix();
}

void table()
{
    float length=20;
    float width=1;

    //base
    glPushMatrix();
    glTranslatef(0,length/2,0);
    glScalef(length,width+1,length);
    glTranslatef(-0.5,-0.5,-0.5);
    cube(1.0,0.992,0.816);
    glPopMatrix();

    // base 1
    glPushMatrix();
    glTranslatef(length/2 -width/2,0,length/2-width/2);
    glScalef(width,length,width);
    glTranslatef(-0.5,-0.5,-0.5);
```

```
    cube(1.0,0.992,0.816);
    glPopMatrix();

    // base 2
    glPushMatrix();
    glTranslatef(length/2 -width/2,0,- length/2 +width/2);
    glScalef(width,length,width);
    glTranslatef(-0.5,-0.5,-0.5);
    cube(1.0,0.992,0.816);
    glPopMatrix();

    // base 3
    glPushMatrix();
    glTranslatef(-length/2 +width/2,0,+ length/2 -width/2);
    glScalef(width,length,width);
    glTranslatef(-0.5,-0.5,-0.5);
    cube(1.0,0.992,0.816);
    glPopMatrix();

    // base 4
    glPushMatrix();
    glTranslatef(-length/2 +width/2,0,- length/2 +width/2);
    glScalef(width,length,width);
    glTranslatef(-0.5,-0.5,-0.5);
    cube(1.0,0.992,0.816);
    glPopMatrix();

    //side1
    glPushMatrix();
    glTranslatef(-9.75,18.5,0);
    glScalef(0.5,15,20);
    glTranslatef(-0.5,-0.5,-0.5);
    cube(1.0,0.992,0.816);
    glPopMatrix();

    //side2
    glPushMatrix();
    glTranslatef(9.75,18.5,0);
    glScalef(0.5,15,20);
    glTranslatef(-0.5,-0.5,-0.5);
```

```
    cube(1.0,0.992,0.816);
    glPopMatrix();

    //back
    glPushMatrix();
    glTranslatef(0,18.5,-9.75);
    glScalef(19,15,0.5);
    glTranslatef(-0.5,-0.5,-0.5);
    cube(1.0,0.992,0.816);
    glPopMatrix();

    //black back
    glPushMatrix();
    glTranslatef(0,13.5,-9.4);
    glScalef(19,5,0.2);
    glTranslatef(-0.5,-0.5,-0.5);
    cube();
    glPopMatrix();

    //switch board
    glPushMatrix();
    glTranslatef(6,13.5,-9.25);
    glScalef(5,3,0.1);
    glTranslatef(-0.5,-0.5,-0.5);
    cube(1,1,1);
    glPopMatrix();

     // left plug
    glPushMatrix();
    glTranslatef(4.25, 13.5, -9.1);
    glScalef(1, 1.5, 0.2);
    glTranslatef(-0.5, -0.5, -0.5);
    cube();
    glPopMatrix();

    // socket upper hole
    glPushMatrix();
    glTranslatef(7.75, 14, -9.195);
    glScalef(0.2, 0.3, 0.01);
    glTranslatef(-0.5, -0.5, -0.5);
```

```
    cube();
    glPopMatrix();

    // socket left hole
    glPushMatrix();
    glTranslatef(7.5, 13, -9.195);
    glScalef(0.2, 0.3, 0.01);
    glTranslatef(-0.5, -0.5, -0.5);
    cube();
    glPopMatrix();

    // socket left hole
    glPushMatrix();
    glTranslatef(8, 13, -9.195);
    glScalef(0.2, 0.3, 0.01);
    glTranslatef(-0.5, -0.5, -0.5);
    cube();
    glPopMatrix();

    // left switch
    glPushMatrix();
    glTranslatef(5.25, 13.5, -9.125);
    glScalef(0.5, 1, 0.15);
    glTranslatef(-0.5, -0.5, -0.5);
    cube(1, 1, 1);
    glPopMatrix();

    // right switch
    glPushMatrix();
    glTranslatef(6.75, 13.5, -9.125);
    glScalef(0.5, 1, 0.2);
    glTranslatef(-0.5, -0.5, -0.5);
    cube(1, 1, 1);
    glPopMatrix();

    // middle switch
    glPushMatrix();
    glTranslatef(6, 13.5, -9.125);
    glScalef(0.5, 1, 0.2);
    glTranslatef(-0.5, -0.5, -0.5);
```

```
    cube(1, 1, 1);
    glPopMatrix();

    // cable
}

void fan_rotation()
{
    fan_rt = fan_rt+ speed;
    if(fan_rt>360)
        fan_rt =0;
    glutPostRedisplay();
}

void fan()
{
    float base = 8;

    glPushMatrix();
    glRotatef(fan_rt,0,1,0);

    //head
    int head_rot = 0;
    for(int i=0; i<100; i++)
    {
        glPushMatrix();
        glTranslatef(0,base/4,0);
        glRotatef(head_rot,0,1,0);
        glScalef(base/4,base*2,base/4);
        glTranslatef(-0.5,0.0,-0.5);
        cube(1.000, 0.5, 0.5);
        glPopMatrix();
        head_rot+=5;
    }

    //round-base
    int base_rot = 0;
    for (int i=0; i<100; i++)
    {
        glPushMatrix();
```

```
        glRotatef(base_rot,0,1,0);
        glScalef(base,base/4,base);
        glTranslatef(-0.5,0.0,-0.5);
        cube(0.8, 0.3, 0.3);
        glPopMatrix();
        base_rot+=5;
    }
    //fan-leg
    for(int i=0; i<=2; i++)
    {
        glPushMatrix();
        glRotatef(120*i,0,1,0);
        glTranslatef((2*base)/2+base/2,base/8,0);
        glScalef(2*base,0.002*base,base/2);
        glTranslatef(-0.5,0.0,-0.5);
        cube(0.000, 0.000, 0.545);
        glPopMatrix();
    }

    fan_rotation();
    glPopMatrix();
}

void light_function_0(float x, float y, float z)
{
    // Light Specification
    GLfloat no_light[] = { 0.0, 0.0, 0.0, 1.0 };
    GLfloat light_ambient[] = {0.1, 0.1, 0.1, 1.0};
    GLfloat light_diffuse[] = { 0.8, 0.8, 0.8, 1 };
    GLfloat light_specular[] = { 1, 1, 1, 1 };
    GLfloat light_position[] = { x, y, z, 1.0 };

    glEnable( GL_LIGHT0);
    if (light_switch_0)
    {
        glLightfv( GL_LIGHT0, GL_AMBIENT, light_ambient);
        glLightfv( GL_LIGHT0, GL_DIFFUSE, light_diffuse);
        glLightfv( GL_LIGHT0, GL_SPECULAR, light_specular);

    }
```

```
    else
    {
        glLightfv( GL_LIGHT0, GL_AMBIENT, no_light);
        glLightfv( GL_LIGHT0, GL_DIFFUSE, no_light);
        glLightfv( GL_LIGHT0, GL_SPECULAR, no_light);

    }


    glLightfv( GL_LIGHT0, GL_POSITION, light_position);

}

void light_function_1(float x, float y, float z)
{
    // Light Specification
    GLfloat no_light[] = { 0.0, 0.0, 0.0, 1.0 };
    GLfloat light_ambient[] = {0.1, 0.1, 0.1, 1.0};
    GLfloat light_diffuse[] = { 0.8, 0.8, 0.8, 1 };
    GLfloat light_specular[] = { 1, 1, 1, 1 };
    GLfloat light_position[] = { x, y, z, 1.0 };

    glEnable( GL_LIGHT1);
    if (light_switch_1)
    {
        glLightfv( GL_LIGHT1, GL_AMBIENT, light_ambient);
        glLightfv( GL_LIGHT1, GL_DIFFUSE, light_diffuse);
        glLightfv( GL_LIGHT1, GL_SPECULAR, light_specular);

    }
    else
    {
        glLightfv( GL_LIGHT1, GL_AMBIENT, no_light);
        glLightfv( GL_LIGHT1, GL_DIFFUSE, no_light);
        glLightfv( GL_LIGHT1, GL_SPECULAR, no_light);

    }


    glLightfv( GL_LIGHT1, GL_POSITION, light_position);
```

```
}


void spot_light_function(float x, float y, float z)
{
    // Light Specification
    GLfloat no_light[] = { 0.0, 0.0, 0.0, 1.0 };
    GLfloat light_ambient[] = {0.5, 0.5, 0.5, 1.0};
    GLfloat light_diffuse[] = { 0.0, 1.0, 0.0, 1 };
    GLfloat light_specular[] = { 1, 1, 1, 1 };
    GLfloat light_position[] = { x, y, z, 1.0 };

    glEnable( GL_LIGHT2);
    if (spot_light_switch)
    {
        glLightfv( GL_LIGHT2, GL_AMBIENT, light_ambient);
        glLightfv( GL_LIGHT2, GL_DIFFUSE, light_diffuse);
        glLightfv( GL_LIGHT2, GL_SPECULAR, light_specular);

    }
    else
    {
        glLightfv( GL_LIGHT2, GL_AMBIENT, no_light);
        glLightfv( GL_LIGHT2, GL_DIFFUSE, no_light);
        glLightfv( GL_LIGHT2, GL_SPECULAR, no_light);

    }


    glLightfv( GL_LIGHT2, GL_POSITION, light_position);
    GLfloat direction[]= {0,-1,0,1};
    GLfloat cut_off=60;
    glLightfv(GL_LIGHT2,GL_SPOT_DIRECTION,direction);
    glLightf(GL_LIGHT2,GL_SPOT_CUTOFF,cut_off);

}
void display_setting()
{
    glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
```

```
    glMatrixMode( GL_PROJECTION );
    glLoadIdentity();
    // xmin, xmax, ymin, ymax, near, far
    glFrustum(-5,5,-5,5, 4, 100);

    glMatrixMode( GL_MODELVIEW );
    glLoadIdentity();
    // eye, look, head
    gluLookAt(eyeX,eyeY,eyeZ, lookX,lookY,lookZ, 0,1,0);

    //glViewport(0, 0, windowHeight, windowWidth);
    glRotatef(theta,axis_x,axis_y,0);
    glTranslatef(0,0,Tzval);
}

void image(){
    glColor3f(1,1,1);
        GLuint texture = LoadGLTexture("Windows.png");
        glEnable(GL_TEXTURE_2D);
        glBindTexture( GL_TEXTURE_2D, texture );
        glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T,
GL_CLAMP_TO_EDGE);
        glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
        glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);

        glBegin(GL_QUADS);
            glTexCoord2f(0.0f, 0.0f);glVertex3f(-7, 14.5, -3);
            glTexCoord2f(1.0f, 0.0f);glVertex3f(7, 14.5, -3);
            glTexCoord2f(1.0f, 1.0f);glVertex3f(7, 23.5, -3);
            glTexCoord2f(0.0f, 1.0f);glVertex3f(-7, 23.5, -3);
        glEnd();

        glDisable(GL_TEXTURE_2D);
}

void display(void)
{
    display_setting();
```

```
    // glPushMatrix();
    // glTranslatef(0,-1,0);
    wall_floor();
    // GLuint texture = LoadGLTexture("Windows.png");
    // glEnable(GL_TEXTURE_2D);
    // glBindTexture( GL_TEXTURE_2D, texture );
    // glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T,
GL_CLAMP_TO_EDGE);
    // glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
    // glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);

    // glBegin(GL_QUADS);
    //     glTexCoord2f(0.0f, 0.0f);glVertex3f(-16.66, 13.33, -34);
    //     glTexCoord2f(1.0f, 0.0f);glVertex3f(16.66, 13.33, -34);
    //     glTexCoord2f(1.0f, 1.0f);glVertex3f(16.66, 26.66, -34);
    //     glTexCoord2f(0.0f, 1.0f);glVertex3f(-16.66, 26.66, -34);
    // glEnd();

    // glDisable(GL_TEXTURE_2D);
    // glPopMatrix();

    wall_light();
    glPushMatrix();
    glTranslatef(-20,34.5,0);
    glScalef(0.4,0.4,0.4);
    fan();
    glPopMatrix();

    glPushMatrix();
    glTranslatef(20,34.5,0);
    glScalef(0.4,0.4,0.4);
    fan();
    glPopMatrix();

    // left light
    glPushMatrix();
    light_function_0(-23,25,0);
    // glTranslatef(-23,-20,0);
    // glScalef(2,2,2);
    // glTranslatef(-0.5,-0.5,-0.5);
```

```
    //cube(1.0,0.0,0.0);
    glPopMatrix();

    // right light
    glPushMatrix();
    light_function_1(23,25,0);
    // glTranslatef(23,-20,0);
    // glScalef(2,2,2);
    // glTranslatef(-0.5,-0.5,-0.5);
    //cube(1.0,0.0,0.0);
    glPopMatrix();

    //Spot light
    glPushMatrix();
    spot_light_function(0,40,-10);
    glTranslatef(0,40,-10);
    glScalef(2,2,2);
    glTranslatef(-0.5,-0.5,-0.5);
    //cube(0.0,1.0,0.0);
    glPopMatrix();

    d=1;
    // First table chair
    for(int i=-20; i<=30; i+=8)
    {

        glPushMatrix();
        glTranslatef(-30,2,i);
        glRotatef(90,0,1,0);
        glScalef(0.2,0.2,0.2);
        chair();
        glPopMatrix();

        glPushMatrix();
        glTranslatef(-35,3,i);
        glRotatef(90,0,1,0);
        glScalef(0.4,0.3,0.3);
        table();
        glPopMatrix();
```

```
        glPushMatrix();
        glTranslatef(-35,3,i);
        glRotatef(90,0,1,0);
        glScalef(0.3,0.3,0.3);
        computer();
        // image();
        glPopMatrix();
    }

    // glColor3f(1,1,1);
    // GLuint texture = LoadGLTexture("CSELAB.png");
    // glEnable(GL_TEXTURE_2D);
    // glBindTexture( GL_TEXTURE_2D, texture );
    // glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T,
GL_CLAMP_TO_EDGE);
    // glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
    // glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);

    // glBegin(GL_QUADS);
    //     glTexCoord2f(0.0f, 0.0f);glVertex3f(-5, 17, -42.99);
    //     glTexCoord2f(1.0f, 0.0f);glVertex3f(5, 17, -42.99);
    //     glTexCoord2f(1.0f, 1.0f);glVertex3f(5, 23, -42.99);
    //     glTexCoord2f(0.0f, 1.0f);glVertex3f(-5, 23, -42.99);
    // glEnd();

    // glDisable(GL_TEXTURE_2D);

    for(int i=-20; i<=30; i+=8)
    {
        glPushMatrix();
        glTranslatef(-10,2,i);
        glRotatef(-90,0,1,0);
        glScalef(0.2,0.2,0.2);
        chair();
        glPopMatrix();

        glPushMatrix();
        glTranslatef(-5,3,i);
        glRotatef(-90,0,1,0);
        glScalef(0.4,0.3,0.3);
```

```
        table();
        glPopMatrix();

        glPushMatrix();
        glTranslatef(-5,3,i);
        glRotatef(-90,0,1,0);
        glScalef(0.3,0.3,0.3);
        computer();

        // glColor3f(1,1,1);
        // GLuint texture = LoadGLTexture("Windows.png");
        // glEnable(GL_TEXTURE_2D);
        // glBindTexture( GL_TEXTURE_2D, texture );
        // glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T,
GL_CLAMP_TO_EDGE);
        // glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER,
GL_LINEAR);
        // glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER,
GL_LINEAR);

        // glBegin(GL_QUADS);
        //     glTexCoord2f(0.0f, 0.0f);glVertex3f(-7, 14.5, -3);
        //     glTexCoord2f(1.0f, 0.0f);glVertex3f(7, 14.5, -3);
        //     glTexCoord2f(1.0f, 1.0f);glVertex3f(7, 23.5, -3);
        //     glTexCoord2f(0.0f, 1.0f);glVertex3f(-7, 23.5, -3);
        // glEnd();

        // glDisable(GL_TEXTURE_2D);
        glPopMatrix();
    }

    for(int i=-20; i<=30; i+=8)
    {
        glPushMatrix();
        glTranslatef(10,2,i);
        glRotatef(90,0,1,0);
        glScalef(0.2,0.2,0.2);
        chair();
        glPopMatrix();
```

```
        glPushMatrix();
        glTranslatef(5,3,i);
        glRotatef(90,0,1,0);
        glScalef(0.4,0.3,0.3);
        table();
        glPopMatrix();

        glPushMatrix();
        glTranslatef(5,3,i);
        glRotatef(90,0,1,0);
        glScalef(0.3,0.3,0.3);
        computer();
        glPopMatrix();
    }

    // Second table chair
    for(int i=-20; i<=30; i+=8)
    {
        glPushMatrix();
        glTranslatef(30,2,i);
        glRotatef(-90,0,1,0);
        glScalef(0.2,0.2,0.2);
        chair();
        glPopMatrix();

        glPushMatrix();
        glTranslatef(35,3,i);
        glRotatef(-90,0,1,0);
        glScalef(0.4,0.3,0.3);
        table();
        glPopMatrix();

        glPushMatrix();
        glTranslatef(35,3,i);
        glRotatef(-90,0,1,0);
        glScalef(0.3,0.3,0.3);
        computer();
        glPopMatrix();
    }
```

```
    glFlush();
    glutSwapBuffers();

}

static void key(unsigned char key, int x, int y)
{
    switch (key)
    {
    case 27 :
    case 'q':
        exit(0);
        break;

    case 'a': // left
        eyeX--;
        lookX--;
        break;
    case 'd': // right
        eyeX++;
        lookX++;
        break;
    case 's': // down
        eyeY--;
        lookY--;
        break;
    case 'w': // up
        eyeY++;
        lookY++;
        break;
    case '1':
        light_switch_0 =! light_switch_0;
        break;
    case '2':
        light_switch_1 =! light_switch_1;
        break;
    case '3':
        spot_light_switch =! spot_light_switch;
        break;
```

```
    case '+': // zoom in
        eyeZ--;
        lookZ--;
        break;
    case '-': // zoom out
        eyeZ++;
        lookZ++;
        break;
    case 'i':
        if(speed<10){
            speed++;
        }
        break;
    case 'u':
        if(speed>0){
            speed--;;
        }
        break;
    case 'o':
        if(!computer_on){
            screen_color[0]= 0;
            screen_color[1]= 1;
            screen_color[2]= 1;
            computer_on= true;
        }
        else{
            screen_color[0]= 0;
            screen_color[1]= 0;
            screen_color[2]= 0;
            computer_on= false;
        }
        break;
    }
    glutPostRedisplay();
}

void motion(int x, int y) {
    GLfloat dx = x - mouseX;
    GLfloat dy = y - mouseY;
```

```
    axis_y=1;
    theta+= dx;

    eyeY-= dy*0.08;

    mouseX = x;
    mouseY = y;

    glutPostRedisplay();
}

int main(int argc, char**argv)
{
    glutInit(&argc,argv);
    glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);
    glutInitWindowPosition(200,200);
    glutInitWindowSize(windowHeight,windowWidth);
    glutCreateWindow("Computer Lab");


    cout<<"--------------------------"<<endl;
    cout<<"--------------------------"<<endl;

    cout<<"Press : 1 -> Left Light ON"<<endl;
    cout<<"Press : 2 -> Right Light ON"<<endl;
    cout<<"Press : 3 -> Spotlight Light ON"<<endl;

    cout<<"--------------------------"<<endl;
    cout<<"--------------------------"<<endl;

    cout<<"Press : w -> move Up"<<endl;
    cout<<"Press : s -> move Down"<<endl;
    cout<<"Press : a -> move left"<<endl;
    cout<<"Press : d -> move Right"<<endl;
    cout<<"Press : + -> Zoom In"<<endl;
    cout<<"Press : - -> Zoom Out"<<endl;

    cout<<"--------------------------"<<endl;
    cout<<"--------------------------"<<endl;
```

```
    cout<<"Press : i -> increase fan speed"<<endl;
    cout<<"Press : u -> decrease fan speed"<<endl;
    cout<<"Press : o -> Turn on / off the computers"<<endl;


    glutDisplayFunc(display);
    glutKeyboardFunc(key);
    glutMotionFunc(motion);
    glutReshapeFunc(resize);

    glShadeModel(GL_SMOOTH);
    glEnable(GL_DEPTH_TEST);
    glEnable(GL_NORMALIZE);
    glEnable(GL_LIGHTING);
    glEnable(GL_BLEND);

    glutMainLoop();
}
```