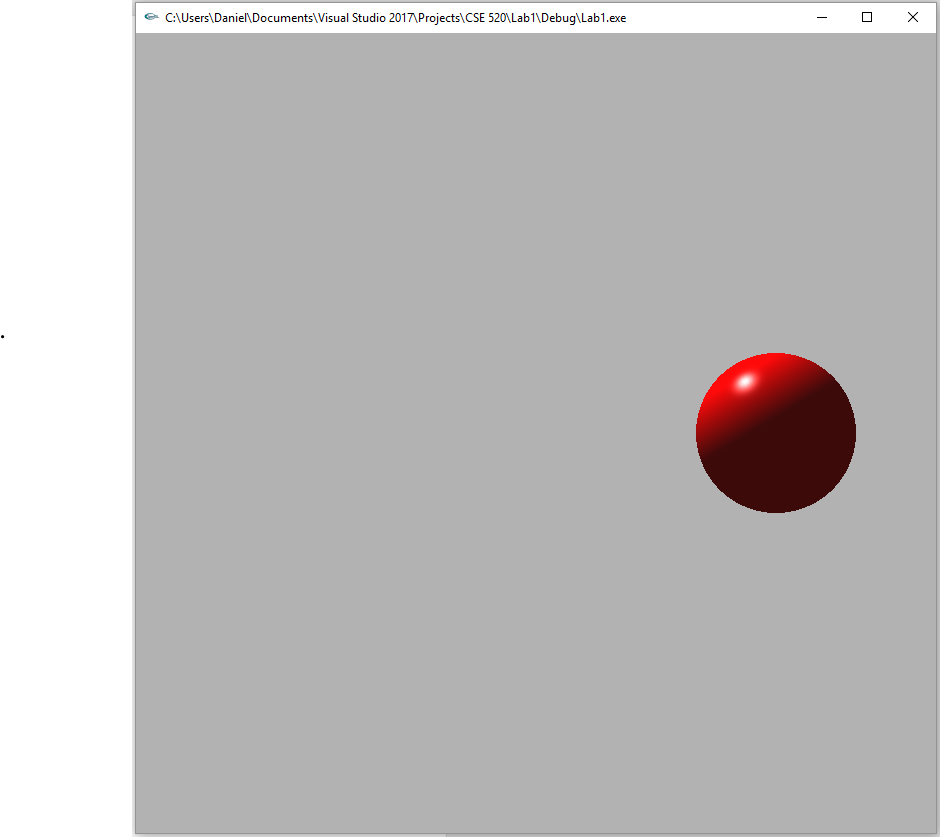
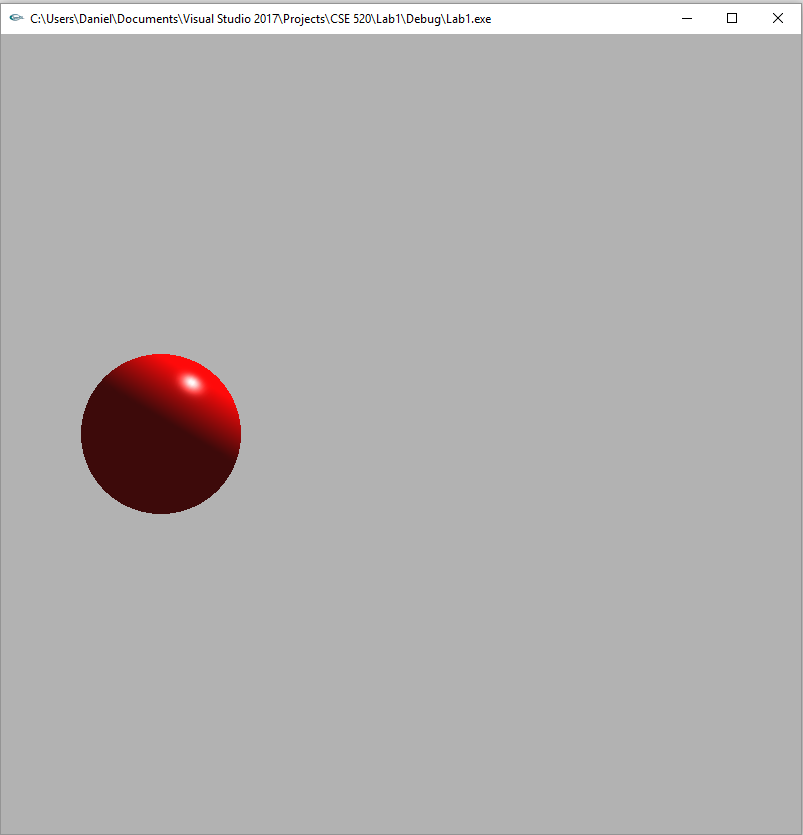
Daniel Meyer

CSE 520-01

Lab 1

Animation and Lighting

**Lab 1 Report**



#include <stdlib.h>

#include <stdio.h>

#include <GL/glut.h>

double ax = 0;

bool moveRight = true;

//initialization

void init(void)

{

GLfloat no\_mat[] = { 0.0, 0.0, 0.0, 1.0 };

GLfloat mat\_specular[] = { 1.0, 1.0, 1.0, 1.0 };

GLfloat diffuseMaterial[4] = { 1.0, 1.0, 1.0, 1.0 };

GLfloat ambient[] = { 1.0, 0.0, 0.0, 1.0 };

GLfloat light[] = { 1.0, 1.0, 1.0 };

GLfloat light\_position0[] = { 0.0, 1.0, 0.0, 1.0 };

GLfloat spot\_direction0[] = { 0.0, 0.0, 0.0 };

glClearColor(0.7, 0.7, 0.7, 0.0);

glShadeModel(GL\_SMOOTH);

glEnable(GL\_DEPTH\_TEST);

glMaterialfv(GL\_FRONT, GL\_DIFFUSE, diffuseMaterial);

glMaterialfv(GL\_FRONT, GL\_SPECULAR, mat\_specular);

glMaterialf(GL\_FRONT, GL\_SHININESS, 100.0);

glLightfv(GL\_LIGHT0, GL\_AMBIENT, ambient);

glLightfv(GL\_LIGHT0, GL\_DIFFUSE, light);

glLightfv(GL\_LIGHT0, GL\_SPECULAR, light);

glLightfv(GL\_LIGHT0, GL\_POSITION, light\_position0);

glLightfv(GL\_LIGHT0, GL\_SPOT\_DIRECTION, spot\_direction0);

glEnable(GL\_LIGHTING);

glEnable(GL\_LIGHT0);

glColorMaterial(GL\_FRONT, GL\_DIFFUSE);

glEnable(GL\_COLOR\_MATERIAL);

//Orthographic 3D

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

glOrtho(-5.0, 5.0, -5.0, 5.0, 0.1, 100);

}

void display(void)

{

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);

glLoadIdentity();

gluLookAt(0, 0, 0, 0, 0, 1, 0, 1, 0); //Looking along z-axis w/ y-axis being upward

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

glPushMatrix();

glColor3f(1, 0, 0);

glTranslatef(ax, 0, 0);

glutSolidSphere(0.2, 100, 100);

glPopMatrix();

glFlush(); //send all output to screen

}

void animate()

{

if (ax == 0.7)

{

moveRight = false;

}

else if (ax == -0.7)

{

moveRight = true;

}

if (moveRight == true)

{

ax += 0.1;

}

else

{

ax -= 0.1;

}

glutPostRedisplay();

}

void timerHandle(int value)

{

animate();

glutPostRedisplay();

glutTimerFunc(100, timerHandle, 0);

}

void visHandle(int visible)

{

if (visible == GLUT\_VISIBLE)

timerHandle(0);

else

;

}

int main(int argc, char\*\* argv)

{

glutInit(&argc, argv); //initialize toolkit

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB | GLUT\_DEPTH); //Lighting display mode

glutInitWindowSize(800, 800); //set window size on screen

glutInitWindowPosition(100, 150); //set window position on screen

glutCreateWindow(argv[0]); //open screen widow

init();

glutDisplayFunc(display); //points to display function

glutVisibilityFunc(visHandle);

glutMainLoop(); //go into perpetual loop

return 0;

}

**Summary:**

The first assignment for this class is designed as a review of the concepts from CSE 420 and requires lighting and animating a glutSolidSphere(). For the lighting I positioned a spot light above the origin where the sphere is placed and pointed it at the origin. To animate the sphere I used visHandle() and timerHandle() along with an animate() function. The animate() function checks a Boolean value that is initialized to true to see if the sphere needs to move to the right, if it is false it will move to the left. Inside of animate I set a maximum and minimum x value and the sphere is translated in the display() function. Overall, the program compiles and runs without error and contains the desired features. Thus, I believe I have earned the full 20 points for this assignment.