Computer Graphics (UCS505)

Project on

The Lighthouse

Submitted by

Ishaan Chhabra 102183053

Sanyam Sood 102003163

Parth Nasa 102003147

COE 7

B.E. Third Year COE

Submitted to:

Dr. Anupam Garg

Computer Science and Engineering Department

Thapar Institute of Engineering & Technology

(Deemed to be University), Patiala, Punjab - 147004

India

Table of Contents

Introduction to the Project

Movies. Who doesn't like movies? Even in our adulthood, we get amazed at scenes like waves hitting against a lighthouse on a stormy night. Inspired by such fascinating visual experiences, our group presents a unique Computer Graphics project – The Lighthouse.

In this project, we have built a lighthouse along the sea shore which works in a simulation of patrolling and guiding in the night and being the centre of attraction for people visiting the beach.

We have used polygons to make the structure of the lighthouse, land, sea and sky. To give the sea a flowing effect at the shore, we made the corner point of the sea polygon move in a sin wave.

We have made a bright yellow sun for the day using Bresenham Circle Drawing Algorithm and filled it with subsequent circles of decreasing radii. For the night we have beautified our sky with twinkling stars which we have simply achieved by changing the colour of pixels at the time of refreshing our buffer.

As for our pride, the guiding light of our lighthouse is another polygon originating from the hole in the lighthouse and translating back and forth continuously with increasing gap between the end points of light at its farthest and decreasing the gap at its nearest.

With this basic idea, let us dive into the actual working and concepts used to make our imagination come true.

Computer Graphics concepts used

OpenGL and GLUT: The project is implemented using OpenGL, a widely-used cross-platform graphics API, and the Graphics Library Utility Toolkit (GLUT), a user interface library, to create an efficient and portable application compatible with multiple platforms.

Output Primitives: The project uses various output primitives, such as triangles, circles, polygons, and rectangles, to create the various shapes and objects found within the lighthouse environment.

Animation: Our day and night switching feature gives a very impressive animation experience which can be controlled via keyboard keys.

Transformation: Transformation techniques, such as translation is used to position and orient objects like our light.

Logical Programming: The project uses logics to create the various shapes and objects found within the lighthouse environment.

User Input: The project incorporates user input to make it interactive, allowing users to control the movement of the boat and switching between daytime/night time.

Circle Drawing Algorithm: The project uses Bresenham Circle Drawing Algorithm for designing the circular components of the project such as the Sun/Moon, and the black circle on top of the lighthouse.

User-defined functions

Function	Description
Land ()	To design the land polygon using glBegin(GL_POLYGON); it includes if-else conditions based on which the color of the land changes according to day amd night.
Lighthouse ()	To design the lighthouse; it consists of various glBegin(GL_POLYGON) functions for drawing different polygonal parts of the lighthouse, while also having if-else conditions to change their color according to day and night time.
SunorMoon ()	To design the Sun/Moon using Bresenham's Circle Drawing Algorithm. It contains an if-else condition to draw the Sun or Moon according to the day and night time.
light ()	To design the guiding light of the lighthouse using glBegin(GL_POLYGON); it includes a SetPiixel() function that has the coordinates of the polygon which are variables that change with time and hence the dimensions of the yellow light Changes
star ()	To draw the stars for the night sky, The function is used various times to draw multiple stars and has their colour as a variable, which on changing, causes a twinkling effect.
Water ()	To draw the water polygon; its colour for the day and night is again changed with the help of if-else condition, while one corner of the polygon is moved in a sine wave to make it feel like flowing water.
sky ()	To design the sky polygon; if-else conditions are used to vary colours for day and night time.
boat ()	To draw the boat, Using glBegin(GL_POLYGON).
boatleft ()	To move the boat towards the left, which is called when the users presses certain button
boatright ()	To move the boat towards the right, which is called when the users presses certain button
keyboard ()	Includes switch cases to toggle between day and night modes, and to change the direction of the boat (left/right).

Code

```
#include <gl/GLUT.h>
#include <iostream>
#include <math.h>
#include<cmath>
using namespace std;

// Water parameters
float waterHeight = 290.0;            // Height of water surface
float waveSpeed = 0.1;                // Speed of waves
float waveAmplitude = 10.0;           // Amplitude of waves
float waveFrequency = 0.025;          // Frequency of waves
float M_PI = 3.14159265358979323846;  //pi
float xpos = 500;                     // Starting position of Light
float d = 50;                         // Thickness of Light
bool moving_right = true;             //For movement of Light
int delay = 0;                        //Delay for Light on ends
int n = 10;                           //Number of Stars
float col[10];                        //Colour of Stars
bool isDayMode = true;
float boatx = 0.0;

void SetPixel(int x, int y) {
    glVertex2f(x, y);
}

void Land() {
    if (isDayMode) {
        glColor3f(251.0/256,185.0/256,149.0/256);   //Light Brown
    }
    else {
        glColor3f(0.4101, 0.164, 0.164);     //Dark Brown

    }
    glBegin(GL_POLYGON);
    SetPixel(500, 0);
    SetPixel(1024, 0);
    SetPixel(1024, 300);
    SetPixel(500, 300);
    SetPixel(200, 0);
    glEnd();
}
void Lighthouse() {
    if (isDayMode) {
        glColor3f(1.0, 1.0, 1.0);   //White Base
    }
    else {
        glColor3f(0.7f, 0.7f, 0.7f);

    }
    glBegin(GL_POLYGON);
    SetPixel(740, 300);
    SetPixel(750, 350);
    SetPixel(850, 350);
    SetPixel(860, 300);
    glEnd();

    if (isDayMode) { ... }
    else { ... }
    glBegin(GL_POLYGON);
    SetPixel(750, 350);
    SetPixel(750, 500);
    SetPixel(850, 500);
    SetPixel(850, 350);
    glEnd();

    if (isDayMode) {
        glColor3f(1.0, 1.0, 1.0);   //White mid
    }
```

CGProject (Global Scope) Lighthouse()

```
74      else {
75          glColor3f(0.7f, 0.7f, 0.7f);
76
77      }
78      glBegin(GL_POLYGON);
79      SetPixel(750, 400);
80      SetPixel(750, 450);
81      SetPixel(850, 450);
82      SetPixel(850, 400);
83      glEnd();
84
85
86      if (isDayMode) {
87          glColor3f(1.0, 0.0, 0.0);   //Red Top
88      }
89      else {
90          glColor3f(0.5f, 0.0f, 0.0f);
91
92      }
93      glBegin(GL_TRIANGLES);
94      SetPixel(765, 540);
95      SetPixel(800, 560);
96      SetPixel(835, 540);
97      glEnd();
98
99      if (isDayMode) {
100         glColor3f(1.0, 1.0, 1.0);   //White Portion under Triangle Top
101     }
102     else {
103         glColor3f(0.7f, 0.7f, 0.7f);
104
105     }
106     glBegin(GL_POLYGON);
107     SetPixel(775, 500);
108     SetPixel(775, 540);
109     SetPixel(825, 540);
110     SetPixel(825, 500);
111     glEnd();
112
113     if (isDayMode) {
114         glColor3f(0.5, 0.5, 0.5);   //Grey Railing
115     }
116     else {
117         glColor3f(0.3, 0.3, 0.3);
118     }
119     glBegin(GL_POLYGON);
120     SetPixel(750, 500);
121     SetPixel(750, 510);
122     SetPixel(755, 510);
123     SetPixel(755, 500);
124     glEnd();
125     glBegin(GL_POLYGON);
126     SetPixel(845, 500);
127     SetPixel(845, 510);
128     SetPixel(850, 510);
129     SetPixel(850, 500);
130     glEnd();
131     glBegin(GL_POLYGON);
132     SetPixel(795, 500);
133     SetPixel(795, 510);
134     SetPixel(805, 510);
135     SetPixel(805, 500);
136     glEnd();
137     glBegin(GL_POLYGON);
138     SetPixel(750, 510);
139     SetPixel(750, 515);
140     SetPixel(850, 515);
141     SetPixel(850, 510);
142     glEnd();
```

```
    float r = 7;
    for (int i = r; i > 0; i--)
    {
        float x = 0, y = i, xc = 800, yc = 525;
        //float p = 1 - r;
        float p = 3 - 2 * i;
        glColor3f(0.0, 0.0, 0.0);
        glPointSize(2.0);
        glBegin(GL_POINTS);
        while (x <= y)
        {
            x++;
            if (p < 0)
            {
                //p = p + 2 * (x + 1) + 1;
                p = p + 4 * (x + 1) + 2;
            }
            else
            {
                y--;
                //p = p + 2 * (x + 1) + 1 - 2 * (y - 1);
                p = p + 4 * (x + 1) + 2 - 4 * (y - 1);
            }
            glVertex2i(xc + x, yc + y);
            glVertex2i(xc - x, yc + y);
            glVertex2i(xc + x, yc - y);
            glVertex2i(xc - x, yc - y);
            glVertex2i(xc + y, yc + x);
            glVertex2i(xc - y, yc + x);
            glVertex2i(xc + y, yc - x);
            glVertex2i(xc - y, yc - x);
        }
        glEnd();

    }
}
void SunorMoon() {
    int xc = 100, yc = 700;
    float r = 50;
    for (int i = r; i > 0; i--)
    {
        float x = 0, y = i;
        //float p = 1 - r;
        float p = 3 - 2 * i;
        if (isDayMode) {
            glColor3f(1.0, 1.0, 0.0);
        }
        else {
            glColor3f(0.9, 0.9, 0.9);

        }

        glPointSize(2.0);
        glBegin(GL_POINTS);
        while (x <= y)
        {
            x++;
            if (p < 0)
            {
                //p = p + 2 * (x + 1) + 1;
                p = p + 4 * (x + 1) + 2;
            }
            else
            {
                y--;
                //p = p + 2 * (x + 1) + 1 - 2 * (y - 1);
                p = p + 4 * (x + 1) + 2 - 4 * (y - 1);
            }
            glVertex2i(xc + x, yc + y);
```

```
            glVertex2i(xc - x, yc + y);
            glVertex2i(xc + x, yc - y);
            glVertex2i(xc - x, yc - y);
            glVertex2i(xc + y, yc + x);
            glVertex2i(xc - y, yc + x);
            glVertex2i(xc + y, yc - x);
            glVertex2i(xc - y, yc - x);
        }
        glEnd();

    }
}

void light() {

    glColor3f(1.0, 1.0, 0.0);//Yellow
    glBegin(GL_POLYGON);
    SetPixel(xpos, 200);
    SetPixel(xpos + d, 200);
    SetPixel(800, 525);
    glEnd();
}
void star(int x, int y, float col) {
    glColor3f(col, col, col);
    glBegin(GL_POINTS);
    SetPixel(x, y);
    SetPixel(x, y + 1);
    SetPixel(x, y + 2);
    SetPixel(x - 1, y + 1);
    SetPixel(x + 1, y + 1);
    SetPixel(x, y + 3);
    SetPixel(x, y - 1);
    SetPixel(x - 2, y + 1);
    SetPixel(x + 2, y + 1);
    glEnd();
}

void Water()
{
    if (isDayMode) {
        glColor3f(0.0f, 0.0f, 1.0f);
    }
    else {
        glColor3f(0.0f, 0.0f, 0.3f);

    }
    glBegin(GL_POLYGON);
    float x = 500 + sin(waveFrequency * glutGet(GLUT_ELAPSED_TIME) * waveSpeed) * waveAmplitude;
    float y = waterHeight + sin(waveFrequency * glutGet(GLUT_ELAPSED_TIME) * waveSpeed + M_PI / 2) * waveAmplitu
    float x2 = sin(waveFrequency * glutGet(GLUT_ELAPSED_TIME) * waveSpeed + M_PI / 2) * waveAmplitude;
    float y2 = waterHeight;
    SetPixel(-30, 0);
    SetPixel(500, 0);
    SetPixel(x, y);
    SetPixel( - 30 + x2, y2);
    glEnd();
}
void sky() {
    if (isDayMode) {
        glColor3f(0.529f, 0.808f, 0.922f);
    }
    else {
        glColor3f(0.0f, 0.0f, 0.2f);
    }
    glClear(GL_COLOR_BUFFER_BIT);
    glBegin(GL_POLYGON);
    SetPixel(0, 0);
    SetPixel(1024, 0);
    SetPixel(1024, 768);
```

```
    SetPixel(0, 768);
    glEnd();
}

void boat() {

    glColor3f(0.6f, 0.0f, 0.2f);
    glBegin(GL_POLYGON);
    SetPixel(0 + boatx, 330 + boatx*0.002*sin(waveFrequency * glutGet(GLUT_ELAPSED_TIME) * waveSpeed) * waveAmpl
    SetPixel(35 + boatx, 280 + boatx * 0.002 * sin(waveFrequency * glutGet(GLUT_ELAPSED_TIME) * waveSpeed) * wav
    SetPixel(115 + boatx, 280 + boatx * 0.002 * sin(waveFrequency * glutGet(GLUT_ELAPSED_TIME) * waveSpeed) * wa
    SetPixel(150 + boatx, 330 + boatx * 0.002 * sin(waveFrequency * glutGet(GLUT_ELAPSED_TIME) * waveSpeed) * wa
    glEnd();
    glColor3f(0.8f, 0.8f, 0.8f);
    glBegin(GL_POLYGON);
    SetPixel(35 + boatx, 330 + boatx * 0.002 * sin(waveFrequency * glutGet(GLUT_ELAPSED_TIME) * waveSpeed) * wav
    SetPixel(85 + boatx, 395 + boatx * 0.002 * sin(waveFrequency * glutGet(GLUT_ELAPSED_TIME) * waveSpeed) * wav
    SetPixel(85 + boatx, 330 + boatx * 0.002 * sin(waveFrequency * glutGet(GLUT_ELAPSED_TIME) * waveSpeed) * wav
    glEnd();
}
void boatleft() {
    if(boatx>0)
    boatx -= 0.001 * 1024;
}
void boatright() {
    if (boatx < 350)
    boatx += 0.001 * 1024;
}

void display()
{
    glClear(GL_COLOR_BUFFER_BIT);
    glMatrixMode(GL_MODELVIEW);
    sky();
    Lighthouse();
    if (isDayMode == false) {
        light();
        star(010, 750, col[0]);
        star(190, 675, col[1]);
        star(185, 590, col[2]);
        star(305, 700, col[3]);
        star(356, 685, col[4]);
        star(502, 650, col[5]);
        star(550, 725, col[6]);
        star(750, 700, col[7]);
        star(852, 750, col[8]);
        star(900, 640, col[9]);
    }
    boat();
    Water();
    Land();
    SunorMoon();
    glFlush();
}


void myInit()
{
    glClearColor(1.0, 1.0, 1.0, 0.0);

    glMatrixMode(GL_PROJECTION);
    gluOrtho2D(0.0, 1024.0, 0.0, 768.0);
}

void update(int value)
{
    for (int i = 0; i < n; i++) {
        col[i] = rand() / 32767.00;
    }
```

```
    }
    if (moving_right) {
        xpos += 10;
        d += -3;
        if (xpos >= 500) {
            delay += 1;
            xpos -= 10;
            d += 3;

            if (delay == 10)
                moving_right = false;
        }
    }
    else {
        xpos -= 10;
        d += 3;
        if (xpos <= -300) {
            delay -= 1;
            xpos += 10;
            d += -3;
            if (delay == 0)
                moving_right = true;
        }
    }
    glutPostRedisplay();    // Request a redraw of the window
    glutTimerFunc(100, update, 0);    // Call update function after 100 milliseconds
}
void keyboard(unsigned char key, int x, int y) {
    switch (key) {
    case 'm': // Switch to day mode
        isDayMode = true;
        break;
    case 'n': // Switch to night mode
        isDayMode = false;
        break;
    case 'a': // Switch to night mode
        boatleft();
        break;
    case 'd': // Switch to night mode
        boatright();
        break;
    }
}

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

    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(1024, 768);
    glutInitWindowPosition(0, 0);
    glutCreateWindow("CG Project");
    myInit();
    glutDisplayFunc(display);
    glutTimerFunc(0, update, 0);    // Call update function immediately
    glutKeyboardFunc(keyboard);
    glutMainLoop();


}
```

Output/Screenshots