

CSE 461 BİLGİSAYAR GRAFİKLERİ

ÖDEV 1

RAPOR

ESRA NUR ARICAN

161044028

ÖDEV TANIMI

Bir 3 boyutlu ev oluşturun ve etkileşimli olarak çevirme, döndürme ve ölçekleme dönüşümlerini uygulayın. (dönüştürmek için klavye tuşlarını kullanın)

ÖDEVİN UYGULANMASI

Ödevin gerçekleştirilmesi için öncelikle bilgisayarına OPENGGL ve Glut kütüphanelerini ekledim. Yazdığım kodun hem Windows hem de ubuntu ortamında çalışmasını test etmek için bu kurumları iki ortamda da uyguladım. Windows üzerinde VisualStudio 2019, ubuntu'da ise SublimeText3 geliştirme ortamlarını kullandım. Programı cpp dilinde yazdım.

Ödevde temel olarak ilgilendiğim aşamaları özetlemek gerekirse; 3 boyutlu evi oluşturmak, klavye ile eylemleri kontrol etme, translation, rotation ve scaling işlemlerini oluşturulan yapı üzerinde çalıştırmak olarak maddeleyebilirim.

Kullanılan Değişkenler

İlk olarak kullanacağım değişkenleri tanımladım. Bunlar sağa sola ve yukarı aşağı hareket için kullandığım değişkenler, scaling işlemi için kullandığım değişkenler ve rotation işlemi için kullandığım değişkenlerdi. Ayrıca programın çalışmasının durdurulması için de bir flag değişkeni kullandım.

```
using namespace std;

float shift = 0;           // variable used to move the house right and left
float shift2 = 0;          // variable to move the house up and down
float a = 0.5, b = 0.5, c = 0.5; // variable that used in scaling
float speed = 0.5;
int x = -1, y = -1, z = -1;
float rotationArray[3] = { 0,0,0 };
int axis = -1;
int flag = 1;              // used to disable rotation
float _angle = 45.0f;
```

Kullanılan değişkenlerin tanımlanması

Klavye Tuşları ile Girdi Alma

Klavye ile kullanıcıdan yapılacak işlemleri almak için iki adet fonksiyon yazdım. `handleKeyPress()` fonksiyonu klavyeden girilecek karakterler ile translation işleminin yönünü belirlemek ve scaling işlemi için evin boyutlarının büyüme küçülme seçeneğini belirlemek için kullanıldı. Kullanıcı klavye üzerinde `esc`'ye basarsa programdan çıkış yapar. 'p' harfine basarsa dönme işleminin durması sağlanır, tekrar basarsa işlem devam ettirilir. 'l', 'r', 'u', 'd' harfleri sırası ile sol, sağ, yukarı ve aşağı yöne evi hareket ettirir. 'b' ve 's' harflerine basılırsa sırası ile büyütme ya da küçültme işlemi uygulanır. Klavyeden her input alındığında değişkenler güncellendikten sonra `glutPostRedisplay()` metodu kullanılarak evin yeniden ekrana basılması işlemi yapılır.

```
void handleKeyPress(unsigned char key, int x, int y) {
    switch (key) {
        case 27: //exits program when user prints esc
            exit(0);

        case 'p': case 'P': // pause and continue
            flag *= -1;
            break;

        case 'l': case 'L': //move left
            shift -= 0.1;
            glutPostRedisplay();
            break;

        case 'r': case 'R': //move right
            shift += 0.1;
            glutPostRedisplay();
            break;

        case 'u': case 'U': //move up
            shift2 += 0.1;
            glutPostRedisplay();
            break;

        case 'd': case 'D': //move down
            shift2 -= 0.1;
            glutPostRedisplay();
            break;

        case 'b': case 'B': //bigger
            if (a < 1.25) {
                a += 0.01;
                b += 0.01;
                c += 0.01;
            }
            break;

        case 's': case 'S': //smaller
            if (a >= 0.4) {
                a -= 0.01;
                b -= 0.01;
                c -= 0.01;
            }
            break;
    }
}
```

handleKeyPress() fonksiyonu

handleKeyboardButtons() fonksiyonu ise klavyedeki sağ, sol, yukarı, aşağı ok tuşlarını kullanarak rotation işlemi için dönüş yapılacak yönün belirlenmesi için kullanıldı.

```
void handleKeyboardButtons(int button, int x, int y)
{
    switch (button)
    {
        case GLUT_KEY_LEFT: //rotate to left
        {
            axis = 0;
            x *= -1;
            glutPostRedisplay();
            break;
        }

        case GLUT_KEY_RIGHT: //rotate to right
        {
            axis = 1;
            y *= -1;
            glutPostRedisplay();
            break;
        }

        case GLUT_KEY_UP: //rotate to up
        {
            axis = 2;
            z *= -1;
            glutPostRedisplay();
            break;
        }
    }
    glutPostRedisplay();
}
```

handleKeyboardButtons() fonksiyonu

3 Boyutlu Evin Çizilmesi

3 boyutlu evin çizilmesi için temel olarak dikdörtgenlerle evin duvarlarını, üçgenler kullanarak çatı yapısını ayrıca yine dikdörtgen şekillerle kapı ve pencere çizmeye çalıştım. Bu işlemleri drawHouse() fonksiyonu içerisinde gerçekleştirdim. Ayrıca rotation, scaling fonksiyonları da bu fonksiyon içinde kullanıldı.

İlk olarak glClearColor(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT) fonksiyonunu kullanarak renklendirme ve derinlik için kullanılan buffer bitlerini temizleme işlemini yaptım.

glClearColor(0.196078, 0.6, 0.8, 0.0) metodu ile açılacak ekranın arka plan rengini değiştirdim. Bu renk default olarak siyah verilir, bunu mavi tonunda bir renk ile değiştirdim.

glMatrixMode(GL_MODELVIEW); glMatrixMode, geçerli matris modunu ayarlar. Mod, dört farklı değer alabilir. Ben mod olarak matris işlemlerini modelview matris yığınınına uygulayan GL_MODELVIEW kullandım.

Sonrasında rotateHouse() ve glScalef(a, b, c) fonksiyonlarını çağırdım. rotateHouse parametre almayan void bir fonksiyon, glScalef'e verilen a, b, c parametreleri ise klavyeden alınan inputa göre güncellenen değerlerdir.

Evin sağa sola, yukarı ve aşağı hareketi için daha önceden tanımlanmış olan shift ve shift2 isimli parametreleri kullandım. Bu parametreler klavye fonksiyonunda alınan girdiye göre artarak, evin çiziminde kullanılan koordinatlara eklendi. Böylece kullanıcı örneğin sağa gitmesi için 'r' tuşuna basarsa artan shift değeri, evin koordinatını değiştirir ve tekrar çizildiğinde sağa ilerlemiş şekilde çizilmiş olur.

3 boyutlu ev yapısını oluşturmak içinse ön, arka, sağ, sol ve evin tabanı için olmak üzere 5 adet dikdörtgen oluşturdum. Bunun için önce glColor3f(0.91, 0.76, 0.65) örneğinde olduğu gibi her şekilden önce rengini belirttim. Sonrasında glBegin(GL_POLYGON) glEnd() yapılarını kullanarak çizmek istediğim şekilleri oluşturdum. glBegin() Fonksiyonuna ev duvarları, kapı ve pencereler için GL_POLYGON parametresini verirken, çatı oluşturmak için GL_TRIANGLES parametresini verdim. Bu fonksiyon ile begin ve end ifadeleri arasında belirttiğimiz koordinatlar arasında oluşturmak istediğimiz yapıyı oluşturabiliyoruz. Çatıyı oluşturmak için aynı şekilde polygon ve üçgenler kullandım, ayrıca kapı ve pencereler için yine dikdörtgen yapılar oluşturdum.

```

/// <summary>
/// This method draws house to the screen with simply drawing rectangles and triangles
/// Rotating, scaling and translation function calls are also done in this method
/// </summary>
void drawHouse() {
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT); //clear color buffer bits
    glClearColor(0.196078, 0.6, 0.8, 0.0); //set background color
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    glTranslatef(-1.0f, -1.5f, -2.0f);
    glPushMatrix();
    glTranslatef(1.0f, 1.0f, 0.0f);
    rotateHouse(); //rotate house
    glScalef(a, b, c); // scale the house with parameters

    //front rectangle
    glColor3f(0.91, 0.76, 0.65);
    glBegin(GL_POLYGON);
    glVertex3f(0.2 + shift, 1.1+shift2, 0.0);
    glVertex3f(0.9 + shift, 1.1+shift2, 0.0);
    glVertex3f(0.9 + shift, 1.575+shift2, 0.0);
    glVertex3f(0.2 + shift, 1.575+shift2, 0.0);
    glEnd();

    //back rectangle
    glColor3f(0.91, 0.76, 0.65);
    glBegin(GL_POLYGON);
    glVertex3f(0.2 + shift, 1.1 + shift2, 0.5);
    glVertex3f(0.9 + shift, 1.1 + shift2, 0.5);
    glVertex3f(0.9 + shift, 1.575 + shift2, 0.5);
    glVertex3f(0.2 + shift, 1.575 + shift2, 0.5);
    glEnd();
}

```

```

//left triangle for roof
glColor3f( 0.647059, 0.164706, 0.164706);
glBegin(GL_TRIANGLES);
glVertex3f(0.9 + shift, 1.575 + shift2, 0.0);
glVertex3f(0.9 + shift, 1.575 + shift2, 0.5);
glVertex3f(0.9 + shift, 1.8 + shift2, 0.25);
glEnd();

//window
glColor3f(0.372549, 0.623529, 0.623529);
glBegin(GL_POLYGON);
glVertex3f(0.27 + shift, 1.25 + shift2, 0.51);
glVertex3f(0.38 + shift, 1.25 + shift2, 0.51);
glVertex3f(0.38 + shift, 1.4 + shift2, 0.51);
glVertex3f(0.27 + shift, 1.4 + shift2, 0.51);
glEnd();

//right triangle for roof
glColor3f(0.647059, 0.164706, 0.164706);
glBegin(GL_TRIANGLES);
glVertex3f(0.2 + shift, 1.575 + shift2, 0.0);
glVertex3f(0.2 + shift, 1.575 + shift2, 0.5);
glVertex3f(0.2 + shift, 1.8 + shift2, 0.25);
glEnd();

```

Evin oluşturulması ile ilgili yazılan fonksiyondan kesitler

Dönme İşleminin Uygulanması

Çizilen evin döndürülmesi için void rotateHouse() isimli bir fonksiyon yazdım. Burada bir switch case yapısı ile hangi yöne döndürüleceğine göre gereken değişkenleri güncelledikten sonra, şeklimizin dönüş işlemini gerçekleştirmesini sağlayan glRotatef() fonksiyonunu çağırdım. Bu fonksiyon evin çizildiği metod içerisinde çağırıldı.

```

/// <summary>
/// Method to rotate the house
/// </summary>
void rotateHouse() {
    if (flag == 1) {
        switch (axis)
        {
            case 0: rotationArray[0] += (speed * x);
                    break;
            case 1: rotationArray[1] += (speed * y);
                    break;
            case 2: rotationArray[2] += (speed * z);
                    break;
        }
    }
    glRotatef(rotationArray[0], 1.0, 0.0, 0.0);
    glRotatef(rotationArray[1], 0.0, 1.0, 0.0);
    glRotatef(rotationArray[2], 0.0, 0.0, 1.0);
}

```

rotateHouse() fonksiyonu

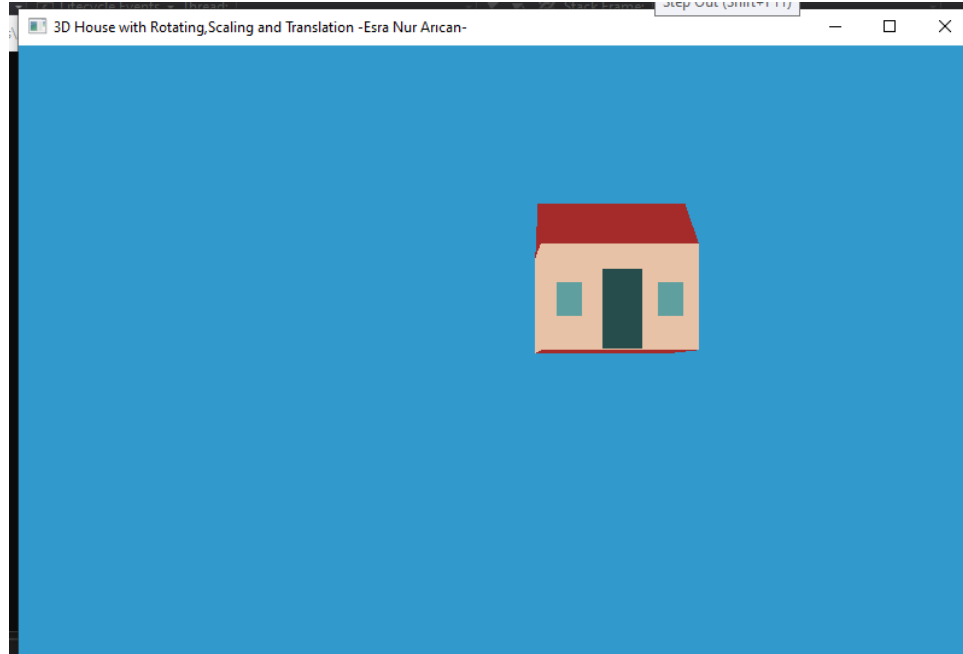
Main Fonksiyon

Son olarak main fonksiyonunda evin çizilmesi, klavye için yazılan fonksiyonlar gibi metodları çağırarak programımı sonlandırdım.

```
int main(int argc, char** argv) {  
    glutInit(&argc, argv);  
    glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);  
    glutInitWindowSize(1000, 800);  
    glutCreateWindow("3D House with Rotating,Scaling and Translation -Esra Nur Arıcan-");  
    initRendering();  
    glutDisplayFunc(drawHouse);  
    glutKeyboardFunc(handleKeypress);  
    glutSpecialFunc(handleKeyboardButtons);  
    glutReshapeFunc(handleResize);  
    glutTimerFunc(25, update, 0);  
    glutMainLoop();  
    return 0;  
}
```

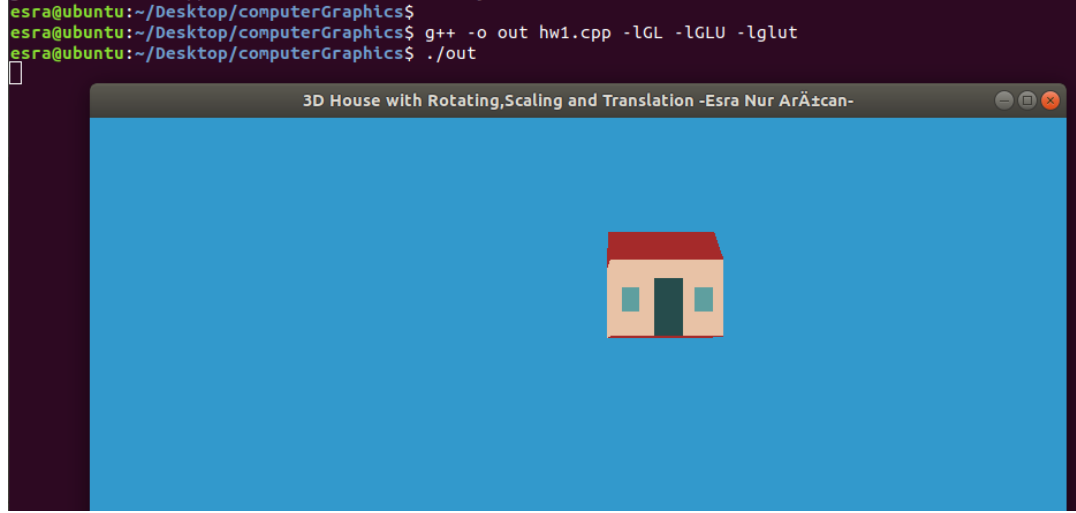
Main Fonksiyon

Programın Ekran Çıktıları



Windows ortamından örnek ekran görüntüsü

```
esra@ubuntu:~/Desktop/computerGraphics$  
esra@ubuntu:~/Desktop/computerGraphics$ g++ -o out hw1.cpp -lGL -lGLU -lglut  
esra@ubuntu:~/Desktop/computerGraphics$ ./out
```



Ubuntu ortamında kodun derlenmesi ve ekran görüntüsü

Video Sunum Linki

Programın çalışan halini çektiğim, istenilen özellikleri test ettiğim videoya erişim için:

<https://youtu.be/vTd6mhtYQcE>