T.C.

BURSA ULUDAĞ ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

BMB3013
BİLGİSAYAR GRAFİKLERİ
PROJE RAPORU

Abdullah AYDAN 031690056

Bünyamin Said İPEK 031690053

Volkan KÖSE 0316KVU90004

BİLGİSAYAR GRAFİKLERİ
PROJE RAPORU

Abdullah AYDAN[1] Bünyamin Said İPEK[2] Volkan KÖSE[3]

Bilgisayar Mühendisliği Bölümü, Bursa Uludağ Üniversitesi, Bursa

[1]e-posta: abdullah.aydan@outlook.com.tr [2] e-posta: bsaid2568@gmail.com
[3] e-posta: volkankose96@gmail.com

Özetçe

Uzayı seviyoruz ve ilgiliyiz bu yüzden ilgili olduğumuz bir konuyla çalışmak istedik. Kendi güneş sistemimizi yaptık. Dünyamızın yer aldığı güneş sisteminin simülasyonunu yapmaya çalışarak kendimiz uzayı keşfederken, çalışma prensiplerini sizlere sergileyerek sıra dışı bir proje yapmayı amaçladık. **Projede kişilere görev dağılımı yapılmadı, üç kişi beraber çalıştık.**

1. Uygulama Grafik İşlem Adımları ve Fonksiyonların Özeti

İlk olarak sabitlerimizi, ilerde kullanacağımız değişkenleri ve renklerimizi belirledik.

```
#define PI 3.14
float  angleEarth=0.0, angleMars=0.0, angleMercury=0.0;

GLfloat sx=0.2,sy=0.2,sz=0.2;
int planet1;
GLfloat black[]={0.0f,0.0f,0.0f,1.0f};
GLfloat yellow[]={0.7f,0.2f,0.0f,1.0f};
GLfloat qAmb[]={0.1,0.1,0.1,1.0};
GLfloat qDif[]={1.0,1.0,1.0,1.0};
GLfloat qSpec[]={.50,.50,.50,.10};
GLfloat qPos[]={0,0,0,0.1};
GLfloat sc[8]={0.295 , 0.40,0.50, 0.60,0.80,1.0,1.05,1.13};
double ang=2*PI/300;
double angular=2*PI/50;
```

Güneşimizi merkez alıp ışık kaynağı yapmak için initLighting() fonksiyonumuzu kullandık.

```
void initLighting()
{

    glEnable(GL_LIGHTING);
    glEnable(GL_LIGHT7);

    glLightfv(GL_LIGHT7,GL_AMBIENT,qAmb);
   glLightfv(GL_LIGHT7,GL_DIFFUSE,qDif);
    glLightfv(GL_LIGHT7,GL_SPECULAR,qSpec);

}
```

Yörüngelerimizi orbit() fonksiyonumuzla çizdirdik. Sade ve anlaşılır olması için Merkür, Dünya, Mars gezegenleri olmak üzere 3 tane yörüngeyi güneş sistemimize çizdirdik.

```
void orbit()
{
glColor3f(0.5,0.5,0.5);


int i=0;
for(i=0;i<4;i++){
glPushMatrix();
if(i==1)
{continue;}
glRotatef(63,1.0,0.0,0.0);
glScalef(sc[i],sc[i],sc[i]);
glBegin(GL_POINTS);
double angl=0.0;
  int i=0;
  for(i=0;i<300;i++)
  {glVertex2d(cos(angl),sin(angl));
   angl+=ang;   }
glEnd();
glPopMatrix();
}
```

Draw() fonksiyonuyla güneşi ve 3 gezegenin çizimi yapıldı. Alttaki gözüken kısımda sadece güneşin çizimi var. Aşağıdaki kod da iki tane glPushMatrix fonksiyonu varken bir tane glPopMatrix(); fonksiyonu var. Diğer gezegenlerin çizimi ve güneş bu glPushMatrix'in içinde yer aldığından sonlandığı kısmı görmüyoruz. Kod kısmının hepsini koymak zor olacağından buraya kısa bir kısmı koyduk.

```
void draw(void)
{
    glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
    background();
    orbit();
    glLoadIdentity();
    glPushMatrix();
    glEnable(GL_DEPTH_TEST);
   glEnable(GL_COLOR_MATERIAL);
      glPushMatrix();
      glColor3f(0.7,0.5,0.0);
      glScalef(sx,sy,sz);
       glLightfv(GL_LIGHT7,GL_POSITION,qPos);
      glMaterialfv(GL_FRONT_AND_BACK,GL_EMISSION,yellow);
      glutSolidSphere(1,50,50);
      glMaterialfv(GL_FRONT_AND_BACK,GL_EMISSION,black);
      glPopMatrix();
```

Yukarıda belirlerdiğimiz gezegen açılarımızı update() fonksiyonu ile güncelledik. Böylece hem ne kadar döneceğini hem de hangi gezegen ne kadar hızlı olacağını belirledik.

```
void update(int value){



angleEarth+=0.7;
if(angleEarth>360){
    angleEarth-=360;}

angleMercury+=2;
if(angleMercury>360){
    angleMercury-=360;}

angleMars+=0.5;
if(angleMars>360){
    angleMars-=360;}
```

Projemizde kullandığımız temel fonksiyonlarımız bunlardı.

1.1. Kütüphaneden Hazır Kullandığımız Başlıca Fonksiyonlar

- Işığı renklendirmek için:

glLightfv(GL_LIGHT7,GL_SPECULAR,qSpec);

- Işığı konumlandırmak için:

glLightfv(GL_LIGHT0,GL_POSITION,
isik_konum);

- Üç boyutlu çizim yapabilmek için:

glColor3f(0.0,0.00,0.00);

glVertex3f(-01.00,01.00,1);

- Görüntüyü x, y ve z eksenleri boyunca öteler:

glTranslatef(1.5,0.0,0.0);

- Görüntüyü x, y ve z eksenleri etrafında saat yönünde döndürür:

glRotatef(angleMercury,0.0,1.0,-0.5);

- Ekrandaki şekillerin verilen oranlarda büyümesi veya küçülmesini sağlar:

glScalef(0.08,0.08,0.08);

- Küre oluşturmak için:

glutSolidSphere(1,50,50);

- Belirlenen zamanda belirtilen fonksiyonu çalıştırmak için:

glutTimerFunc(20,update,0);

- Görüntüyü yenilemek için:

glutPostRedisplay();

Başlıca kütüphaneden hazır kullandığımız fonksiyonlar bunlardı.

2. Uygulama Kodu

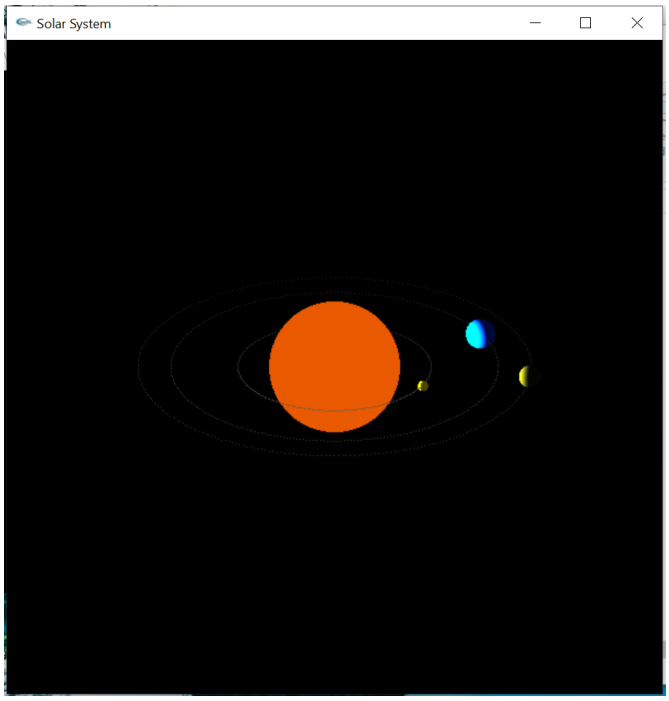
```
#include<stdio.h>
#include<stdlib.h>
#include <GL/glut.h>
#include<math.h>
#include<time.h>
#include<windows.h>
#define PI 3.14
float  angleEarth=0.0, angleMars=0.0, angleMercury=0.0;

GLfloat sx=0.2,sy=0.2,sz=0.2;
int planet1;
GLfloat black[]={0.0f,0.0f,0.0f,1.0f};
GLfloat yellow[]={0.7f,0.2f,0.0f,1.0f};
GLfloat qAmb[]={0.1,0.1,0.1,1.0};
GLfloat qDif[]={1.0,1.0,1.0,1.0};
GLfloat qSpec[]={.50,.50,.50,.10};
GLfloat qPos[]={0,0,0,0.1};
GLfloat sc[8]={0.295 , 0.40,0.50, 0.60,0.80,1.0,1.05,1.13};
double ang=2*PI/300;
double angular=2*PI/50;


void initLighting()
{

    glEnable(GL_LIGHTING);
    glEnable(GL_LIGHT7);

    glLightfv(GL_LIGHT7,GL_AMBIENT,qAmb);
   glLightfv(GL_LIGHT7,GL_DIFFUSE,qDif);
    glLightfv(GL_LIGHT7,GL_SPECULAR,qSpec);

}
void myinit()
{
  glClearColor(0.0,0.0,0.0,0.0);
   glPointSize(1.0);
   glLineWidth(2.0);

}

void background()
{
    glBegin(GL_QUADS);
      glColor3f(0.0,0.00,0.00);
      glVertex3f(-01.00,01.00,1);
```

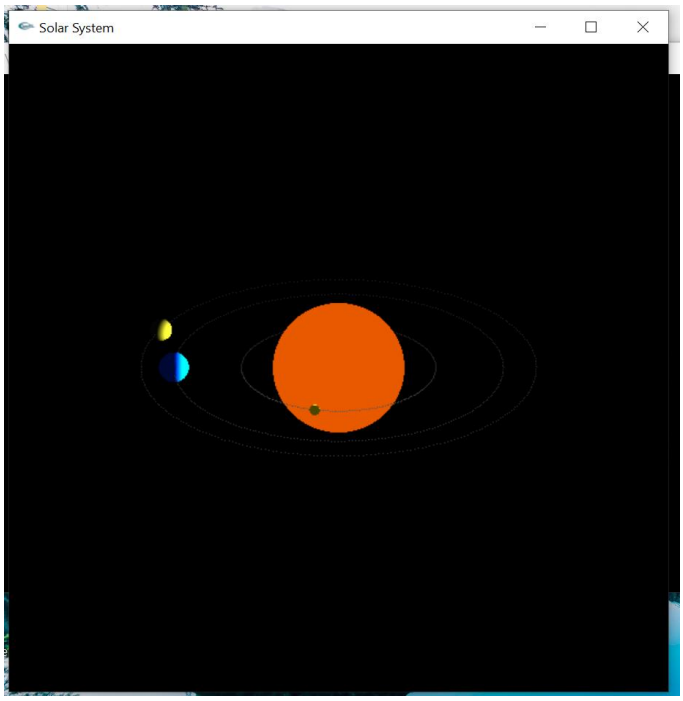
```
      glColor3f(.20,0.0,0.70);
      glVertex3f(01.00,1.00,1);
      glColor3f(0,0.0,0.0);
      glVertex3f(1.00,-1.00,1);
      glColor3f(.70,.10,.20);
      glVertex3f(-1.00,-1.00,1);
    glEnd();
}

void orbit()
{
glColor3f(0.5,0.5,0.5);


int i=0;
for(i=0;i<4;i++){
glPushMatrix();
if(i==1)
{continue;}
glRotatef(63,1.0,0.0,0.0);
glScalef(sc[i],sc[i],sc[i]);
glBegin(GL_POINTS);
double ang1=0.0;
  int i=0;
  for(i=0;i<300;i++)
  {glVertex2d(cos(ang1),sin(ang1));
   ang1+=ang;  }
glEnd();
glPopMatrix();
}
}

void draw(void)
{
    glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
    background();
    orbit();
    glLoadIdentity();
    glPushMatrix();
    glEnable(GL_DEPTH_TEST);
   glEnable(GL_COLOR_MATERIAL);
       glPushMatrix();
       glColor3f(0.7,0.5,0.0);
       glScalef(sx,sy,sz);
        glLightfv(GL_LIGHT7,GL_POSITION,qPos);
       glMaterialfv(GL_FRONT_AND_BACK,GL_EMISSION,yellow);
       glutSolidSphere(1,50,50);
       glMaterialfv(GL_FRONT_AND_BACK,GL_EMISSION,black);
       glPopMatrix();
```

```
      glScalef(0.2,0.2,0.2);
      glPushMatrix();
        glRotatef(angleMercury,0.0,1.0,-0.5);
        glTranslatef(1.5,0.0,0.0);
        glColor3f(1.0,0.9,0.0);
        glScalef(0.08,0.08,0.08);
        glutSolidSphere(1,50,50);
      glPopMatrix();

      glPushMatrix();
        glRotatef(angleEarth,0.0,1.0,-0.5);
        glTranslatef(2.5,0.0,0.0);
        glColor3f(0.0,0.1,0.7);
        glScalef(0.23,0.23,0.23);
        glutSolidSphere(1,50,50);

      glPopMatrix();//earth made

      glPushMatrix();
        glRotatef(angleMars,0.0,1.0,-0.5);
        glTranslatef(-3.0,0.0,0.0);
        glColor3f(0.05,0.05,0.01);
        glScalef(0.17,0.17,0.17);
        glutSolidSphere(1,50,50);
      glPopMatrix();

glPopMatrix();
    glFlush();
}


void update(int value){



angleEarth+=0.7;
if(angleEarth>360){
   angleEarth-=360;}

angleMercury+=2;
if(angleMercury>360){
   angleMercury-=360;}

angleMars+=0.5;
if(angleMars>360){
   angleMars-=360;}

glutPostRedisplay();
glutTimerFunc(20,update,0);
}
```

```
int main(int argc, char **argv)
{
   glutInit(&argc,argv);
   glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
   glutInitWindowPosition(0,0);
   glutInitWindowSize(600,600);
   glutCreateWindow("Solar System");
   initLighting();
   myinit();
   glutDisplayFunc(draw);
   glutTimerFunc(25,update,0);
   glutMainLoop();
   return 0;
}
```

3. Uygulamanın Ürettiği OpenGL Pencere Örnekleri





4. Kaynakça

[1] https://www.codemiles.com/c-opengl-examples/rotate-sphere-in-a-circle-with-light-t9130.html?mobile=on
[2] https://cs.lmu.edu/~ray/notes/openglexamples/
[3] https://stackoverflow.com/questions/21088157/creating-a-solar-system-in-opengl-lighting-not-working-as-planned
[4] https://stackoverflow.com/questions/5744929/opengl-all-planets-have-the-same-speed-around-the-sun-when-animating
[5] https://community.khronos.org/t/i-dont-understand-what-glutpostredisplay-does/58715
[6] https://www.opengl.org/resources/libraries/glut/spec3/node20.html