UNIVERZITET U TUZLI
FAKULTET ELEKTROTEHNIKE

IZVJEŠTAJ O PROJEKTU
RAČUNARSKA GRAFIKA

Student: Tamara Simić
Broj indeksa: 16135
Usmjerenje: Računarstvo i informatika

Profesor: dr. sc. Emir Skejić,
vanr. prof.

Tuzla, juni 2019

Sadržaj

Zadatak

Potrebno je napisati OpenGL program koji će prikazati Mjesec, Zemlju i Sunce uz zadane funkcionalnosti.

Rješenje

Prikaz Sunca, Zemlje i Mjeseca

Prikaz Sunca

```
//funckija za iscrtavanje sunca
void drawSun(){
  glPushMatrix();
  glEnable(GL_LIGHT1);
  if(texture){ //zelimo imati teksturu na animaciji ili ne
    glEnable(GL_TEXTURE_2D);
    glBindTexture(GL_TEXTURE_2D,sunTextureId);
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER,
GL_NEAREST);
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER,
GL_NEAREST);
    GLUquadricObj *quad = gluNewQuadric();
    gluQuadricTexture(quad,GLU_TRUE);
    gluSphere(quad,SUN_RADIUS,30,30);
    glDisable(GL_TEXTURE_2D);
  }
  else{ //bez teksture
    glClear(GL_COLOR_BUFFER_BIT);
    glColor3f(0.650,0.650,0.650);
    GLUquadricObj *quad = gluNewQuadric();
    gluSphere(quad,SUN_RADIUS,30,30);
  }

  glDisable(GL_LIGHT1);
  glPopMatrix();

}
```

Prikaz Zemlje

```
double EarthRevolutionW = 2.0*PI/3560; // brzina okretanja Zemlje
oko Sunca
double EarthRevolutionTheta = 0; //ugao revolucije

double EarthRotationW = 30; //brzina okretanja Zemlje oko svoje
ose
double EarthRotationTheta = 0; //ugao rotacije

void drawEarth(){ //funckija za iscrtavanje Zemlje
```

```
  double x = cos(EarthRevolutionTheta);
  double y = sin(EarthRevolutionTheta);
  glPushMatrix();
  glTranslatef(40*x,40*y,0);
  glRotatef(EarthRotationTheta,0,0,1);
  if(texture){ //zelimo imati teksturu
    glEnable(GL_TEXTURE_2D);
    glBindTexture(GL_TEXTURE_2D,earthTextureId);
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER,
GL_NEAREST);
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER,
GL_NEAREST);
    GLUquadricObj *quad = gluNewQuadric();
    gluQuadricTexture(quad,GLU_TRUE);
    gluSphere(quad,EARTH_RADIUS,30,30);
    glDisable(GL_TEXTURE_2D);
  }
  else{ //bez teksture
    GLUquadricObj *quad = gluNewQuadric();
    glColor3f(0,0,1);
    gluSphere(quad,EARTH_RADIUS,30,30);
  }

  glPopMatrix();
  glPushMatrix();
  glTranslatef(40*x,40*y,0);
  MoonUpdate(); //poziv funckije za mijenjanje polozaja Mjeseca
  glPopMatrix();
}

void EarthUpdate(){ // funkcija za mijenjanje polozaja Zemlje
  drawEarth();
  EarthRevolutionTheta += EarthRevolutionW;
  EarthRotationTheta += EarthRotationW;
}
```

Prikaz Mjeseca

```
double MoonRevolutionW = 2.0*PI/273; //brzina okretanja Mjeseca
oko Zemlje
double MoonRevolutionTheta = 0; //ugao revolucije

void drawMoon(){ //funckija za iscrtavanje
  glPushMatrix();
  double x = cos(MoonRevolutionTheta);
  double y = sin(MoonRevolutionTheta);
  glTranslatef(10*x,10*y,0);
  if(texture){ //sa teksturom
    glEnable(GL_TEXTURE_2D);
    glBindTexture(GL_TEXTURE_2D,moonTextureId);
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER,
GL_NEAREST);
```

```
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER,
GL_NEAREST);
    GLUquadricObj *quad = gluNewQuadric();
    gluQuadricTexture(quad,GLU_TRUE);
    gluSphere(quad,MOON_RADIUS,30,30);
  }
  else{ //bez teksture
    GLUquadricObj *quad = gluNewQuadric();
    glColor3f(0.192,0.192,0.192);
    gluSphere(quad,MOON_RADIUS,30,30);
  }

  glPopMatrix();
}

void MoonUpdate(){ // funkcija za mijenjanje polozaja Mjeseca
  drawMoon();
  MoonRevolutionTheta += MoonRevolutionW;
}
```

U prethodnim funckijama, vidimo da prilikom iscrtavanja, vodimo računa o tome da li je korisnik sa tastature unio opciju za prikaz teksture. Ukoliko jeste, crtamo sferu sa teksturom, ali ako nije, crtamo Sunce, Zemlju i Mjesec kao bijelosivu, plavu i sivu teksturu.

Funkcija za inicijalizaciju

```
void init(){ // funkcija za inicijalizaciju
  glClearColor(0.0f,0.0f,0.0f,1.0f);
  glEnable(GL_DEPTH_TEST);
  glShadeModel(GL_SMOOTH);

  //postavljanje tekstura
  Image* image = loadBMP("earth.bmp");
  earthTextureId = loadTexture(image);
  delete image;

  image = loadBMP("sun.bmp");
  sunTextureId = loadTexture(image);
  delete image;

  image = loadBMP("moon.bmp");
  moonTextureId = loadTexture(image);
  delete image;

  glEnable(GL_NORMALIZE);
}
```

Funckija za postavljanje osvjetljenja

```
void makeEnvironmentWithLight(){
```

```
  GLfloat ambient[4] = {0.2,0.2,0.2,1};
  GLfloat diffuse[4] = {1,1,1,1}; //sunce kao difuzni izvor
svjetlosti
  GLfloat position[4] = {0,0,0,1};
  GLfloat specular[4] = { 1, 1, 1, 1}; //sunce sa bijelom
spekularnom komponentom
  glLightfv(GL_LIGHT0,GL_AMBIENT,ambient);
  glLightfv(GL_LIGHT0,GL_DIFFUSE,diffuse);
  glLightfv(GL_LIGHT0,GL_POSITION,position);
  glLightfv(GL_LIGHT0, GL_SPECULAR, specular);

  GLfloat ambient1[4] = {5,5,5,1};
  GLfloat diffuse1[4] = {1,1,1,1};
  GLfloat position1[4] = {0,0,0,1};
  glLightfv(GL_LIGHT1,GL_AMBIENT,ambient1);
  glLightfv(GL_LIGHT1,GL_DIFFUSE,diffuse1);
  glLightfv(GL_LIGHT1,GL_POSITION,position1);
}
```

Funckija za postavljanje kamere

```
void makeCamera(){
  int w = glutGet((GLenum)GLUT_WINDOW_WIDTH); //trenutna sirina
prozora
  int h = glutGet((GLenum)GLUT_WINDOW_HEIGHT); // trenutna visina
prozora
  glViewport(0,0,w,h);

  glMatrixMode(GL_PROJECTION);
  glLoadIdentity();
  gluPerspective(60,1.0*w/h,1.0,MAX_LENGTH); //odredjivanje
perspektive

  glMatrixMode(GL_MODELVIEW);
  glLoadIdentity();
  gluLookAt(50,50,20,0,0,0,0,0,2); //postavljanje polozaja kamere

}
```

Callback funckije

Display funkcija

```
void display(){ //display funckija
  glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

  makeEnvironmentWithLight(); //poziv funckije za postavljanje
osvjetljenja

  makeCamera(); //poziv funkcije za postavljanje kamere
```

```
  drawSun(); //poziv funkcije za iscrtavanje sunca
  EarthUpdate(); //poziv funckije koja iscrtava zemlju i vrsi
njenu rotaciju i revoluciju

  glutSwapBuffers();
  glFlush();
}
```

Timer funkcija

```
void timer(int e){ // funckija za pokretanje tajmera koji oznacava
pocetak animacije
  glutPostRedisplay();
  glutTimerFunc(30,timer,0);
}
```

Mouse funckija

```
void mouse(int button, int state, int x, int y){ //funkcija koja
mijenja brzinu rotacije i revolucije u zavisnosti od klika na mis
  if(button==GLUT_RIGHT_BUTTON){ //pritisak na lijevo dugme
oznacava povecanje brzine tj. smanjenje trajanja dana za jednu
polovinu
    EarthRotationW*=2;
    EarthRevolutionW*=2;
    MoonRevolutionW*=2;
  }
  if(button==GLUT_LEFT_BUTTON){ //pritisak na desno dugme oznacava
smanjenje brzine tj. povecanje trajanja dana za jednu polovinu
    EarthRotationW/=2;
    EarthRevolutionW/=2;
    MoonRevolutionW/=2;
  }
}
```

Funckija za tastaturu

```
void kbd(unsigned char key, int x, int y){
  switch(key){
    case 27: // ukoliko se pritisne tipka Esc, izlazi se iz
prozora
      exit(0);
      break;
    case 'n': // 'n' oznacava prikaz animacije bez osvjetljenja i
teksture
    case 'N':
      glDisable(GL_LIGHTING);
      glDisable(GL_LIGHT0);
```

```
      texture=false;
      break;
    case 'l': // 'l' oznacava prikaz animacije sa osvjetljenjem
    case 'L':
      glEnable(GL_LIGHTING);
      glEnable(GL_LIGHT0);
      break;
    case 't': // 't' oznacava prikaz animacije sa teksturom
    case 'T':
      texture=true;
      break;
  }
}
```

Main funckija

```
int main(int argc, char **argv){
  glutInit(&argc,argv);
  glutInitWindowSize(800,600);
  glutInitWindowPosition(100, 100);
  glutCreateWindow("Sunce, Zemlja i Mjesec");
  glutInitDisplayMode(GL_RGB | GL_DEPTH | GL_DOUBLE);

  glutDisplayFunc(display);
  glutTimerFunc(30,timer,0);
  glutMouseFunc(mouse);
  glutKeyboardFunc(kbd);

  init();
  glutMainLoop();
  return 0;
}
```

Tekstura

Za dodavanje teksture, korištena je klasa `Image` zajedno sa dodatnim metodima i funkcijama koji vrše dodavanje teksture na Mjesec, Zemlju i Sunce, koji se nalaze u fajlu `image.h`. Funckija `loadBMP` učitava sliku iz memorije u `.bmp` formatu, obrađuje je i vraća objekat tipa `Image` kao vrijednost koju će funckija `loadTexture` koristiti za postavljanje teksture. Također, imamo i globalnu varijablu `texture`, tipa `bool`, koja nam govori da li korisnik želi prikazati teksturu na animaciji ili ne. Korisnik vrši taj odabir prilikom pritiska tipke 't' ili 'T', a na taj način se poziva callback funckija `kbd`, koja će omogućiti prikaz teksture.

Sadržaj "image.h" fajla

```
#ifndef IMAGE_H
#define IMAGE_H
#include <fstream>
#include <assert.h>
```

```
struct Image {

     char* pixel;
     int width;
     int height;

     Image(char* data, int w, int h)
     {
         pixel = data;
         width  = w;
         height = h;
     }

     ~Image()
     {
         delete [] pixel;
     }

};

GLuint Texture;
Image* planet;
GLUquadricObj *ball;

template<class T>
struct auto_array
{
    T* array;
    bool isReleased;

    auto_array(T* array_ = NULL) :
        array(array_), isReleased(false) {}

    ~auto_array()
    {
        if (!isReleased && array != NULL)
        {
        delete[] array;
        }
    }

    T* release()
    {
        isReleased = true;
        return array;
    }
};

int readInt(std::ifstream &ulaz)
{
    char buffer[4];
    ulaz.read(buffer, 4);
    return (int)(((unsigned char)buffer[3] << 24) |
```

```
        ((unsigned char)buffer[2] << 16) |
        ((unsigned char)buffer[1] << 8) |
        (unsigned char)buffer[0]);
}

Image* loadBMP(const char* file)
{
     std::ifstream ulaz;
     ulaz.open(file, std::ifstream::binary);
    char buffer[2];
     ulaz.read(buffer, 2);
     ulaz.ignore(8);

     int dataOffset = readInt(ulaz);
     int headerSize = readInt(ulaz);
     int width= readInt(ulaz);
    int height = readInt(ulaz);
    ulaz.ignore(2);

     int bytesPerRow = ((width * 3 + 3) / 4) * 4 - (width * 3 %
4);
     int size = bytesPerRow * height;

     auto_array<char> pixels(new char[size]);
     ulaz.seekg(dataOffset, std::ios_base::beg);
     ulaz.read(pixels.array, size);

     auto_array<char> pixels2(new char[width * height * 3]);
     for (int y = 0; y < height; ++y)
    {
          for (int x = 0; x < width; ++x)
          {
               for (int c = 0; c < 3; ++c)
               {
                    pixels2.array[3 * (width * y + x) + c] =
                pixels.array[bytesPerRow * y + 3 * x + (2 - c)];
               }
          }
     }

     ulaz.close();
     return new Image(pixels2.release(), width, height);
}

#endif
```

Funkcija za učitavanje teksture

```
GLuint loadTexture(Image* image){
  GLuint textureId;
  glGenTextures(1, &textureId);
  glBindTexture(GL_TEXTURE_2D, textureId);
```

```
  glTexImage2D(GL_TEXTURE_2D, 0, GL_RGB, image->width,
image->height, 0, GL_RGB, GL_UNSIGNED_BYTE, image->pixel);
  return textureId;
}
```

Zaključak

U rješenju su implementirane sljedeće funkcionalnosti:

- prikaz Mjesec koji kruži oko Zemlje koja kruži oko Sunca
- rotacija i revolucija Zemlje oko Sunca kao i revolucija Mjeseca oko Zemlje
- mogućnost korisnika da udvostruči dužinu dana klikom na lijevo dugme miša i prepolovi dužinu dana klikom na desno dugme miša
- omogućavanje/onemogućavanje osvjetljenja pomoću tastature
- dodavanje teksture na Zemlju, Mjesec i Sunce