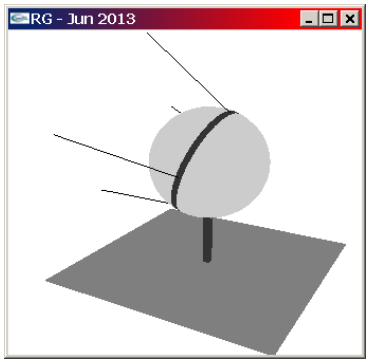
ELEKTROTEHNIČKI FAKULTET 10.06.2013.
Katedra za RTI

# ISPIT IZ RAČUNARSKE GRAFIKE

**1)** [30] **OpenGL:** Napisati na jeziku C ili C++ deo programa za crtanje scene prikazane na slici 1 primenom grafičke biblioteke OpenGL. Scena se sastoji od figure koja predstavlja pojednostavljen model satelita Sputnjik postavljen na postolje kvadratnog oblika. Telo satelita je sfernog oblika. Četiri kraka dodiruju sferu na polutaru, pod uglom od 20° u odnosu na tangentu odgovarajućeg meridijana. Površ sfere obojena je svetlo sivo, a pojas na polutaru tamno sivo. Kraci se crtaju linijama crne boje. Postolje je kvadratnog oblika, sive boje. Postolje i satelit povezuje držač oblika kvadra tamno sive boje. Scena se crta na beloj podlozi. Koristi se projekcija sa perspektivom. Napisati posebnu funkciju koja vrši osnovnu inicijalizaciju OpenGL sistema potrebnu za crtanje scene. Parametre projekcije podesiti i posmatračku kameru približno postaviti u položaj koji bi proizveo sliku 1. Smatrati da je otvaranje prozora za crtanje realizovano u glavnom programu koji nije potrebno pisati.



Slika 1

**2)** [20] Korišćenjem Bresenham-ovog algoritma za crtanje kružnice kao osnove, potprograma za crtanje linije `Line(xa, ya, xb, yb)` koji crta liniju od tačke ($x_a$, $y_a$) do tačke ($x_b$, $y_b$) i potprograma za crtanje tačke `Plot(x, y)` koji crta tačku (x, y), napisati potprogram čija je deklaracija

*void Crtaj(int x1, int y1, int x2, int y2, bool mod);*

gde *mod* određuje da li će se crtati figura 2a ili figura 2b, na slici 2. *mod* je *true* za figuru 2b. Tačka (x1, y1) predstavlja donji levi, a (x2, y2) gornji desni ugao figure. Pretpostaviti da je pre poziva potprograma izvršena provera validnosti datih koordinata. Crtanje je crnom olovkom na beloj pozadini.


figura 2a figura 2b

Slika 2

**3)** [20] Iterativni algoritam popunjavanja oblasti.

**4)** [30] Odgovoriti koncizno (jedna do dve rečenice) na sledeća pitanja:

**a)** Koja primitivna operacija sa kojim vrednostima argumenata (opisno, nije potrebno navoditi preciznu sintaksu) se koristi da se pomoću SRGP-a nacrta krug?

**b)** Napisati o objasniti uslove trivijalnog prihvatanja i odbacivanja linije u *Cohen-Sutherland* algoritmu.

**c)** Opisati komponente boje u HSV sistemu i njihove opsege.

**Napomene:** 1. Ispit traje 180 minuta.

2. Rad se predaje isključivo u vežbanci za ispite. Nije dozvoljeno imati pored sebe druge listove papira, niti uz sebe imati mobilni telefon, bez obzira da li je uključen ili isključen.
3. Nije dozvoljena upotreba literature niti programabilnih kalkulatora.
4. Dozvoljena je upotreba OpenGL podsetnika.
5. Voditi računa o urednosti. Nečitki delovi teksta će biti tretirani kao nepostojeći. Rešenja zadataka navesti po gornjem redosledu (-1 poen za loš redosled). Preporučuje se rad običnom grafitnom olovkom.

**Rešenja zadataka**

***Jun 2013***

1) Rešenje (delovi programa obeleženi sivo nisu deo očekivanog rešenja)

```
/* RG 2012-2013: Jun */
#include <stdlib.h>
#include <string.h>
#include <math.h>
#include "GL/glut.h"
#include "GL/GL.H"
#define D2R(x)  ((x)*3.14159265358979/180)
const float tamno_siva[] =
                      {0.2f, 0.2f, 0.2f};
const float svetlo_siva[] =
                      {0.8f, 0.8f, 0.8f};
const float siva[] = {0.5f, 0.5f, 0.5f};
const float crna[] = {0, 0, 0};

void promenaProzora(int width, int height)
{
  glViewport(0, 0, (GLint) width, (GLint)
height);
}

void crtajSferu(int brSegPhi, int
brSegTheta, const float *boja1,
                        const float
*boja2) {
  float korakPhi = 360.f / brSegPhi;
  float korakTheta = 90.f / brSegTheta;
  static const float r = 0.5f;
  glPushMatrix();
  glPushAttrib(GL_CURRENT_BIT);
  for(int strana = 0;strana < 2;strana++){
    float theta = 0;
    for(int j=0; j<brSegTheta-1; j++){
      float phi = 0;
      if( j == 0 ) glColor3fv(boja1);
      else         glColor3fv(boja2);
      glBegin(GL_QUAD_STRIP);
      for(int i = 0; i <= brSegPhi; i++) {
      float x =
          r*cos(D2R(phi))*cos(D2R(theta));
      float z =
          r*sin(D2R(phi))*cos(D2R(theta));
      float y = r*sin(D2R(theta));
      glVertex3f(x, y, z);
      x =r*cos(D2R(phi))*
         cos(D2R(theta+korakTheta));
      z = r*sin(D2R(phi))*
         cos(D2R(theta+korakTheta));
      y = r*sin(D2R(theta+korakTheta));
      glVertex3f(x, y, z);
      phi += korakPhi;
      }
      glEnd();
      theta += korakTheta;
    }
    glBegin(GL_TRIANGLE_FAN);
    glVertex3f(0, r, 0);
    float phi = 0;
    for(int i = 0; i <= brSegPhi; i++) {
      float x = r*cos(D2R(phi))*
                cos(D2R(theta));
      float z = r*sin(D2R(phi))*
                cos(D2R(theta));
      float y = r*sin(D2R(theta));
      glVertex3f(x, y, z);
      phi += korakPhi;
    }
    glEnd();
    glScalef(1, -1, 1);
  }
  glPopAttrib();
  glPopMatrix();
}

void crtajKrake(int brojKraka, float
duzina, const float *boja) {
  glPushMatrix();
  glPushAttrib(GL_CURRENT_BIT);
  glColor3fv(boja);
  float korak = 360.f/brojKraka;
  for(int i = 0; i < brojKraka; i++) {
    glRotatef(korak, 0, 1, 0);
    glPushMatrix();
    glTranslatef(0.5f, 0, 0);
    glRotatef(20, 0, 0, 1);
    glBegin(GL_LINES);
      glVertex3f(0, 0, 0);
      glVertex3f(0, -duzina, 0);
    glEnd();
    glPopMatrix();
  }
  glPopAttrib();
  glPopMatrix();
}

void crtajSatelit() {
  glPushMatrix();
  glTranslatef(0, 0.5, 0);
  glRotatef(110, 1, 0, 0);
  crtajSferu(30, 30, tamno_siva,
svetlo_siva);
  crtajKrake(4, 1, crna);
  glPopMatrix();
}

void crtajPostolje() {
  glPushMatrix();
  glPushAttrib(GL_CURRENT_BIT);
  glColor3fv(siva);
  glTranslatef(0, -0.5, 0);
  glBegin(GL_QUADS);
    glVertex3f(-1, 0, -1);
    glVertex3f( 1, 0, -1);
    glVertex3f( 1, 0,  1);
    glVertex3f(-1, 0,  1);
  glEnd();
  const float d = 0.03f;
  glColor3fv(tamno_siva);
  glBegin(GL_QUAD_STRIP);
    glVertex3f(-d, 0, -d);
    glVertex3f(-d, 0.5, -d);
    glVertex3f( d, 0, -d);
    glVertex3f( d, 0.5, -d);
    glVertex3f( d, 0, d);
    glVertex3f( d, 0.5, d);
    glVertex3f( -d, 0, d);
    glVertex3f( -d, 0.5, d);
    glVertex3f(-d, 0, -d);
    glVertex3f(-d, 0.5, -d);
  glEnd();
  glPopAttrib();
  glPopMatrix();
}

void crtajScenu(void) {
  glClear(GL_COLOR_BUFFER_BIT |
          GL_DEPTH_BUFFER_BIT);
  glLoadIdentity();
  glTranslatef( 1, -1, -21 );
  glRotatef(20, 1, 0, 0);
  glRotatef(60, 0, 1, 0);
  glScalef(5, 5, 5);
  crtajSatelit();
  crtajPostolje();
  glFlush();
}

void init() {
  glPolygonMode(GL_FRONT_AND_BACK,
                GL_FILL);
  glDisable(GL_CULL_FACE);
  glClearColor (1.0, 1.0, 1.0, 1.0);
  glEnable(GL_DEPTH_TEST);
  glMatrixMode(GL_PROJECTION);
  glLoadIdentity();
  gluPerspective(40, 1, 1, 600);
  glMatrixMode(GL_MODELVIEW);
  glLoadIdentity();
}

int main(int argc, char **argv) {
  glutInit(&argc, argv);
  glutInitDisplayMode (GLUT_RGB |
GLUT_DEPTH);
  glutInitWindowSize(500, 500);
  glutInitWindowPosition(100, 100);
  glutCreateWindow("RG - Jun 2013");
  init();
  glutDisplayFunc(crtajScenu);
  glutSpecialFunc(specFunc);
  glutReshapeFunc(promenaProzora);
  glutMainLoop();
  return 0;
}
```

2)

```
void crtaj(int x1, int y1, int x2, int y2, bool mod) {
int d, x, y, cx, cy, r;

  line(x1, y1, x2, y1);
  line(x2, y1, x2, y2);
  line(x2, y2, x1, y2);
  line(x1, y2, x1, y1);

  cy = (y2+y1)/2;
  cx = (x2+x1)/2;
  r = (x2-x1)/2;

  x = r;
  y = 0;
  d = 3-2*x;
  while(x >= y)
  {
    if( mod )
    {
      line(cx-x, y2-y, cx+x, y2-y);
      line(x1, cy+r-x, cx-y, cy+r-x);
      line(cx+y, cy+r-x, x2, cy+r-x);

      line(cx-x, y1+y, cx+x, y1+y);
      line(x1, cy-r+x, cx-y, cy-r+x);
      line(cx+y, cy-r+x, x2, cy-r+x);
    }
    else
    {
      line(cx-y, cy+r-x, cx+y, cy+r-x);
      line(x1, y2-y, cx-x, y2-y);
      line(cx+x, y2-y, x2, y2-y);

      line(cx-y, cy-r+x, cx+y, cy-r+x);
      line(x1, y1+y, cx-x, y1+y);
      line(cx+x, y1+y, x2, y1+y);

    }

    if( d < 0 )
      d += 4*y+6;
    else
    {
      d += 4 * (y-x)+10;
      x--;
    }
    y++;
    }
}
```