II kolokvijum 2020

Definisati perspektivnu projekciju sa FOV = 50° i ispuniti funkcije PrepareScene(), DrawScene() i Reshape() odgovarajućim OpenGL funkcijskim pozivima kako bi se omogućilo dalje crtanje. [5 poena]

```
void CGLRenderer::PrepareScene(CDC *pDC)
{
   wglMakeCurrent(pDC->m_hDC, m_hrc);

   glClearColor(1.0, 1.0, 1.0, 1.0);
   glEnable(GL_DEPTH_TEST);

   wglMakeCurrent(NULL, NULL);
}
```

```
void CGLRenderer::Reshape(CDC *pDC, int w, int h)
{
   wglMakeCurrent(pDC->m_hDC, m_hrc);
   glViewport(0, 0, (GLsizei)w, (GLsizei)h);
   glMatrixMode(GL_PROJECTION);
   glLoadIdentity();
   gluPerspective(50, (double)w / (double)h, 0.1, 2000);
   glMatrixMode(GL_MODELVIEW);
   wglMakeCurrent(NULL, NULL);
}
```

```
void CGLRenderer::DrawScene(CDC *pDC)
{
   wglMakeCurrent(pDC->m_hDC, m_hrc);
   glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
   glLoadIdentity();
   // …
   glFlush();
   SwapBuffers(pDC->m_hDC);
   wglMakeCurrent(NULL, NULL);
}
```

Napisati funkciju void CGLRenderer::DrawAxes(), koja crta koordinatne ose dužine 50 jedinica, obojene različitim bojama. Neka je linija duž X-ose plavo, linija duž Y-ose crveno, a duž Z-ose zelena. Preći na sledeću tačku tek kada koordinatne ose budu vidljive. [5 poena]

```
void CGLRenderer::DrawAxes()
{
        glLineWidth(2.0);

        glDisable(GL_TEXTURE_2D);
        glBegin(GL_LINES);

        glColor3f(0, 0, 1);
        glVertex3f(0, 0, 0);
        glVertex3f(50, 0, 0);

        glColor3f(1, 0, 0);
        glVertex3f(0, 0, 0);
        glVertex3f(0, 50, 0);

        glColor3f(0, 1, 0);
        glVertex3f(0, 0, 0);
        glVertex3f(0, 0, 50);

        glEnd();
}
```

Napisati funkciju UINT CGLRenderer::LoadTexture(char* fileName), koja učitava teksturu sa datim imenom (fileName) i vraća ID kreirane teksture. Korišćenjem ove funkcije u okviru PrepareScene() učitati teksture: spider.png, front.jpg, left.jpg, right.jpg, back.jpg, top.jpg i bot.jpg. Teksture obrisati u DistoryScene(). [10 poena]

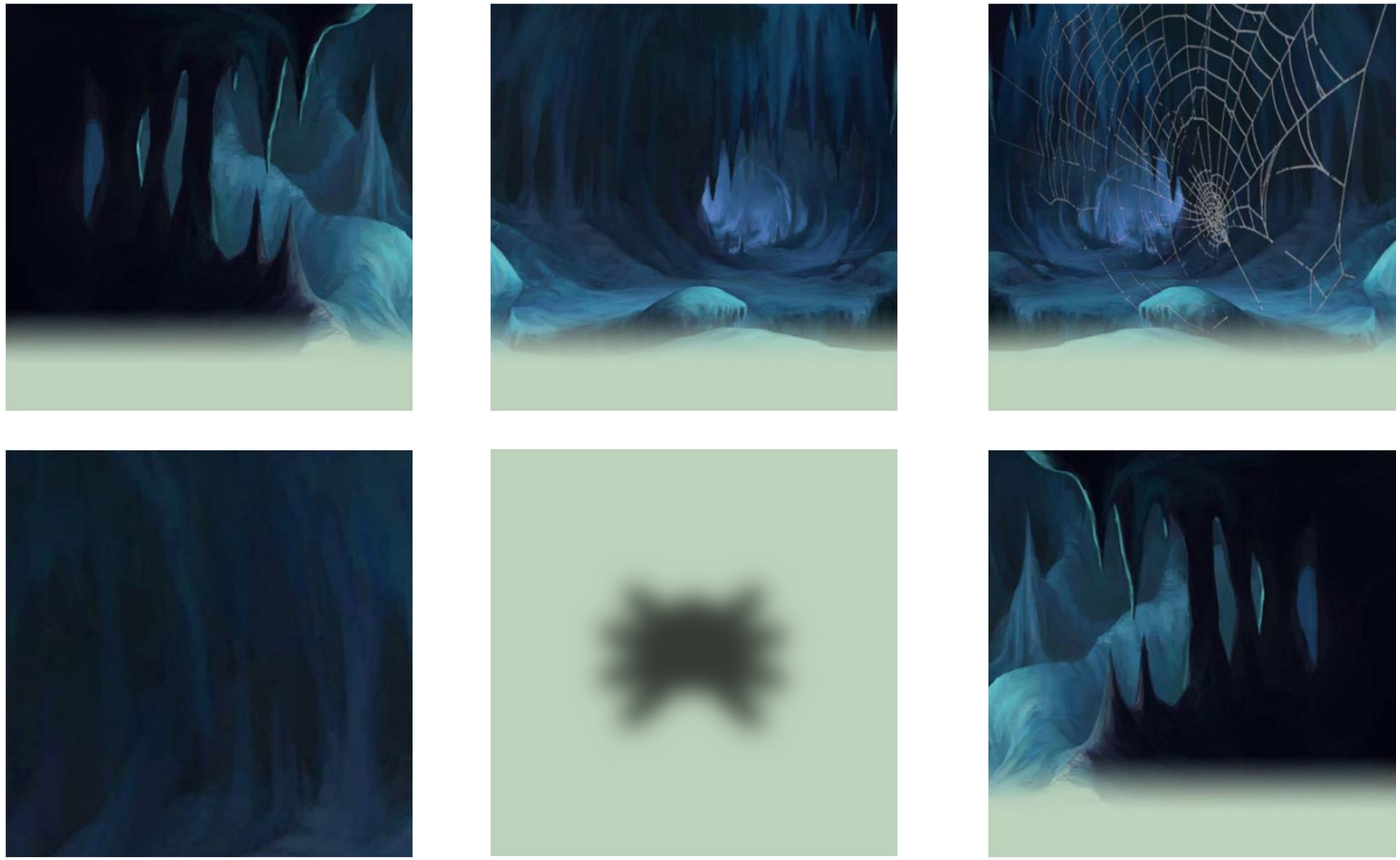
```
UINT CGLRenderer::LoadTexture(char* fileName)
{
	UINT texID;
	DImage img;
	img.Load(CString(fileName));
	glPixelStorei(GL_UNPACK_ALIGNMENT, 4);
	glGenTextures(1, &texID);
	glBindTexture(GL_TEXTURE_2D, texID);
	glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_CLAMP_TO_EDGE);
	glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_CLAMP_TO_EDGE);
	glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
	glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER,
                                                        GL_LINEAR_MIPMAP_LINEAR);
	glTexEnvf(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_MODULATE);

	gluBuild2DMipmaps(GL_TEXTURE_2D, GL_RGBA, img.Width(), img.Height(),
		GL_BGRA_EXT, GL_UNSIGNED_BYTE, img.GetDIBBits());

	return texID;
}
```

```
void CGLRenderer::PrepareScene(CDC *pDC)
{
        wglMakeCurrent(pDC->m_hDC, m_hrc);

        glClearColor(1.0, 1.0, 1.0, 1.0);
        glEnable(GL_DEPTH_TEST);

        m_texSpider = LoadTexture("spider.png");

        m_texEnv[0] = LoadTexture("front.jpg");
        m_texEnv[1] = LoadTexture("back.jpg");

        m_texEnv[2] = LoadTexture("left.jpg");
        m_texEnv[3] = LoadTexture("right.jpg");

        m_texEnv[4] = LoadTexture("top.jpg");
        m_texEnv[5] = LoadTexture("bot.jpg");

        glEnable(GL_TEXTURE_2D);

        wglMakeCurrent(NULL, NULL);
}
```

```
void CGLRenderer::DestroyScene(CDC *pDC)
{
        wglMakeCurrent(pDC->m_hDC, m_hrc);

        glDeleteTextures(1, &m_texSpider);
        glDeleteTextures(6, m_texEnv);

        wglMakeCurrent(NULL,NULL);
        if(m_hrc)
        {
                wglDeleteContext(m_hrc);
                m_hrc = NULL;
        }
}
```

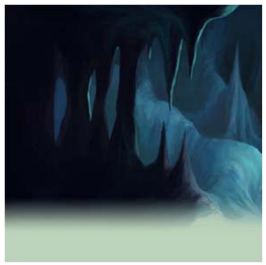
```
UINT m_texSpider;
UINT m_texEnv[6];
```

Napisati funkciju void CGLRenderer::**DrawEnvCube**(double a), koja iscrtava kocku stranice dužine **a**, kojom se uokvirava scena (centrirana na poziciji kamere). Kocka se uvek vidi samo sa unutrašnje strane i na njenim stranicama su „nalepljene“ teksture: front.jpg, left.jpg, right.jpg, back.jpg, top.jpg i bot.jpg. U DrawScene funkciji nacrtati ovu kocku, stranice 100 jedinica. [10 poena]

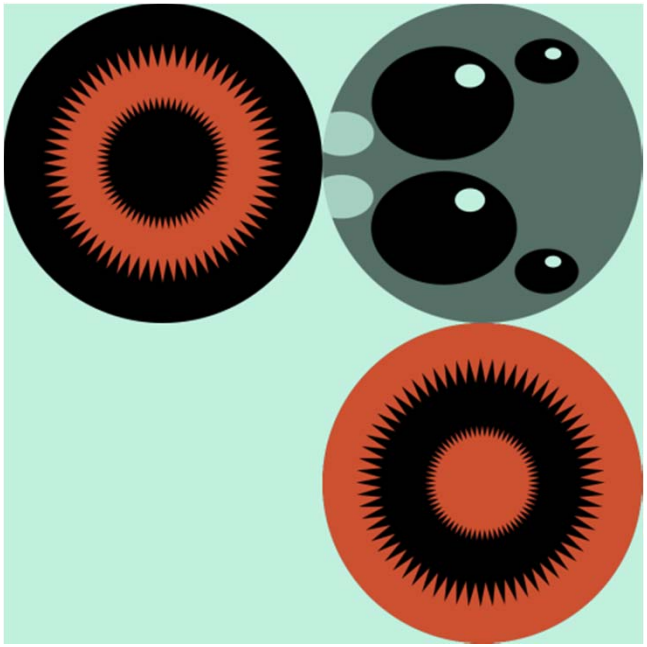


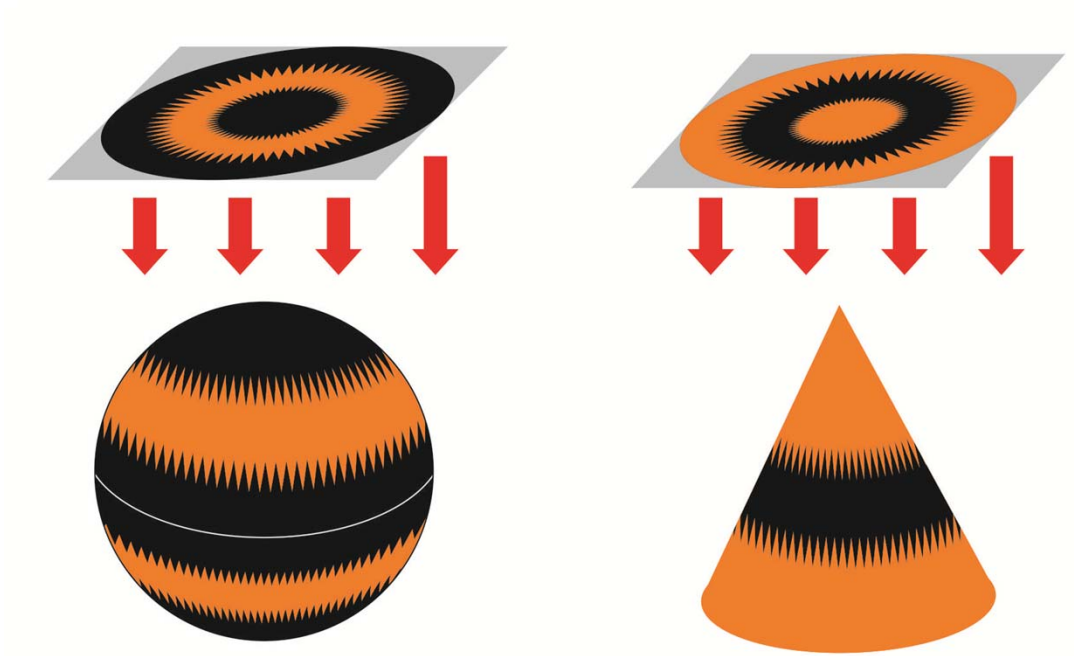
```
void CGLRenderer::DrawEnvCube(double a)
{
	glEnable(GL_TEXTURE_2D);

	// Front
	glBindTexture(GL_TEXTURE_2D,
				m_texEnv[0]);

	glBegin(GL_QUADS);

	glColor3f(1.0, 1.0, 1.0);

	glTexCoord2f(0.0, 0.0);
	glVertex3d(-a / 2, a / 2, -a / 2);

	glTexCoord2f(0.0, 1.0);
	glVertex3d(-a / 2, -a / 2, -a / 2);

	glTexCoord2f(1.0, 1.0);
	glVertex3d(a / 2, -a / 2, -a / 2);

	glTexCoord2f(1.0, 0.0);
	glVertex3d(a / 2, a / 2, -a / 2);
	glEnd();
```

```
	// Back
	glBindTexture(GL_TEXTURE_2D, m_texEnv[1]);


	glBegin(GL_QUADS);

	glColor3f(1.0, 1.0, 1.0);

	glTexCoord2f(1.0, 0.0);
	glVertex3d(-a / 2, a / 2, a / 2);
	glTexCoord2f(1.0, 1.0);
	glVertex3d(-a / 2, -a / 2, a / 2);
	glTexCoord2f(0.0, 1.0);
	glVertex3d(a / 2, -a / 2, a / 2);
	glTexCoord2f(0.0, 0.0);
	glVertex3d(a / 2, a / 2, a / 2);
	glEnd();

	…
```

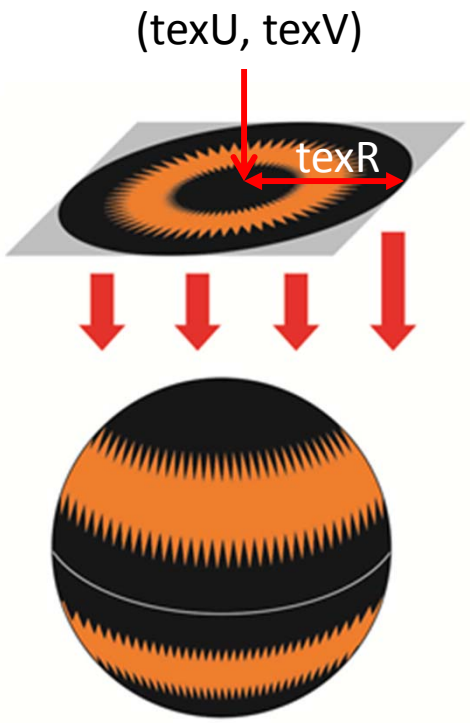


Napisati funkciju void CGLRenderer::**DrawSphere**(double r, int nSeg, double texU, double texV, double texR), koja iscrtava sferu poluprečnika **r**, sa **nSeg** segmenata po latitudi i **2*nSeg** segmenata po longitudi. Na sferu se primenjuje tekstura, tako što se „projektuje“ odozgo (vidi sliku). **texU** i **texV** predstavljaju teksturne koordinate „vrha“ sfere, a **texR** poluprečnik kruga u teksturnim koordinatama koji se primenjuje na sferu (vidi sliku). [15 poena]





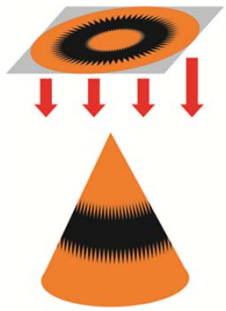
```
void CGLRenderer::DrawSphere(double r,int nSeg, double texU,double texV, double texR)
{
        int i, j;
        double ang1, ang2;
        double dAng1, dAng2;
        dAng1 = PI /(double)nSeg;
        dAng2 = 2 * PI / (double)nSeg;
        ang1 = -PI / 2;

        for (i = 0; i < nSeg; i++) {

                ang2 = 0;

                glBegin(GL_QUAD_STRIP);

                for (j = 0; j < nSeg+1; j++) {

                        double x1 = r * cos(ang1) * cos(ang2);
                        double y1 = r * sin(ang1);
                        double z1 = r * cos(ang1) * sin(ang2);

                        double x2 = r * cos(ang1 + dAng1) * cos(ang2);
                        double y2 = r * sin(ang1 + dAng1);
                        double z2 = r * cos(ang1 + dAng1) * sin(ang2);
```




```
                double tx1 = texR *x1 / r + texU;
                double ty1 = texR *z1 / r + texV;

                double tx2 = texR *x2 / r + texU;
                double ty2 = texR *z2 / r + texV;

                glTexCoord2f(tx1, ty1);
                glVertex3d(x1, y1, z1);
                glTexCoord2f(tx2, ty2);
                glVertex3d(x2, y2, z2);

                ang2 += dAng2;

            }

            glEnd();

            ang1 += dAng1;

        }

    }
```

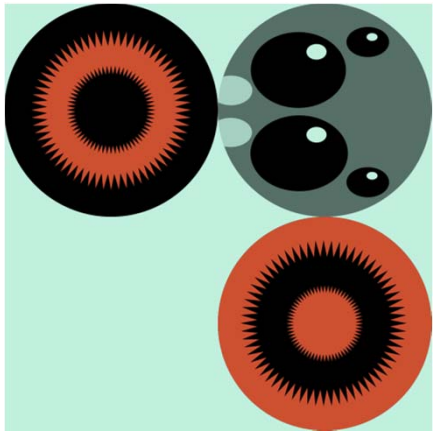
Napisati funkciju void CGLRenderer::**DrawCone**(double r, double h, int nSeg, double texU, double texV, double texR), koja iscrtava kupu poluprečnika **r**, aproksimiranu sa **nSeg**, i visine **h**. **texU** i **texV** predstavljaju teksturne koordinate vrha kupe, a **texR** poluprečnik kruga u teksturnim koordinatama koji se primenjuje na kupu (vidi sliku). [15 poena]

```
void CGLRenderer::DrawCone(double r, double h,
int nSeg, double texU, double texV, double texR)
{
        int i;
        double ang1, dAng1;
        double ;

        dAng1 = 2 * PI / (double)nSeg;

        ang1=0;

        glBegin(GL_TRIANGLE_FAN);

        glColor3f(1.0,1.0,1.0);
        glTexCoord2f(texU, texV);
        glVertex3d(0.0, h, 0.0);

        for (i = 0; i < nSeg + 1; i++) {

                double x = r * cos(ang1);
                double y = 0;
                double z = r * sin(ang1);

                double tx = texR *x / r + texU;
                double ty = texR *z / r + texV;

                glTexCoord2f(tx, ty);
                glVertex3d(x, y, z);

                ang1 += dAng1;
        }

        glEnd();

}
```



Napisati funkciju void CGLRenderer::**DrawLegSegment**(double r, double h, int nSeg), koja iscrtava jedan segment noge pauka, pozivom prethodno definisanih funkcija DrawSphere i DrawCone. Poluprečnici sfere i kupe su **r**, visina kupe **h**, a broj segmenata sfere i kupe su **2*nSeg** i **nSeg**, respektivno. Ostale parametre funkcija za crtanje sfere i kupe uskladiti sa teksturom **spider** koja je priložena (vidi sliku). [5 poena]

```
void CGLRenderer::DrawLegSegment(double r, double h, int nSeg) {

        glPushMatrix();

        glTranslatef(0, r, 0);
        DrawSphere(r, 2*nSeg, 0.25, 0.25, 0.24);
        DrawCone(r, h, nSeg, 0.75, 0.75, 0.24);

        glPopMatrix();

}
```

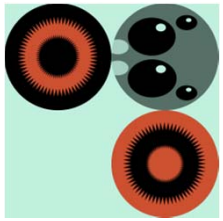


Napisati funkciju void CGLRenderer::**DrawLeg**(), koja iscrtava nogu pauka, sastavljenu od dva segmenta (DrawLegSegment) koji se nastavljaju jedan na drugi. Oba segmenta treba da imaju poluprečnik 1 i broj segmenata 5. Prvi segment je dužine 10, a drugi 15 i zarotiran je u odnosu na početni položaj za 85°. [5 poena]

```
void CGLRenderer::DrawLeg() {

        double ang = 85;

        glPushMatrix();

        DrawLegSegment(1, 10, 5);
        glTranslatef(0, 11, 0);
        glRotatef(ang,1.0,0.0,0.0);
        DrawLegSegment(1, 15, 5);

        glPopMatrix();

}
```

Napisati funkciju void CGLRenderer::**DrawSpiderBody**(), koja iscrtava telo pauka, sastavljenu od tri sfere (DrawSphere): glave, grudnog dela i stomaka, poluprečnika 2, 3 i 5, respektivno. Broj segmenata za sve tri sfere postaviti na 10, a teksturne koordinate tako da se grudni deo i stomak mapiraju na gornju levu četvrtinu, a glava na gornju desnu četvrtinu teksture **spider**. Glava i grudni deo su „spljošteni“ na 50%, a stomak na 80% po jednoj od osa. [10 poena]

```
void CGLRenderer::DrawSpiderBody() {

        double ang = 45;

        glPushMatrix();

        /// CENTER
        glPushMatrix();
        glScalef(1.0, 0.5, 1.0);
        DrawSphere(3,10,0.25,0.25,0.24);
        glPopMatrix();

        /// TAIL
        glPushMatrix();
        glTranslatef(6.0, 0.0, 0.0);
        glScalef(1.0, 0.8, 1.0);
        DrawSphere(5, 10, 0.25, 0.25, 0.24);
        glPopMatrix();

        /// HEAD
        glPushMatrix();
        glTranslatef(-4.0, 0.0, 0.0);
        glScalef(1.0, 0.5, 2.0);
        DrawSphere(2,10, 0.75, 0.25, 0.24);
        glPopMatrix();

        glPopMatrix();
```



```
}
```

Napisati funkciju void CGLRenderer::**DrawSpider**(), koja iscrtava celog pauka, sastavljenog od tela i 8 nogu. Noge polaze iz sredine grudnog dela. Prvi segmenti su zarotirani za 45° u odnosu na grudni deo i 30° jedna u odnosu na drugu (vidi sliku). [10 poena]

```
void CGLRenderer::DrawSpider() {

        int i;

        glEnable(GL_TEXTURE_2D);
        glBindTexture(GL_TEXTURE_2D, m_texSpider);

        glPushMatrix();

          glTranslatef(0.0,5.0, 0.0);

          glPushMatrix();
                DrawSpiderBody();
          glPopMatrix();

          for (i = 0; i < 4; i++) {
              glPushMatrix();
                  glRotatef(i*30 - 45, 0.0, 1.0, 0.0);
                  glRotatef(45, 1.0, 0.0, 0.0);
                  DrawLeg();
              glPopMatrix();
          }

          for (i = 0; i < 4; i++) {
            glPushMatrix();
               glRotatef(i*30 -45+180, 0.0, 1.0, 0.0);
               glRotatef(45, 1.0, 0.0, 0.0);
               DrawLeg();
            glPopMatrix();

          }

          glPopMatrix();

          glEnable(GL_TEXTURE_2D);

}
```

Iscrtati celu scenu (pauka na sredini donje stranice kocke okruženja, vidi sliku) u funkciji **DrawScene** i omogućiti njeno animiranje tako da pritisak na taster:

→ – rotira posmatrača oko Y-ose udesno oko centra scene (tačka [0,10,0]),
← – rotira posmatrača oko Y-ose ulevo oko centra scene (tačka [0,10,0]),
↑ – rotira posmatrača naviše,
↓ – rotira posmatrača naniže (ne dozvoliti prolaz kroz podlogu),
+ – približava posmatrača centru scene (maksimalno udaljenje je 50),
– – udaljava posmatrača od centra scene (minimalno udaljenje je 8) [10 poena]

```
void CGLRenderer::DrawScene(CDC *pDC)
{
        wglMakeCurrent(pDC->m_hDC, m_hrc);
        glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
        glLoadIdentity();
        UpdateCamera();
        glTranslatef(0, -10, 0);

        DrawSpider();

        glPushMatrix();
        glTranslatef(0.0, 50, 0.0);
        DrawEnvCube(100.0);
        glPopMatrix();
        glFlush();
        SwapBuffers(pDC->m_hDC);
        wglMakeCurrent(NULL, NULL);
}
```

```
void CGLRenderer::UpdateCamera()
{
        double toDeg = 57.29577951308

        glTranslatef(0, 0, -m_CamDist);
        glRotatef(m_CamBeta*toDeg, 1, 0, 0);
        glRotatef(-m_CamAlpha*toDeg, 0, 1, 0);
}
```