ЛАБОРАТОРИЙН АЖИЛ №6

SHADING & LIGHTING & TEXTURING

Лабораторийн ажлын даалгавар:

1. Дурын 3D хэмжээст объектыг сонгон авч зурна.
2. “Lighting” өөрчлөлтүүдийг нэмнэ.

Энэхүү лабораторийн ажлаар өмнөх код дээр гэрэлтүүлгийг тооцоолж, үр дүн гарган авна. Параметр утгыг өөрчлөх замаар жишээ кодыг бүрэн эзэмшинэ үү.

Жишээ:

```
/*
 * material.c
 * Энэүү програм нь GL гэрэлтүүлгийн моделийн хэрэглээг харуулдаг.
 * Янз бүрийн материалын шинж чанаруудыг ашиглан хэд хэдэн объектыг зурдаг.
 * Дан ганц light source объектийг гэрэлтүүлнэ.
 */
#include <cstdlib>
#include <GL/glut.h>

/*  z-buffer, projection matrix, light source болон lighting model Initialize
хийнэ.
* Энд материалын шинж чанарыг тодорхойлохгүй.
*/
void init(void)
{
    GLfloat ambient[] = { 0.0, 0.0, 0.0, 1.0 };
    GLfloat diffuse[] = { 1.0, 1.0, 1.0, 1.0 };
    GLfloat specular[] = { 1.0, 1.0, 1.0, 1.0 };
    GLfloat position1[] = { 0.0, 4.0, 3.0, 1.0 };
    GLfloat position2[] = { 0.0, -4.0, 3.0, 1.0 };
    GLfloat lmodel_ambient[] = { 0.4, 0.4, 0.4, 1.0 };
    GLfloat local_view[] = { 0.0 };

    glClearColor(0.0, 0.1, 0.1, 0.0);
    glEnable(GL_DEPTH_TEST);
    glShadeModel(GL_SMOOTH);

    glLightfv(GL_LIGHT0, GL_AMBIENT, ambient);
    glLightfv(GL_LIGHT0, GL_DIFFUSE, diffuse);
    glLightfv(GL_LIGHT0, GL_POSITION, position1);

    glLightfv(GL_LIGHT1, GL_AMBIENT, ambient);
    glLightfv(GL_LIGHT1, GL_DIFFUSE, diffuse);
    glLightfv(GL_LIGHT1, GL_POSITION, position2);

    glLightModelfv(GL_LIGHT_MODEL_AMBIENT, lmodel_ambient);
    glLightModelfv(GL_LIGHT_MODEL_LOCAL_VIEWER, local_view);

    glEnable(GL_LIGHTING);
    glEnable(GL_LIGHT0);
    glEnable(GL_LIGHT1);
}


//void init(void)
//{
// GLfloat light_position[] = {0.0, 2.0, 3.0, 1.0};
// GLfloat light_position2[] = {0.0, -2.0, 3.0, 1.0};
//
// glClearColor (0.0, 0.0, 0.0, 0.0);
```

```
// glShadeModel (GL_SMOOTH);
//
// glEnable(GL_LIGHTING);
// glEnable(GL_LIGHT0);
// glEnable(GL_LIGHT1);
//
// glLightfv(GL_LIGHT0, GL_POSITION, light_position);
// glLightfv(GL_LIGHT1, GL_POSITION, light_position2);
// glEnable(GL_DEPTH_TEST);
//}

void display(void)
{

   glClear (GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

   GLfloat no_mat[] = { 0.0, 0.0, 0.0, 1.0 };
   GLfloat mat_ambient[] = { 0.7, 0.7, 0.7, 1.0 };
   GLfloat mat_ambient_color[] = { 0.8, 0.8, 0.2, 1.0 };
   GLfloat mat_diffuse[4] = { 0.0, 0.0, 0.0, 1.0 };
   GLfloat mat_specular[] = { 1.0, 1.0, 1.0, 1.0 };
   GLfloat no_shininess[] = { 0.0 };
   GLfloat low_shininess[] = { 5.0 };
   GLfloat high_shininess[] = { 100.0 };
   GLfloat mat_emission[] = {0.3, 0.2, 0.2, 0.0};

   glPushMatrix();
   glTranslatef(-3,0,0);
   glMaterialfv(GL_FRONT, GL_AMBIENT, no_mat);
   mat_diffuse[1] = 0.5;
   glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_diffuse);
   glMaterialfv(GL_FRONT, GL_SPECULAR, no_mat);
   glMaterialfv(GL_FRONT, GL_SHININESS, no_shininess);
   glMaterialfv(GL_FRONT, GL_EMISSION, no_mat);
   glScalef(1.5,1.5,1.5);
   glutSolidSphere (1, 20, 16);
   glPopMatrix();

   glPushMatrix();
   glTranslatef(3,0,0);
   glMaterialfv(GL_FRONT, GL_AMBIENT, no_mat);
   mat_diffuse[1] = 0.0;
   mat_diffuse[2] = 0.5;
   glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_diffuse);
   glMaterialfv(GL_FRONT, GL_SPECULAR, mat_specular);
   glMaterialfv(GL_FRONT, GL_SHININESS, high_shininess);
   glMaterialfv(GL_FRONT, GL_EMISSION, no_mat);
   glutSolidSphere (1, 20, 16);
   glPopMatrix();

   glFlush ();
}

void reshape (int w, int h)
{
    glViewport(0, 0, w, h);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    if (w <= (h * 2))
           glOrtho (-6.0, 6.0, -3.0*((GLfloat)h*2)/(GLfloat)w,
            3.0*((GLfloat)h*2)/(GLfloat)w, -10.0, 10.0);
    else
           glOrtho (-6.0*(GLfloat)w/((GLfloat)h*2),
         6.0*(GLfloat)w/((GLfloat)h*2), -3.0, 3.0, -10.0, 10.0);
    glMatrixMode(GL_MODELVIEW);
```

```
    glLoadIdentity();
}

void keyboard(unsigned char key, int x, int y)
{
   switch (key) {
      case 27:
         exit(0);
         break;
   }
}

int main(int argc, char** argv)
{
   glutInit(&argc, argv);
   glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB | GLUT_DEPTH);
   glutInitWindowSize (500, 500);
   glutInitWindowPosition (100, 100);
   glutCreateWindow (argv[0]);
   init ();
   glutDisplayFunc(display);
   glutReshapeFunc(reshape);
   glutKeyboardFunc(keyboard);
   glutMainLoop();
   return 0;
}
```

3. Сүүдэрлэлт оруулна.
4. Texture оруулна.

Энэ дасгалд өмнөх код дээр үндэслэн тухайн орчны өнгө (specular colour) болон объектын эрчмийг тус тусад нь ялгах ambient болон diffuse тодорхойлсон 2 texture нэмж оруулах шаардлагатай.