Компьютерын ухаан

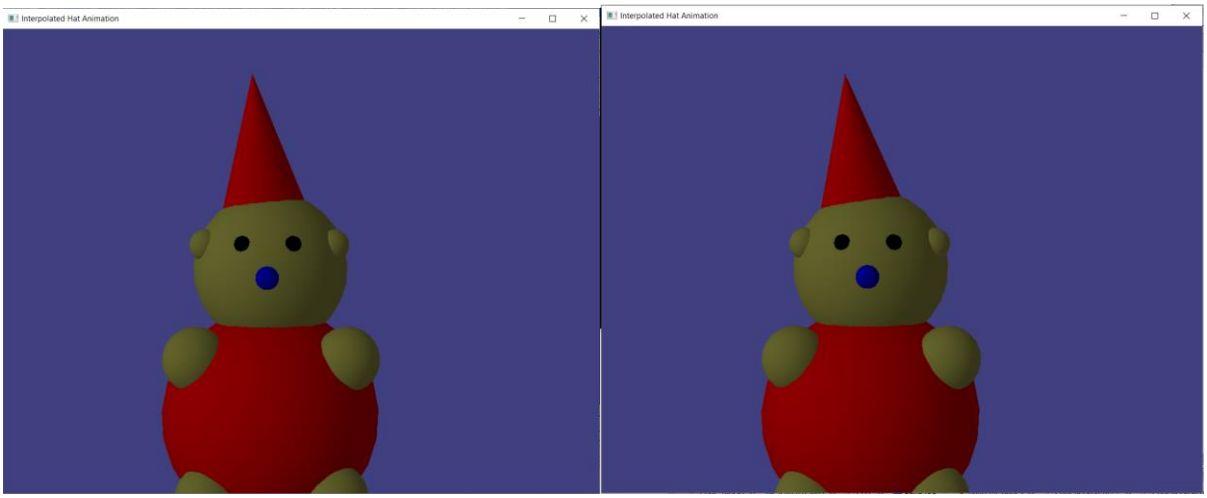
А. Гаусар 21b1num1182

Лабораторі №8

GLU сангийн quadric гадаргууг ашиглан ямар нэг биетийн 3-хэмжээст загварыг байгуулж, хөдөлгөөнд оруулах

Дараах загвартай 3D модел байгуулсан бөгөөд үүнийг малгайг нэг тал руугаа унаж байгаа мэтээр tweening арга ашиглаад хөдөлгөөн оруулсан.

```c
#include "GL/freeglut.h"

static GLfloat xRot = 0.0f;
static GLfloat yRot = 0.0f;
static float t = 0.0f;

void ChangeSize(int w, int h) {
    GLfloat fAspect;

    if (h == 0) h = 1;

    glViewport(0, 0, w, h);

    fAspect = (GLfloat)w / (GLfloat)h;

    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluPerspective(35.0f, fAspect, 1.0, 40.0);
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
}

void SpecialKeys(int key, int x, int y) {
    if (key == GLUT_KEY_UP) xRot -= 5.0f;
    if (key == GLUT_KEY_DOWN) xRot += 5.0f;
    if (key == GLUT_KEY_LEFT) yRot -= 5.0f;
    if (key == GLUT_KEY_RIGHT) yRot += 5.0f;

    xRot = (GLfloat)((const int)xRot % 360);
    yRot = (GLfloat)((const int)yRot % 360);
    glutPostRedisplay();
}

void interpolatePosition(float t, float Ax, float Ay, float Az, float Bx, float By, float Bz, float &Px, float &Py, float &Pz) {
    Px = (1 - t) * Ax + t * Bx;
    Py = (1 - t) * Ay + t * By;
    Pz = (1 - t) * Az + t * Bz;
}
```



```
void RenderScene(void) {
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    GLUquadricObj *pObj = gluNewQuadric();
    gluQuadricNormals(pObj, GLU_SMOOTH);

    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    glPushMatrix();
    glTranslatef(0.0, -0.2, -5.0);
    glRotatef(yRot, 0.0, 1.0, 0.0);
    glRotatef(xRot, 1.0, 0.0, 0.0);

    // Variables for hat positions
    float Ax = -0.2, Ay = 0.5, Az = 0.2; // "hazaisan malgai"
    float Bx = -0.3, By = 0.4, Bz = 0.2; // "tsaah malgai"
    float hatX, hatY, hatZ;

    interpolatePosition(t, Ax, Ay, Az, Bx, By, Bz, hatX, hatY, hatZ);

    // "malgai"
    glPushMatrix();
    glTranslatef(hatX, hatY, hatZ);
    glRotatef((1 - t) * -75 + t * -90, 1.0, t, t);
    glColor3f(1, 0.0, 0.0);
    gluCylinder(pObj, 0.3, 0.0, 0.93, 30, 30);
    glPopMatrix();

    //tsagaan sphere tolgoi
    glPushMatrix();
        glTranslatef(-0.2, 0.15, 0.2);
        glColor3f(0.7, 0.7, 0.3);
        gluSphere(pObj , 0.5 , 25 ,25);
    glPopMatrix();

    //ih bie
    glPushMatrix();
        glTranslatef(-0.2, -0.8, 0.2);
        glColor3f(1.0, 0.0, 0.0);
        gluSphere(pObj , 0.7 , 25 ,25);
    glPopMatrix();
```

```
     //hamar
     glPushMatrix();
         glTranslatef(-0.2, 0.1, 0.7);
         glColor3f(0.0, 0.0, 1.0);
         gluSphere(pObj , 0.07 , 15 ,15);
     glPopMatrix();

     //nud
     glPushMatrix();
         glTranslatef(-0.05, 0.3, 0.64);
         glColor3f(0.0, 0.0, 0.0);
         gluSphere(pObj , 0.05 , 15 ,15);
     glPopMatrix();
     glPushMatrix();
         glTranslatef(-0.35, 0.3, 0.64);
         glColor3f(0.0, 0.0, 0.0);
         gluSphere(pObj , 0.05 , 15 ,15);
     glPopMatrix();

     //chih
     glPushMatrix();
         glTranslatef(0.2, 0.3, 0.4);
         glColor3f(0.7, 0.7, 0.3);
         gluSphere(pObj , 0.1 , 15 ,15);
     glPopMatrix();
     glPushMatrix();
         glTranslatef(-0.6, 0.3, 0.4);
         glColor3f(0.7, 0.7, 0.3);
         gluSphere(pObj , 0.1 , 15 ,15);
     glPopMatrix();

     //hol
     glPushMatrix();
         glTranslatef(0.2, -1.2, 0.64);
         glColor3f(0.7, 0.7, 0.3);
         gluSphere(pObj , 0.2 , 25 ,25);
     glPopMatrix();
     glPushMatrix();
```

```
    glPushMatrix();
        glTranslatef(-0.55, -1.2, 0.64);
        glColor3f(0.7, 0.7, 0.3);
        gluSphere(pObj , 0.2 , 25 ,25);
    glPopMatrix();

    //gar
    glPushMatrix();
        glTranslatef(0.28, -0.4, 0.5);
        glColor3f(0.7, 0.7, 0.3);
        gluSphere(pObj , 0.2 , 25 ,25);
    glPopMatrix();
    glPushMatrix();
        glTranslatef(-0.65, -0.4, 0.5);
        glColor3f(0.7, 0.7, 0.3);
        gluSphere(pObj , 0.2 , 25 ,25);
    glPopMatrix();

    //ar tal ai
    glPushMatrix();
        glTranslatef(-0.2, -1.2, -0.4);
        glColor3f(0.7, 0.7, 0.3);
        gluSphere(pObj , 0.2 , 25 ,25);
    glPopMatrix();

    glPopMatrix();
    glutSwapBuffers();
}

void TimerFunction(int value) {
    t += 0.01f;
    if (t > 1.0f) t = 0.0f;
    glutPostRedisplay();
    glutTimerFunc(33, TimerFunction, 1);
}
```

```
void SetupRC() {
    GLfloat whiteLight[] = {0.05f, 0.05f, 0.05f, 1.0f};
    GLfloat sourceLight[] = {0.25f, 0.25f, 0.25f, 1.0f};
    GLfloat lightPos[] = {-10.f, 5.0f, 5.0f, 1.0f};

    glEnable(GL_DEPTH_TEST);
    glFrontFace(GL_CCW);
    glEnable(GL_CULL_FACE);
    glEnable(GL_LIGHTING);

    glLightModelfv(GL_LIGHT_MODEL_AMBIENT, whiteLight);
    glLightfv(GL_LIGHT0, GL_AMBIENT, sourceLight);
    glLightfv(GL_LIGHT0, GL_DIFFUSE, sourceLight);
    glLightfv(GL_LIGHT0, GL_POSITION, lightPos);
    glEnable(GL_LIGHT0);

    glEnable(GL_COLOR_MATERIAL);
    glColorMaterial(GL_FRONT, GL_AMBIENT_AND_DIFFUSE);

    glClearColor(0.25f, 0.25f, 0.50f, 1.0f);
}

int main(int argc, char *argv[]) {
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);
    glutInitWindowSize(900, 700);
    glutCreateWindow("Interpolated Hat Animation");
    glutReshapeFunc(ChangeSize);
    glutSpecialFunc(SpecialKeys);
    glutDisplayFunc(RenderScene);
    SetupRC();
    glutTimerFunc(33, TimerFunction, 1);
    glutMainLoop();
    return 0;
}
```