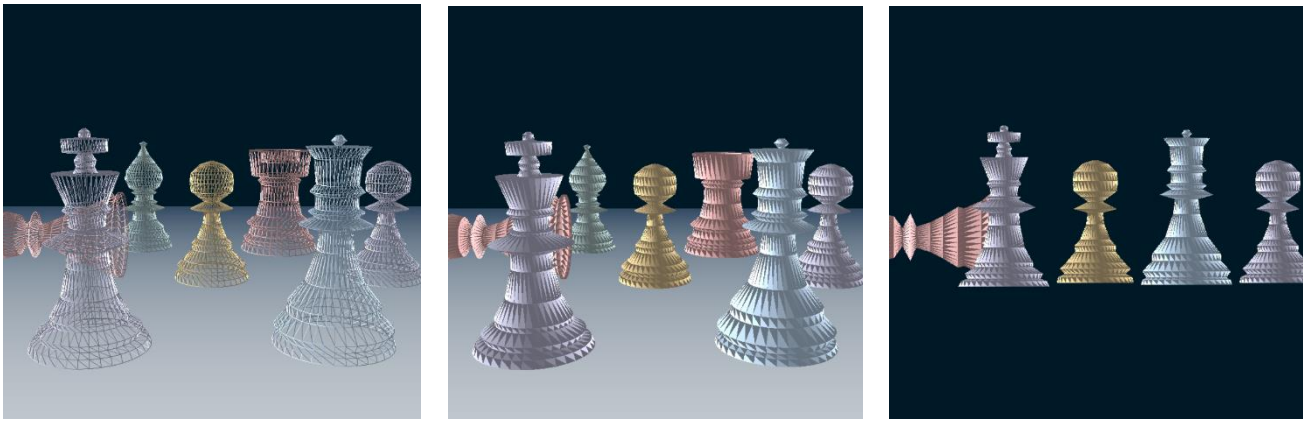
Final project

컴퓨터그래픽스02분반_20203180_송정후

Ⅰ. 서론

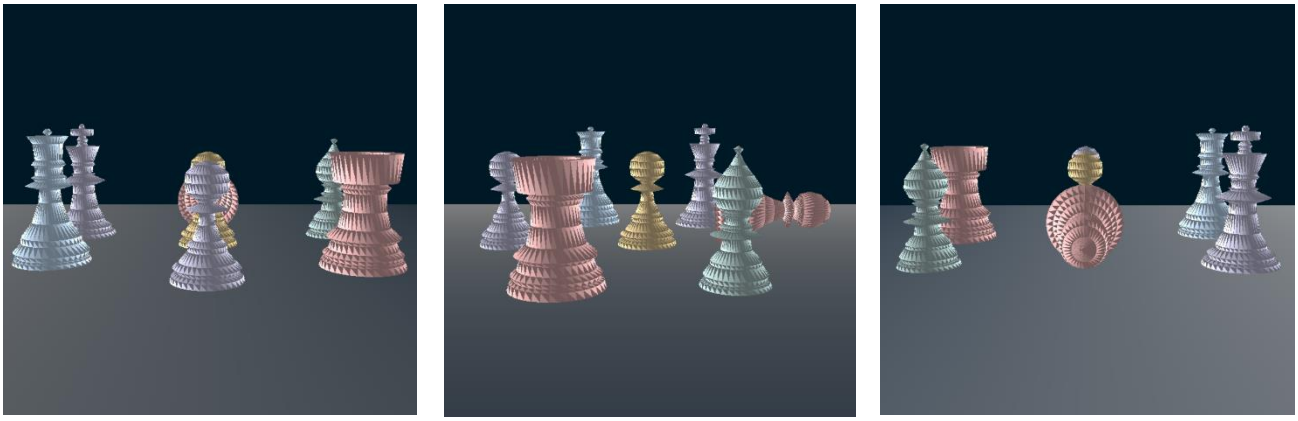
점을 찍고 X축, Y축으로 회전시켜 만든 3D모델들을 이용하여 가상공간을 만들어 보았다. 키보드와 마우스 조작을 통해 렌더링 스타일, 투영 방법, 시점을 변환시킬 수 있다.

1. 키보드(w: wire / s: shading / p: 투영방법 변환)



2. 방향키(←: 왼쪽으로 이동 / →: 오른쪽으로 이동 / ↑: 앞으로 이동 / ↓: 뒤로 이동)

3. 마우스(LEFT: 왼쪽으로 90도 회전 / RIGHT: 오른쪽으로 90도 회전)



Ⅱ. 본론

1. save model

클릭한 점을 회전시켜 만든 회전체를 저장해보자. 회전체의 ①꼭짓점의 좌표와 ②꼭짓점들을 연결하여 만든 삼각형을 저장해보도록 하자. clickpt는 클릭한 점들이 저장되어 있고, newpt는 clickpt를 포함하여 회전체의 모든 꼭짓점이 저장되어 있다.

1-1. mface에 삼각형 저장

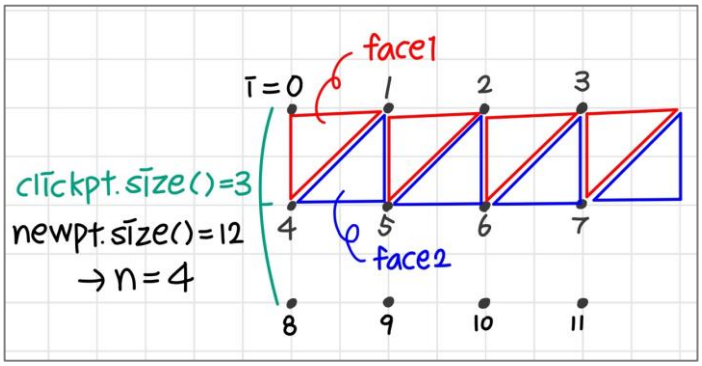
```
typedef struct { unsigned int ip[3]; } Face;
std::vector<Face> mface;

void SaveModel()
{
    int n = newpt.size() / clickpt.size();
    Face face1; Face face2;
    for (int i = 0; i < newpt.size() - n; i++)
    {
        if (i % n != n - 1) {
            face1.ip[0] = i;
            face1.ip[1] = i + 1;
            face1.ip[2] = i + n;

            face2.ip[0] = i + 1;
            face2.ip[1] = i + n;
            face2.ip[2] = i + n + 1;
        }
        else {
            face1.ip[0] = i;
            face1.ip[1] = i + 1 - n;
            face1.ip[2] = i + n;

            face2.ip[0] = i + 1 - n;
            face2.ip[1] = i + n;
            face2.ip[2] = i + 1;
        }
        mface.push_back(face1);
        mface.push_back(face2);
    }
```

먼저 mface에 삼각형을 저장해보자. Face라는 struct를 만들어 세개의 꼭짓점을 리스트로 저장할 수 있게 하였다.



n은 newpt개수 / clickpt개수이고, 위의 그림에선 n=4이다. 위의 그림에서 i > 7일 때는 삼각형을 만들 수 없으므로 i가 newpt.size() - n만큼 돌게 했다. i = 0일 때 그림처럼 face1(0, 1, 4)과 face2(1, 4, 5)를 만들었다. i가 3, 7일 때 (i % n = n - 1)처럼 예외가 발생하는 경우에는 else를 이용해 잘 연결될 수 있게 했다.

1-2. 점과 삼각형 저장

```
FILE* fout;
fopen_s(&fout, "c:\\data\\pawn.dat", "w");

fprintf(fout, "VERTEX = %d\n", newpt.size());
for (int i = 0; i < newpt.size(); i++)
{
    fprintf(fout, "%.1f %.1f %.1f\n", newpt[i].x, newpt[i].y, newpt[i].z);
}

fprintf(fout, "FACE = %d\n", mface.size());
for (int i = 0; i < mface.size(); i++)
{
    fprintf(fout, "%d %d %d\n", mface[i].ip[0], mface[i].ip[1], mface[i].ip[2]);
}
fclose(fout);
```

pawn이라는 이름으로 파일을 저장해보자. newpt에 회전체의 꼭짓점이 모두 저장되어 있으므로 VERTEX는 newpt.size()이고, for을 이용하여 꼭짓점의 x좌표, y좌표, z좌표를 차례로 저장하였다. mface를 이용하여 삼각형 리스트를 저장하였다.

2. Read Model

```
using namespace std;
typedef struct { float x; float y; float z; } Point;
typedef struct { unsigned int ip[3]; } Face;
int pnum; int fnum;
Point* mpoint = NULL;
Face* mface = NULL;
```

ReadModel함수는 파일 주소(fname)를 입력받아 파일에 담긴 3d모델을 mpoint와 mface에 저장하는 함수이다.

```
void ReadModel(string fname)
{
    FILE* f1; char s[80]; int i;
    if (mpoint != NULL) delete mpoint;
    if (mface != NULL) delete mface;
    if ((f1 = fopen(fname.c_str(), "rt")) == NULL) { printf("No file₩n"); exit(0); }
    fscanf(f1, "%s", s);  fscanf(f1, "%s", s);
    fscanf(f1, "%d", &pnum);
    mpoint = new Point[pnum];
    for (i = 0; i < pnum; i++) {
        fscanf(f1, "%f", &mpoint[i].x); fscanf(f1, "%f", &mpoint[i].y); fscanf(f1, "%f", &mpoint[i].z);
    }
    fscanf(f1, "%s", s);
    fscanf(f1, "%s", s);
    fscanf(f1, "%d", &fnum);
    mface = new Face[fnum];
    for (i = 0; i < fnum; i++) {
        fscanf(f1, "%d", &mface[i].ip[0]); fscanf(f1, "%d", &mface[i].ip[1]); fscanf(f1, "%d", &mface[i].ip[2]);
    }
    fclose(f1);
}
```

3. Make GL Model

```
void MakeGL_Model(void)
{
    glShadeModel(GL_SMOOTH);
    if (glIsList(1)) glDeleteLists(1, 1);
    glNewList(1, GL_COMPILE);
    glScalef(0.6, 0.6, 0.6);
    if (view == 0) glTranslatef(x_position, 0, z_position);
    if (view == 1) glTranslatef(z_position, 0, -x_position);
    if (view == 2) glTranslatef(-x_position, 0, -z_position);
    if (view == 3) glTranslatef(-z_position, 0, x_position);

    for (int i = 0; i < fnum; i++) {
        Point norm = cnormal(mpoint[mface[i].ip[2]], mpoint[mface[i].ip[1]], mpoint[mface[i].ip[0]]);
        glBegin(GL_TRIANGLES);
        glNormal3f(norm.x, norm.y, norm.z);
        glVertex3f(mpoint[mface[i].ip[0]].x, mpoint[mface[i].ip[0]].y, mpoint[mface[i].ip[0]].z);
        glNormal3f(norm.x, norm.y, norm.z);
        glVertex3f(mpoint[mface[i].ip[1]].x, mpoint[mface[i].ip[1]].y, mpoint[mface[i].ip[1]].z);
        glNormal3f(norm.x, norm.y, norm.z);
        glVertex3f(mpoint[mface[i].ip[2]].x, mpoint[mface[i].ip[2]].y, mpoint[mface[i].ip[2]].z);
        glEnd();
    }
    glEndList();
}
```

MakeGL_Model함수는 mpoint, mface를 이용하여 3d모델을 만들고 리스트에 저장하는 함수이다. 이때 cnormal은 삼각형의 법선벡터를 반환한다.

4. display

```
void display(void)
{
    glClearColor(0, 0.1, 0.15, 1.0);
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glEnable(GL_COLOR_MATERIAL);

    glEnable(GL_DEPTH_TEST);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    if (projection == 0) gluPerspective(40.0, 1.0, 1.0, 2500.0); //원근 투영
    else glOrtho(-350, 350, -350, 350, -1000, 1000); //직교 투영

    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    if (view == 0) gluLookAt(0, y_view, 1000, 0.0, 0.0, 0.0, 0.0, 1.0, 0); //front view
    if (view == 1) gluLookAt(1000, y_view, 0, 0.0, 0.0, 0.0, 0.0, 1.0, 0); //right view
    if (view == 2) gluLookAt(0, y_view, -1000, 0.0, 0.0, 0.0, 0.0, 1.0, 0); //back view
    if (view == 3) gluLookAt(-1000, y_view, 0, 0.0, 0.0, 0.0, 0.0, 1.0, 0); //left view

    if (status == 0) glPolygonMode(GL_FRONT_AND_BACK, GL_LINE); //wire
    else glPolygonMode(GL_FRONT_AND_BACK, GL_FILL); //shading
```

```
    glPushMatrix();
    glPushMatrix();
    ReadModel("c:₩₩data₩₩pawn.dat");
    glColor3f(0.9, 0.8, 0.6);
    MakeGL_Model();
    glCallList(1);
    glPopMatrix();

    glPushMatrix();
    ReadModel("c:₩₩data₩₩pawn2.dat");
    glTranslatef(-300, -40, 0);
    glColor3f(0.9, 0.7, 0.7);
    MakeGL_Model();
    glCallList(1);
    glPopMatrix();

    glPushMatrix();
    ReadModel("c:₩₩data₩₩pawn.dat");
    glTranslatef(300, 0, 0);
    glColor3f(0.8, 0.8, 0.9);
    MakeGL_Model();
    glCallList(1);
    glPopMatrix();

    glPushMatrix();
    ReadModel("c:₩₩data₩₩bishop.dat");
    glTranslatef(-150, 0, -300);
    glColor3f(0.7, 0.8, 0.8);
    MakeGL_Model();
    glCallList(1);
    glPopMatrix();

    glPushMatrix();
    ReadModel("c:₩₩data₩₩rook.dat");
    glTranslatef(150, 0, -300);
    glColor3f(0.9, 0.7, 0.7);
    MakeGL_Model();
    glCallList(1);
    glPopMatrix();

    glPushMatrix();
    ReadModel("c:₩₩data₩₩king.dat");
    glTranslatef(-150, 0, 300);
    glColor3f(0.8, 0.8, 0.9);
    MakeGL_Model();
    glCallList(1);
    glPopMatrix();

    glPushMatrix();
    ReadModel("c:₩₩data₩₩queen.dat");
    glTranslatef(150, 0, 300);
    glColor3f(0.8, 0.9, 1.0);
    MakeGL_Model();
    glCallList(1);
    glPopMatrix();

    glPushMatrix();
    glPolygonMode(GL_FRONT_AND_BACK, GL_FILL);
    glBegin(GL_POLYGON);
    glColor3f(1, 1, 1);
    glVertex3f(1000, -130, 1000);
    glVertex3f(-1000, -130, 1000);
    glColor3f(0.2, 0.3, 0.4);
    glVertex3f(-1000, -130, -1000);
    glVertex3f(1000, -130, -1000);
    glEnd();
    glPopMatrix();
    glPopMatrix();

    glutSwapBuffers();
```

glPushMatix(), glPopMatrix()를 이용하여 7개의 3D모델과 바닥(사각형)을 그려보았다. glColor3f를이용하여 색을 바꿔주었고, glTranslatef를 이용하여 위치를 바꿔주었다.

5. wire VS shading

```
void keyboard(unsigned char key, int x, int y)
{
    switch (key)
    {
    case 'w':
        status = 0; glutPostRedisplay(); break;
    case 's':
        status = 1; glutPostRedisplay(); break;
    case 'p':
        if (projection == 0) { projection = 1; y_view = 0;  glutPostRedisplay(); break; }
        else { projection = 0; y_view = 150; glutPostRedisplay(); break; }
    }
}
```

```
if (status == 0) glPolygonMode(GL_FRONT_AND_BACK, GL_LINE); //wire
else glPolygonMode(GL_FRONT_AND_BACK, GL_FILL); //shading
```

키보드를 이용하여 렌더링 스타일을 바꿔보았다. w를 클릭하면 status = 0(기본값)이고, glPolygonMode를 GL_LINE로 설정해 3D모델이 wire로 표시되게 했다. s를 클릭하면 status = 1이 되고 glPolygonMode를 GL_FILL로 설정해 3D모델이 shading되어 나타나게 했다.

6. 원근투영 VS 직교투영

```
void keyboard(unsigned char key, int x, int y)
{
    switch (key)
    {
    case 'w':
        status = 0; glutPostRedisplay(); break;
    case 's':
        status = 1; glutPostRedisplay(); break;
    case 'p':
        if (projection == 0) { projection = 1; y_view = 0;  glutPostRedisplay(); break; }
        else { projection = 0; y_view = 150; glutPostRedisplay(); break; }
    }
}
```

```
glEnable(GL_DEPTH_TEST);
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
if (projection == 0) gluPerspective(40.0, 1.0, 1.0, 2500.0); //원근 투영
else glOrtho(-350, 350, -350, 350, -1000, 1000); //직교 투영

glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
if (view == 0) gluLookAt(0, y_view, 1000, 0.0, 0.0, 0.0, 0.0, 1.0, 0); //front view
if (view == 1) gluLookAt(1000, y_view, 0, 0.0, 0.0, 0.0, 0.0, 1.0, 0); //right view
if (view == 2) gluLookAt(0, y_view, -1000, 0.0, 0.0, 0.0, 0.0, 1.0, 0); //back view
if (view == 3) gluLookAt(-1000, y_view, 0, 0.0, 0.0, 0.0, 0.0, 1.0, 0); //left view
```

키보드를 이용하여 투영 방식을 바꿀 수 있게 하였다. p를 클릭하면 projection과 y_view가 바뀐다. projection이 0(기본값)일 때에는 원근 투영, projection이 1일 때에는 직교투영이다. y_view는 눈의 위치의 y좌표값이다. 원근투영일 때에는 150, 직교투영할 때에는 0이 되도록 하였다.

7. 시점 변환

```
void mouse(int button, int state, int x, int y)
{
    if (button == GLUT_RIGHT_BUTTON && state == GLUT_DOWN)
    {
        if (view == 3) view = 0;
        else view += 1;
    }
    if (button == GLUT_LEFT_BUTTON && state == GLUT_DOWN)
    {
        if (view == 0) view = 3;
        else view -= 1;
    }
}
```

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
if (view == 0) gluLookAt(0, y_view, 1000, 0.0, 0.0, 0.0, 0.0, 1.0, 0); //front view
if (view == 1) gluLookAt(1000, y_view, 0, 0.0, 0.0, 0.0, 0.0, 1.0, 0); //right view
if (view == 2) gluLookAt(0, y_view, -1000, 0.0, 0.0, 0.0, 0.0, 1.0, 0); //back view
if (view == 3) gluLookAt(-1000, y_view, 0, 0.0, 0.0, 0.0, 0.0, 1.0, 0); //left view
```

마우스를 이용하여 시점을 변환할 수 있게 하였다. 오른쪽 마우스를 클릭하면 view가 1씩 증가하고, 왼쪽 마우스를 클릭하면 1씩 감소한다. view = 0(기본값)일 때에는 front view이고, view = 1일 때에는 right view이고, view = 2일 때에는 back view이고, view = 3일 때에는 left view이다. gluLookAt을 적절하게 바꿔주었다.

8. 앞, 뒤, 왼쪽, 오른쪽으로 이동

```
void special(int key, int x, int y)
{
    switch (key) {
    case GLUT_KEY_LEFT:
        x_position += 10;
        glutPostRedisplay(); break;
    case GLUT_KEY_RIGHT:
        x_position -= 10;
        glutPostRedisplay(); break;
    case GLUT_KEY_UP:
        z_position += 10;
        glutPostRedisplay(); break;
    case GLUT_KEY_DOWN:
        z_position -= 10;
        glutPostRedisplay(); break;
    }
}
```

```
if (view == 0) glTranslatef(x_position, 0, z_position);
if (view == 1) glTranslatef(z_position, 0, -x_position);
if (view == 2) glTranslatef(-x_position, 0, -z_position);
if (view == 3) glTranslatef(-z_position, 0, x_position);
```

방향키(special)를 이용하여 앞, 뒤, 왼쪽, 오른쪽으로 이동할 수 있게 하였다. 왼쪽 방향키를 클릭하면 모든 3D모델의 x값을 10씩 증가시켜 오른쪽으로 이동하게 하여 왼쪽으로 이동하는 것처럼 보이게 하였다. 오른쪽 방향키를 클릭하면 3D모델의 x값을 10씩 감소시켰고, 위쪽 방향키를 클릭하면 3D모델의 z값을 10씩 증가시키고, 아래쪽 방향키를 클릭하면 3D모델의 z값을 10씩 감소시켰다. 시점이 변하면 방향도 바뀌기 때문에 시점에 따라 적절하게 바꿔주었다.

9. 빛

```
void InitLight() {
    GLfloat mat_diffuse[] = { 0.5, 0.4, 0.3, 1.0 };
    GLfloat mat_specular[] = { 1.0, 1.0, 1.0, 1.0 };
    GLfloat mat_ambient[] = {0.5, 0.5, 0.5, 1.0};
    GLfloat mat_shininess[] = { 50.0 };
    GLfloat light_diffuse[] = { 0.5, 0.5, 0.5, 1.0 };
    GLfloat light_ambient[] = { 0.3, 0.3, 0.3, 1.0 };
    GLfloat light_position[] = { 500, 0, -100, 0.0 };
    glShadeModel(GL_SMOOTH);
    glEnable(GL_LIGHTING);
    glEnable(GL_LIGHT0);
    glLightfv(GL_LIGHT0, GL_POSITION, light_position);
    glLightfv(GL_LIGHT0, GL_DIFFUSE, light_diffuse);
    glLightfv(GL_LIGHT0, GL_AMBIENT, light_ambient);
    glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_diffuse);
    glMaterialfv(GL_FRONT, GL_AMBIENT, mat_ambient);
    glMaterialfv(GL_FRONT, GL_SHININESS, mat_shininess);
}
```

InitLight는 빛은 설정해주는 함수이다.

10. 기타 설정

```
int main(int argc, char** argv)
{
    glutInit(&argc, argv);
    glutInitWindowSize(700, 700);
    glutInitWindowPosition(100, 10);
    glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);
    glutCreateWindow("Final Project");
    glutDisplayFunc(display);
    glutKeyboardFunc(keyboard);
    glutMouseFunc(mouse);
    glutSpecialFunc(special);
    InitLight();
    glutMainLoop();
    return 0;
}
```

window의 크기는 (700, 700)으로 설정해 주었고, 이름은 final project로 설정하였다.

Ⅲ. 결론

중간고사때 제작한 프로그램을 이용하여 다양한 체스말들을 만들어 가상 공간을 꾸며보았다. Y축으로 회전시켜 만들었고, X축으로 회전시켜 쓰러진 체스말도 만들어 보았다. 중간고사때 제작한 프로그램에선 입체적이지 않아서 3D로 안보이고 점이 잘 회전되고 확인할 수 없었는데 투영 방법을 바꾸고, 3D모델에 빛을 비춰주니까 입체적으로 보여 신기했다.