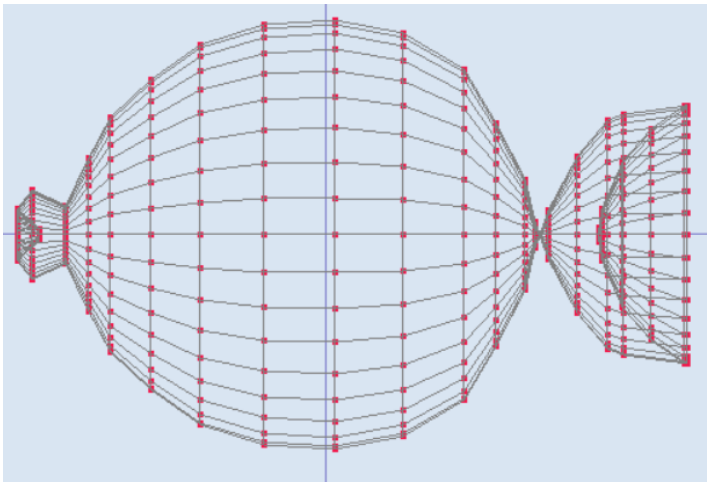
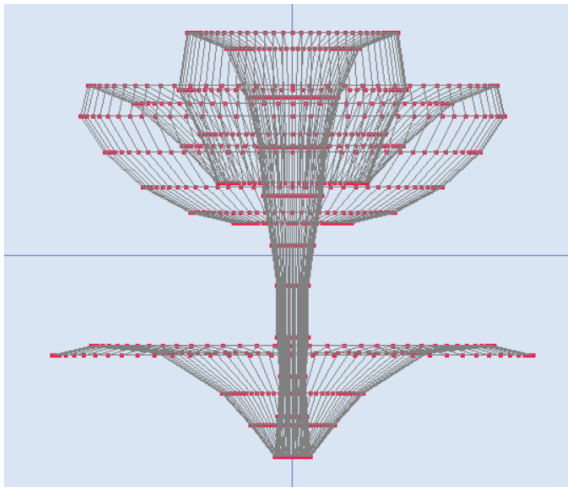
SOR(Surface Of Revolution) 모델링

컴퓨터그래픽스02분반_20203180_송정후_중간과제

Ⅰ. 프로젝트 설명

OpenGL을 이용하여 클릭한 곳에 점을 찍고, x축, y축을 기준으로 5, 10, 30, 50, 60, 90, 120도씩 회전하며 새로운 점과 wire frame을 생성하는 프로젝트를 구현해보았다.





-실행 순서

①click point : 마우스 왼쪽 버튼으로 클릭한 곳에 점을 찍고, 찍은 점들을 저장한다.

②menu : 마우스 오른쪽 버튼을 누르면 메뉴가 실행된다. (main menu) X_rotation, Y_rotation을 먼저 선택하고 (sub menu) 각도를 선택한다.

③new point : 선택한 축을 기준으로 선택한 각도만큼 회전하며 점을 찍고, wire frame을 그리고, 찍은 점들을 저장한다.

Ⅱ. 코드 설명

```
GLsizei winWidth = 1000, winHeight = 600;

class point3d
{
    public:
    float x, y, z;
};
std::vector<point3d> clickpt; //클릭한 점 저장
std::vector<point3d> newpt; //회전하며 도는 점 저장

void init(void)
{
    glClearColor(0.85, 0.9, 0.95, 1); //하늘색
    glMatrixMode(GL_PROJECTION);
}

void winReshapeFcn(int newWidth, int newHeight)
{
    glViewport(0, 0, newWidth, newHeight);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(0.0, GLdouble(newWidth), 0.0, GLdouble(newHeight));
    winWidth = newWidth;
    winHeight = newHeight;
}
```

1. 기본 설정

①window 설정 : winWidth와 winHeight를 각각 100, 600으로 설정했고, 배경 색은 하늘색(0.85, 0.9, 0.95)로 설정하였다.

②point3d : class를 생성하여 점을 3d좌표로 쉽게 접근할 수 있게 하였다.

③vector : clickpt는 클릭한 점들을 저장하고, newpt는 회전하며 도는 점들을 저장한다.

④winReshapeFcn : viewport를 window의 크기에 맞게 설정하여 왜곡 문제를 해결하였다.

```
void displayLine(void) //x축, y축 그리기
{
    glColor3f(0.5, 0.5, 0.8); //파란색
    glBegin(GL_LINES);
    glVertex2i(500, 0); glVertex2i(500, 600);
    glVertex2i(0, 300); glVertex2i(1000, 300); glEnd();
    glFlush();
}
```

⑤ displayLine : x축과 y축을 보여주는 함수이다.

2. 점 찍는 함수, 선 그리기

```
void plotPoint(GLint x, GLint y, GLint z) //점 찍는 함수
{
    glPointSize(4);
    glColor3f(0.95, 0.1, 0.3); //핑크색
    glBegin(GL_POINTS);
    glVertex3i(x, y, z); glEnd();
}
```

```
void drawLine(point3d a, point3d b) //선 그리는 함수
{
    glBegin(GL_LINES);
    glColor3f(0.5, 0.5, 0.5); //회색
    glVertex3i(a.x, a.y, a.z);
    glVertex3i(b.x, b.y, b.z);
    glEnd();
}
```

①plotPoint : 좌표(3d)를 입력 받아 핑크색 점을 찍는 함수이다.

②drawLine : 두 점을 point3d값으로 입력 받아 회색 선을 그리는 함수이다. wire frame에 이용하였다.

3. click point

```
void clickPoint(GLint button, GLint action, GLint xMouse, GLint yMouse) //클릭한 곳에 점 찍고 clickpt에 저장
{
    GLint x = xMouse;
    GLint y = winHeight - yMouse;

    if (button == GLUT_LEFT_BUTTON && action == GLUT_DOWN) {
        plotPoint(x, y, 0);
        point3d pt;
        pt.x = x; pt.y = y; pt.z = 0;
        clickpt.push_back(pt);
    }
    glFlush();
}
```

마우스의 왼쪽 버튼을 클릭하면 클릭한 곳에 점을 찍고, 점을 clickpt에 저장하는 함수이다. window좌표계와 3차원 좌표계의 값을 맞춰 주기 위해 winHeight – yMouse를 해주었다. plotPoint를 이용해 점을 찍고, push _back을 이용하여 찍은 점들을 저장하였다.

```
GLboolean Xrotation = true;

void newPoint(GLint angle) //angle만큼 회전하면서 점 찍고 newpt에 저장
{
    int angleSize = 360 / angle - 1;
    double radian = angle * (3.141592653589793 / 180);
    point3d pt;
    for (int i = 0; i < clickpt.size(); i++) {
        for (int j = 1; j < angleSize + 1; j++) {
            if (Xrotation)
            {
                float radius = 300 - clickpt[i].y; //X_rotation
                pt.x = clickpt[i].x;
                pt.y = 300 - radius * cos(radian * j);
                pt.z = radius * sin(radian * j);
            }
            else
            {
                float radius = 500 - clickpt[i].x; //Y_rotation
                pt.x = 500 - radius * cos(radian * j);
                pt.y = clickpt[i].y;
                pt.z = radius * sin(radian * j);
            }
            plotPoint(pt.x, pt.y, pt.z);
            newpt.push_back(pt);
        }
    }
    wireFrame(angleSize);
    glFlush();
}
```

4. new point

각도를 입력 받아 각도만큼 회전하면서 점을 찍고, wire frame을 그리고, 찍은 점들을 newpt에 저장하는 함수이다.

①Xrotation : boolean값으로 선언하여 x축과 y축을 구별할 때 이용하였다.

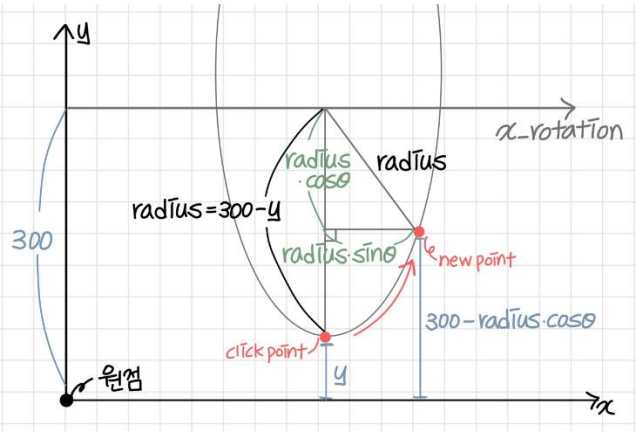
②angleSize : click point하나 당 찍어야 할 new point의 개수이다. 360에서 angle을 나눈 몫에 1을 빼서 구해주었다.

③radian : angle을 degree값으로 입력 받기 때문에 'angle * 원주율 / 180'를 통해 radian값으로 바꿔주었다.

④for : clickpt에 저장한 점들을 순서대로 불러왔고(clickpt[i]), radian값을 배로 증가시키며 점을 찍어주었다(radian * j).

1)X_rotation

x축을 기준으로 angle만큼 회전했을 때의 좌표를 구해보자. 그림에서 click point의 좌표는 (x, y, z)이다.



radius : 점이 회전하며 그리는 원의 반지름이다. winHeight가 600이므로 radius는 300에서 y를 뺀 값이다.

new point의 좌표

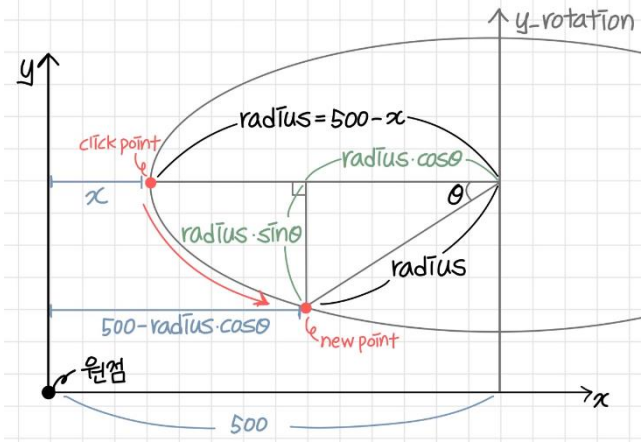
X : x

Y : 300 - radius * cos(radian)

Z : radius * sin(radian)

2)Y_rotation

y축을 기준으로 angle만큼 회전했을 때의 좌표를 구해보자.



radius: winWeight가 1000이므로 radius는 500에서 x를 뺀 값이다.

new point의 좌표

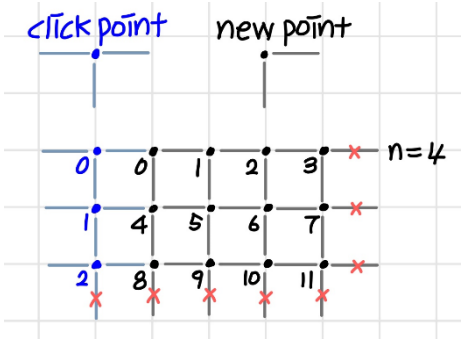
X : 500 – radius * cos(radian)

Y : y

Z : radius * sin(radian)

5. wire frame

```
void wireFrame(int n) //wire frame 그리기
{
    for (int i = 0; i < clickpt.size(); i++) {
        drawLine(clickpt[i], newpt[n * i]);
        drawLine(clickpt[i], newpt[(i + 1) * n - 1]);
        if (i != clickpt.size() - 1) {
            drawLine(clickpt[i], clickpt[i + 1]);
        }
        for (int j = 0; j < newpt.size(); j++) {
            if (j % n != n - 1) {
                drawLine(newpt[j], newpt[j + 1]);
            }
            if (j < newpt.size() - n) {
                drawLine(newpt[j], newpt[j + n]);
            }
        }
    }
}
```



입력 받는 n은 angleSize이다. 위의 그림을 보면 click point(파랑) 하나당 네 개의 new point(검정)가 찍히므로 n은 4이다. clickpt[0]를 보면 clickpt[1], newpt[0], newpt[3]과 연결되며 총 3개의 선을 그린다. 첫번째 new point(newpt[0])은 newpt[0], newpt[4]와 연결되며 총 2개의 선을 그린다. 즉, for을 통해 clickpt를 돌면서 clickpt[i]가 각각 clickpt[i+1], newpt[n*i], newpt[(i+1)*n – 1]와 연결되고, newpt를 돌면서 newpt[j]가 newpt[j + 1], newpt[j+n]과 연결시켜주었다. 하지만 오른쪽 그림에서 X친 부분들은 연결시킬 수 없으므로 조건을 설정해주었다.

6. Menu

```
GLint XmenuID = glutCreateMenu(Xmenu); //sub menu: X_rotation
glutAddMenuEntry("5", 5);
glutAddMenuEntry("10", 10);
glutAddMenuEntry("30", 30);
glutAddMenuEntry("50", 50);
glutAddMenuEntry("60", 60);
glutAddMenuEntry("90", 90);
glutAddMenuEntry("120", 120);
GLint YmenuID = glutCreateMenu(Ymenu); //sub_menu: Y_rotation
glutAddMenuEntry("5", 5);
glutAddMenuEntry("10", 10);
glutAddMenuEntry("30", 30);
glutAddMenuEntry("50", 50);
glutAddMenuEntry("60", 60);
glutAddMenuEntry("90", 90);
glutAddMenuEntry("120", 120);
GLint MyMainMenuID = glutCreateMenu(Mmenu); //main menu
glutAddSubMenu("X_rotation", XmenuID);
glutAddSubMenu("Y_rotation", YmenuID);
glutAddMenuEntry("Exit", 0);
glutAttachMenu(GLUT_RIGHT_BUTTON);
```

```
void Mmenu(int entryID) //main menu
{
    if (entryID == 0) exit(0);
}

void Xmenu(int entryID) //sub menu: X_rotation
{
    Xrotation = true;
    newPoint(entryID);
    clickpt.clear(); newpt.clear(); glClear(GL_COLOR_BUFFER_BIT);
}

void Ymenu(int entryID) //sub menu: Y_rotation
{
    Xrotation = false;
    newPoint(entryID);
    clickpt.clear(); newpt.clear(); glClear(GL_COLOR_BUFFER_BIT);
}
```

Mmenu(Main Menu)에는 X_rotation, Y_rotation, Exit를 선택할 수 있게 하였다. X_rotation, Y_rotaiton은 각각 서브메뉴인 Xmenu, Ymenu로 연결된다. Exit는 프로그램을 종료한다. Xmenu, Ymenu 둘다 5, 10, 30, 50, 60, 90, 120을 선택할 수 있고 선택한 숫자의 각도만큼 회전하며 점을 찍는다. entryID를 각도와 동일하게 설정하여 각도에 쉽게 접근할 수 있게 하였다. 둘 다 newPoint(entryID)를 이용하여 점을 찍었다. clickpt.clear(), newpt.clear()을 이용하여 메뉴를 실행시키고 나면 저장된 점들을 지웠고, glClear을 통해 점들을 화면에서 지워 다시 새로운 점을 찍을 수 있게 하였다.

7. 콜백함수

```
glutInitWindowPosition(100, 100);
glutInitWindowSize(winWidth, winHeight);
glutCreateWindow("20203180_SongJunghu");
glutReshapeFunc(winReshapeFcn);
glClearColor(0.85, 0.9, 0.95, 1);
glClear(GL_COLOR_BUFFER_BIT);
glOrtho(-1, 1, -1, 1, 1000, -1000);
```

```
glutDisplayFunc(displayLine);
glutMouseFunc(clickPoint);
glutMainLoop();
```

콜백함수들을 이용하여 windowPosition, windowSize, window의 이름을 설정해 주었다.

Ⅲ. 결론

Window에 x축, y축을 그리고, 점을 찍고 메뉴를 선택하면 선택한 축을 선택한 각도만큼 회전하며 점과 wire frame을 그리는 프로그램을 구현해보았다. Window에 그려지는 x축, y축을 기준으로 한 좌표와, 점들의 실제 좌표가 달라 new point의 좌표를 설정해주는 것이 힘들었다. 이 과정을 통해 window좌표계에 대해 더 잘 이해할 수 있었다. Wire frame을 그릴 때 생각해야 할 변수들이 많아 힘들었지만 wire frame을 통해 물체를 더욱 입체적으로 보이게 할 수 있었다. 3d로 구현된 점, 선들을 원근감 없이 평면에 직교 투영했는데도, 일정한 각도로 회전하며 찍힌 점들과 wire frame만으로 입체적으로 보일 수 있다는 것이 신기했다.