

컴퓨터 그래픽스 과제 HW2

학과: 컴퓨터공학부

학번: 201801569

이름: 송혜민

1.

(1) 주어진 세 점에 의해 만들어지는 평면에 수직인 벡터를 구하는 방법은 벡터 P_1P_2 와 P_1P_3 의 외적을 구하면 된다.

구하는 계산과정은 다음과 같다.

$$P_1 = (1, 1, 1), P_2 = (1, 2, 1), P_3 = (3, 0, 4)$$
$$\vec{P_1P_2} = (0, 1, 0), \vec{P_1P_3} = (2, -1, 3)$$
$$\vec{n} = \vec{P_1P_2} \times \vec{P_1P_3} = \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 3 \\ -1 & 3 & -1 \end{vmatrix} = (3, 0, -2)$$

따라서 Normal Vector은 $(3, 0, -2)$ 이다.

(2) 위 (1)번의 계산과정을 glm라이브러리를 이용하여 작성하기 위해선 cross함수를 이용하면 된다.

이용하여 작성한 결과는 다음과 같이 나온다.

```
Microsoft Visual Studio 디버그 콘솔
cross( p1p2, p1p3 )
3 0 -2
C:\Users\송혜민\source\repos\ConsoleApplication2\Debug\ConsoleApplication2.exe(프로세스 8716개)이(가) 종료되었습니다
(코드: 0개).
이 창을 닫으려면 아무 키나 누르세요...
```

(3) 세 점이 만약 같은 직선 위에 존재한다면, 그 직선을 포함한 평면은 무수히 많이 존재할 것이다.

따라서 평면이 무수히 많으므로 법선 벡터 또한 무수히 많을 것이다.

2.

(1) (1, 2, 0)을 반시계 방향으로 45도 회전시켰을 때의 변환행렬은 다음과 같다.

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos 45^\circ & 0 & \sin 45^\circ & 0 \\ 0 & 1 & 0 & 0 \\ -\sin 45^\circ & 0 & \cos 45^\circ & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 0 \\ 1 \end{bmatrix}$$
$$= \begin{bmatrix} \frac{\sqrt{2}}{2} \\ 2 \\ -\frac{\sqrt{2}}{2} \\ 1 \end{bmatrix}$$

(2) glm 라이브러리를 이용하여 결과와 동일한지 알아보려 한다.

수업시간에 이용한 예제를 활용하여 glm::rotate의 두 번째 인자는 45도로 수정하였고 glm::vec3에서 (0,1,0)을 값으로 주어 y축으로 회전하게 설정하였다.

또한 점(1,2,0)을 회전시키는 것이므로 점의 값도 수정하였다.

출력결과를 위에서 구한 값과 동일하다.

```
Microsoft Visual Studio 디버그 콘솔
0.707107, 2.000000, -0.707107
C:\Users\송혜민\source\repos\ConsoleApplication2\Debug\ConsoleApplication2.exe(프로세스 10716개)이(가) 종료되었습니다(코드: 0개).
이 창을 닫으려면 아무 키나 누르세요...
```

3.

(1) 우선 45도로 먼저 회전을 했을 때의 좌표를 출력하고 그 다음 translate, scaling을 진행하였다. 행렬 곱은 교환법칙이 성립 안하기에 곱하는 순서에 유의하여 코드를 적고 실행한 결과 다음과 같이 나왔다.

```
C:\Users\송혜민\source\repos\ConsoleApplication2\Debug\ConsoleApplication2.exe
2.828427, 2.000000, 1.414213
0.828427, 5.000000, 2.414213
1.242641, -10.000000, 2.414213
계속하려면 아무 키나 누르십시오 . . .
```

(2) 행렬 곱은 교환법칙이 성립 안하기에 순서에 맞춰서 계산을 진행하면 다음과 같다.

손으로 계산한 결과 코드에서 얻어낸 값과 동일함을 알 수 있다.

① CCW

$$\begin{bmatrix} \frac{\sqrt{2}}{2} & 0 & \frac{\sqrt{2}}{2} & 0 \\ 0 & 1 & 0 & 0 \\ -\frac{\sqrt{2}}{2} & 0 & \frac{\sqrt{2}}{2} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{\sqrt{2}}{2} + \frac{3\sqrt{2}}{2} \\ 2 \\ -\frac{\sqrt{2}}{2} + \frac{3\sqrt{2}}{2} \\ 1 \end{bmatrix} = \begin{bmatrix} 2\sqrt{2} \\ 2 \\ \sqrt{2} \\ 1 \end{bmatrix} \quad \text{LP } P_1$$

② translate

$$P_1 + T_2 = \begin{bmatrix} 2\sqrt{2} - 2 \\ 2 + 3 \\ \sqrt{2} + 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 2\sqrt{2} - 2 \\ 5 \\ \sqrt{2} + 1 \\ 1 \end{bmatrix} \quad \text{P}_2$$

③ Scaling

$$P_3 = \begin{bmatrix} 1.5 & 0 & 0 & 0 \\ 0 & -2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2(\sqrt{2}-1) \\ 5 \\ \sqrt{2}+1 \\ 1 \end{bmatrix} = \begin{bmatrix} 3(\sqrt{2}-1) \\ -10 \\ \sqrt{2}+1 \\ 1 \end{bmatrix}$$

4.

(1) frustum의 좌표 값 8개는 $(+-(far/near)left, +-(far/near)top, -far)$, $(+left, +top, -near)$ 의 값이다
그 값을 구하면 다음과 같다.

```
void drawScene(void)
{
    glClear(GL_COLOR_BUFFER_BIT);
    glColor3f(0.0, 0.0, 0.0);

    float t = tan(30 * pi / 180);

    glBegin(GL_POLYGON);
    glVertex3f(-20 * t, 10 * t, -10); //
    glVertex3f(-20 * t, -10 * t, -10);
    glVertex3f(20 * t, 10 * t, -10);
    glVertex3f(20 * t, -10 * t, -10);
    glEnd();

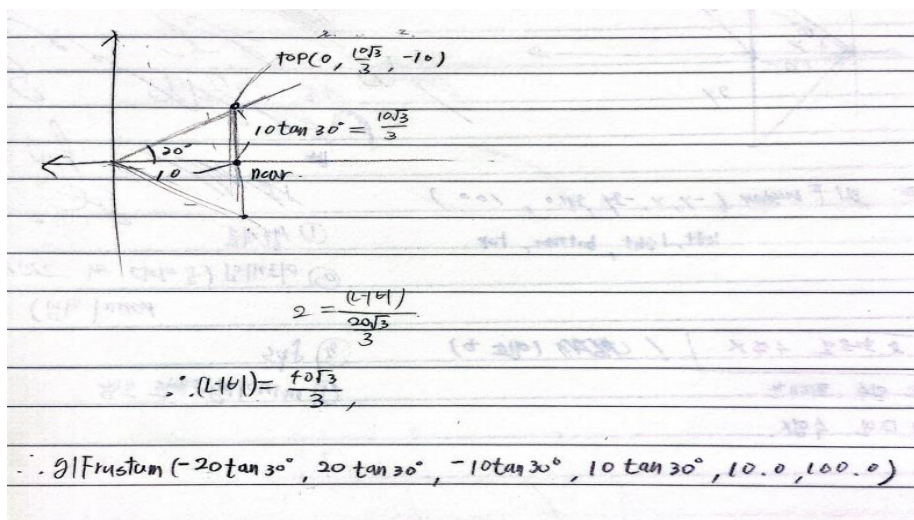
    glBegin(GL_POLYGON);
    glVertex3f(-200 * t, 100 * t, -100);
    glVertex3f(-200 * t, -100 * t, -100);
    glVertex3f(200 * t, 100 * t, -100);
    glVertex3f(200 * t, -100 * t, -100);
    glEnd();
    glFlush();
}
```

(2) 우선 fovy가 60이므로 apex에서 -z축 방향으로 위쪽 끝을 봤을 때 30도, 아래쪽 끝을 봤을 때 30도이다.

apex에서 viewing face까지의 z축으로의 거리, 즉 near가 5이고 viewing face의 y축에서 위쪽 끝이 10이기 때문에 삼각함수 공식을 이용하면 $10 \tan 30^\circ$ 의 값이 y축에서 위쪽 끝의 거리이다.

이를 두 배 한 값으로 너비를 나눌 경우 2(aspect)가 나와야 하므로 $40 \tan 30^\circ$ 의 값이 너비의 값이고 이것의 절반이 right와 left의 값이 된다.

이를 그림으로 정리하고 전체적인 값을 구하면 아래 사진과 같다.



5.

문제에서 카메라의 위치는 (4,4,4)이고 카메라가 바라보는 위치는 (0,1,0)이라 하였다.

따라서 +z축방향 벡터는 카메라의 위치 - 카메라가 바라보는 방향이므로 (4,3,4)가 나오게된다.

다음으로 +x축 방향 벡터는 up벡터와 +z축 벡터의 외적이다. 따라서 그 값은 (4,0,-4)가 된다.

마지막으로 +y축 방향 벡터는 +z축 벡터와 +x축 벡터의 외적이다. 값을 계산하면 (-12,32,-12)가 나온다.

이를 코드로 작성하여 계산한 값을 출력한 결과는 다음과 같이 나왔다.

 C:\Users\송혜민\source\repos\ConsoleApplication2\x64\Debug\ConsoleApplication2.exe

```
4.000000,3.000000,4.000000
4.000000,0.000000,-4.000000
-12.000000,32.000000,-12.000000
계속하려면 아무 키나 누르십시오 . . .
```