단국대학교 멀티미디어공학 그래픽스 프로그래밍 중간고사 (2010년 봄학기) 2010년 4월 20일

학과 _____ 학번 _____ 이름 _____

중간고사

담당교수: 단국대학교 멀티미디어공학전공 박경신

- **답은 반드시 답안지에 기술할 것. 공간이 부족할 경우 반드시 답안지 몇 쪽의 뒤에 있다고 명기한 후 기술할 것. 그 외의 경우의 답안지 뒤쪽이나 연습지에 기술한 내용은 답안으로 인정 안 함. 답에는 반드시 네모를 쳐서 확실히 표시할 것.**
- **답안지에 학과, 학번, 이름 외에 본인의 암호를 기입하면 성적공고시 학번 대신 암호를 사용할 것임.**

1. 맞으면 true, 틀리면 false를 적으시오. (10점)

1) OpenGL은 오른손 좌표계(right-handed coordinates)를 사용하므로, 다각형 그리기는 시계방향(clock-wise winding order)를 사용한다. ___F___

2) 벡터 그래픽 시스템은 무한 화소를 가지고 있으며 Aliasing이 없다. ___T___

3) 두 벡터의 곱이 0 이면 두 벡터는 직교한다. ___T___

4) 이동(translation)과 크기(scaling) 변환의 곱은 아핀 강체 변환(affine rigid-body Transformation)이다. ___F___

5) 이동(translation)과 회전(rotate) 변환의 곱은 아핀 강체 변환(affine rigid-body Transformation)이다. ___T___

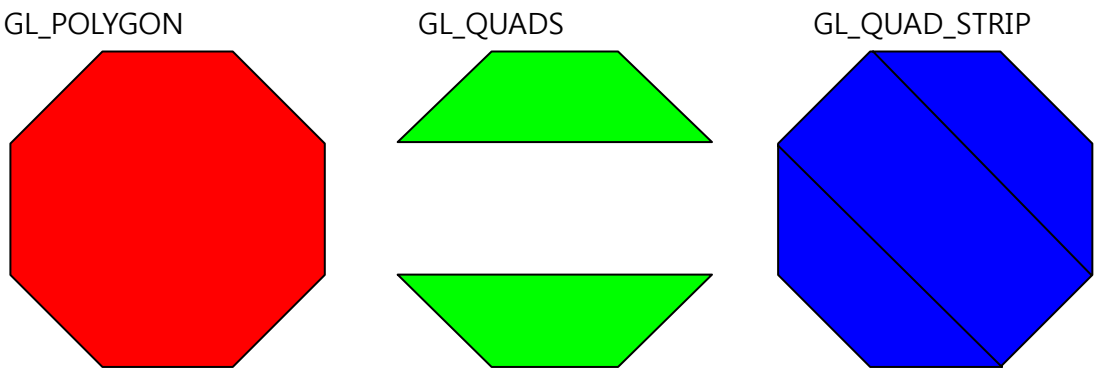
2. 다음에 주어진 간단한 OpenGL/GLUT Geometry Primitives 예제 코드를 보고 아래 문제에 답하시오. (30점)

```
#include <GL/glut.h>
#include <GL/gl.h>

int g_mode = 1;
float octagon_points[8][2] = {
        7.0,   3.5,
        3.5,   7.0,
       -3.5,   7.0,
       -7.0,   3.5,
       -7.0,   -3.5,
       -3.5,   -7.0,
        3.5,   -7.0,
        7.0,   -3.5
};

void display()
{
      glClearColor(1.0, 1.0, 1.0, 0.0);       // white background
      glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

      if (g_mode == 1) {
                glColor3f(1.0, 0.0, 0.0);     // red
                glBegin(GL_POLYGON);
                for (int i=0; i<8; i++)
                          glVertex2fv(octagon_points[i]);
```

```
                glEnd();
        }
        else if (g_mode == 2) {
                glColor3f(0.0, 1.0, 0.0);      // green
                glBegin(GL_QUADS);
                for (int i=0; i<8; i++)
                        glVertex2fv(octagon_points[i]);
                glEnd();
        }
        else if (g_mode == 3) {
                glColor3f(0.0, 0.0, 1.0);      // blue
                glBegin(GL_QUAD_STRIP);
                        glVertex2fv(octagon_points[0]);
                        glVertex2fv(octagon_points[1]);
                        glVertex2fv(octagon_points[7]);
                        glVertex2fv(octagon_points[2]);
                        glVertex2fv(octagon_points[6]);
                        glVertex2fv(octagon_points[3]);
                        glVertex2fv(octagon_points[5]);
                        glVertex2fv(octagon_points[4]);
                glEnd();
        }
        glutSwapBuffers();
}

void keyboard(unsigned char key, int x, int y)
{
        switch (key)
        {
        case 0x1B:
        case 'q':
        case 'Q':
                exit(0);
                break;
        case ' ':
                g_mode++;
                if (g_mode == 3) g_mode = 1;
                break;
        }
        glutPostRedisplay();
}

void reshape(int w, int h)
{
    glViewport (0, 0, (GLsizei) w, (GLsizei) h);
    glMatrixMode (GL_PROJECTION);
    glLoadIdentity ();
    gluPerspective(60.0, (GLfloat) w/(GLfloat) h, 1.0, 100.0);
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    gluLookAt (0.0, 0.0, 15.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);
}

int main(int argc, char** argv)
{
        glutInit(&argc, argv);
        glutInitDisplayMode(GLUT_RGB | GLUT_DOUBLE | GLUT_DEPTH);
```

```
    glutInitWindowSize(400, 400);
    glutInitWindowPosition(0,0);
    glutCreateWindow(argv[0]);
    glutDisplayFunc(display);
    glutKeyboardFunc(keyboard);
    glutReshapeFunc(reshape);
    glEnable(GL_DEPTH_TEST);
    glutMainLoop();
}
```

1) 위의 OpenGL/GLUT 코드에서 void display() 함수에서 기하학적 객체의 도형 모드 (glPolygonMode)를 GL_POLYGON, GL_QUADS, GL_QUAD_STRIP에 따른 도형을 그림으로 나타내어라. (5점)



2) 더블 버퍼링이 무엇인지 간단히 서술하시오. 그리고 위의 OpenGL/GLUT 코드에서 더블 버퍼링과 관련된 함수를 모두 찾아서 적으시오. (5점)

색 버퍼를 전면과 후면버퍼로 나누어서 사용함.
비디오 제어기가 항상 완성된 이미지를 그리도록 하기 위하여, 프로세서는 이미지의 내용을 계산하여 후면 버퍼에 축적하고, 그동안 비디어 제어기는 전면 버퍼의 내용을 읽어 화면에 이미지 도시함.
싱글 버퍼링보다 훨씬 부드러운 애니메이션 생성함.

```
glutInitDisplayMode(GLUT_DOUBLE| ....);
glutSwapBuffers();
```

3) Z-버퍼링이 무엇인지 간단히 서술하시오. 그리고 위의 OpenGL/GLUT 코드에서 Z-버퍼링과 관련된 함수를 모두 찾아서 적으시오. (5점)

z-버퍼는 일명 깊이버퍼라 불리며, 색 버퍼와 동일한 크기를 가지며, 픽셀 단위로 기하 개체 (geometry)의 깊이 값 (즉, Z-값)이 가장 작은 것이 앞에 그리도록 깊이 정보를 축적하는 버퍼이다.
OpenGL/GLUT에서 Z-버퍼링을 사용하기 위하여 깊이버퍼를 사용한다는 초기화가 필요하며 깊이정보 테스트를 활성화해야 하며, 디스플레이 함수 내에서 매 프레임마다 색 버퍼를 지울 때 깊이버퍼도 같이 지워야 한다.

```
glutInitDisplayMode(GLUT_DEPTH| ....);
glClear(GL_DEPTH_BUFFER_BIT | ....);
glEnable(GL_DEPTH_TEST);
```

학과 _____ 학번 _____ 이름 _____

4) 위의 OpenGL/GLUT 코드에서 void keyboard(unsigned char key, int x, int y) 함수가 프로그램에서 언제 불리는 지와 이 프로그램에서 keyboard 함수는 어떤 기능을 하는지 간단히 서술하시오. 그리고 keyboard 함수 안에 glutPostRedisplay 함수의 기능을 간단히 서술하시오 (5점)

keyboard 함수는 ASCII-character 키보드 키를 눌려졌을 때 불리는 답신함수이다.
이 프로그램에서 keyboard 함수가 불리면, q-key나 Esc-key가 눌리면 프로그램이 종료되며, spacebar-key가 눌리면 g_mode를 하나씩 증가하며 전환한다.

glutPostRedisplay 함수는 바뀐 g_mode가 적용되어 다시 화면에 그림을 그리도록, display 답신함수가 불릴 수 있도록 그리기 이벤트를 생성한다.

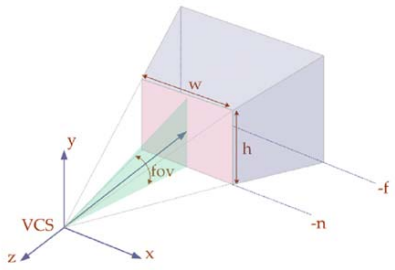
5) 위의 OpenGL/GLUT 코드에서 void reshape(int w, int h) 함수가 프로그램에서 언제 불리는 지와 이 프로그램에서 reshape 함수는 어떤 기능을 하는지 간단히 서술하시오. (5점)

reshape 함수는 프로그램 윈도우의 크기를 변형했을 때 불리는 답신함수이다.

이 프로그램에서 reshape 함수가 불리면, 변화된 윈도우 크기(w)와 높이(h)에 따라, glViewport 변환을 수행하고, gluPerspective 투영행렬 변환을 수행하고, gluLookAt 함수를 이용하여 카메라 변환 행렬을 수행한다.

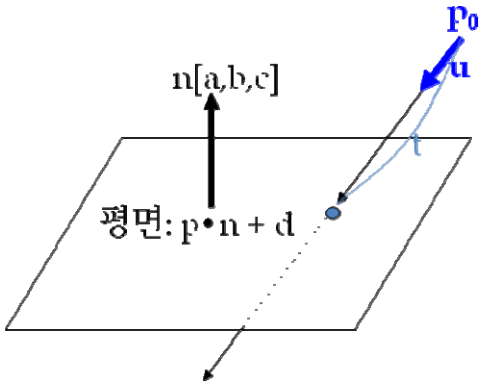
6) reshape 함수 안에 gluPerspective 함수가 생성해 내는 관측 공간을 그림으로 나타내어 서술하시오. 또한 OpenGL에서 gluPerspective와 같이 투영행렬을 생성해내는 관측함수를 2개 이상 적으시오. (5점)

gluOrtho2D(left, right, bottom, top);
gluPerspective(fov, aspect, znear, zfar);
glOrtho(left, right, bottom, top, znear, zfar);
glFrustum(left, right, bottom, top, znear, zfar);



학과 _____ 학번 _____ 이름 _____

3. 다음은 평면(Plane), 점(Point), Ray(광선)에 관한 질문이다. 빈 칸에 알맞은 답을 적으시오. (30점)



----------vector3.h & vector3.cpp

```
class vector3
{
public:
        // constructor..
        vector3(float x_, float y_, float z_);

        // static utility methods: vector3 v1(x1, y1, z1) & vector3 v2(x2, y2, z2)
        static float dotProduct(const vector3 & v1, const vector3 & v2);
        static vector3 crossProduct(const vector3 & v1, const vector3 & v2);
        //  중간생략...
protected:
        float x, y, z;
};

float vector3::dotProduct (const vector3 &v1, const vector3 &v2)
{
      return (v1.x * v2.x + v1.y * v2.y + v1.z * v2.z) ;      // 1)
}
vector3 vector3::crossProduct(const vector3 &v1, const vector3 &v2)
{
        vector3 result;
        result.x = v1.y * v2.z – v1.z * v2.y;
        result.y = v1.z * v2.x – v1.x * v2.z;                  // 2)
        result.z = v1.x * v2.y – v1.y * v2.x;                  // 3)
        return result;
}
```

----------plane.h & plane.cpp

```
class plane
{
public:
        // constructor..
        plane(float a_, float b_, float c_, float d_);

        // static utility methods: plane p(a, b, c, d) & vector3 v(x, y, z)
        static float dotNormal(const plane & p, const vector3 & v);// a*x + b*y + c*z + d*0
        static float dotCoord(const plane & p, const vector3 & v);  // a*x + b*y + c*z + d*1
        static vector3& closestPointOnPlane(const plane & p, const vector3 & v);
        //  중간생략...
protected:
        float a, b, c, d;
};
```

// 공간의 한 점(vector3 v)에서 평면(plane p)에 가장 가까운 점(vector3 out) 계산

```
vector3& plane::closestPointOnPlane (const plane &p, const vector3 &v)
{
      vector3 out;

    out[0] = v[0] - dotCoord(p, v) * a;   // 4)
    out[1] = v[1] - dotCoord(p, v) * b;   // 5)
    out[2] = v[2] - dotCoord(p, v) * c;   // 6)

    return out;
}
```

// 공간의 한 점(vector3 v)이 평면(plane p)에 있는지, 평면 밖에 있는지, 안에 있는지

```
int plane::isPointInsideOutside(const plane &p, const vector3 &v)
{
    float det = dotCoord(p, v);
    if (det == 0.0f)
        return ON_PLANE;              // on the plane
    else if (det > 0.0f)              // 7)
        return OUTSIDE_PLANE;         // outside
    else if (det < 0.0f)              // 8)
        return INSIDE_PLANE;          // inside
}
```

학과 _____ 학번 _____ 이름 _____

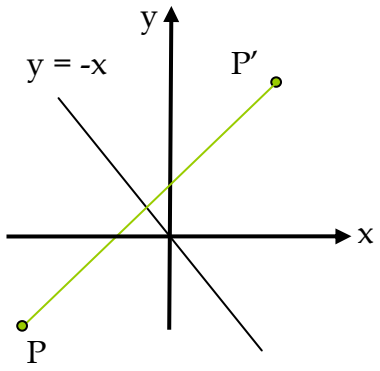
----------vector-matrix-plane-test.cpp

```
typedef struct _RAY {
        vector3 p0;       // 시작점
        vector3 u;        // 방향
} RAY;

// 광선(p(t) = p0 + tu)이 평면(plane p)과 만나는 점(vector3 out) 계산
bool RayPlaneIntersection(plane p, RAY line, vector3& out)
{
    float denom = plane::dotNormal(p, line.u);
    if (denom == 0)
        return false;

    float t = -plane::dotCoord(p, line.p0)/denom;      // 9)
    if (t < 0)
        return false;

    out = line.p0 + t*(line.u);                        // 10)

    return true;
}

int main(int argc, char *argv[])
{
        plane plane0(1, 0, 0, 1);
        vector3 Q(2, 1, -1), P(-1, 1, 0);
        int ret = plane::isPointInsideOutside(plane0, Q);
        cout << "ret = " << ret << endl;            // 11) ret = 1 (outside the plane)
        ret = plane::isPointInsideOutside(plane0, P);
        cout << "ret = " << ret << endl;            // 12) ret = 0 (on the plane)
        P = plane::closestPointOnPlane(plane0, Q);
        cout << "P = " << P << endl;                // 13) P = (-1, 1, -1)

        RAY ray1, ray2;
        ray1.p = vector3(1, 0, 0);
        ray1.u = vector3(-1, -1, 0);
        ray2.p = vector3(1, 0, -2);
        ray2.u = vector3(1, 1, 1);
```

```
    vector3 out1, out2;
    if (RayPlaneIntersection(plane0, ray1, out1))
        cout << "out1        = " << out1 << endl; // 14) __out1 = (-1, -2, 0)_____
    if (RayPlaneIntersection(plane0, ray2, out2))
        cout << "out2        = " << out2 << endl; // 15) ___nothing_____________

    return 0;
}
```

4. 다음 행렬에 대한 문제에 답하시오. (30점)

1) 아래 그림에서와 같이 2차원 공간의 점 P(x, y)를 직선 L (y = -x)에 대하여 반사(reflection)시켜 P′(x′, y′)으로 변환시키는 함수와 3x3 아핀변환 행렬을 나타내라. (5점)



x' = -y
y' = -x

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} 0 & -1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

2) 다음 3차원 아핀변환 행렬 M_1=T(5, 0, 0) R_z(90)를 계산하라. (cos90 = 0, sin90 = 1) (5점)

$$T = \begin{pmatrix} 1 & 0 & 0 & 5 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad R_z = \begin{pmatrix} \cos 90 & -\sin 90 & 0 & 0 \\ \sin 90 & \cos 90 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$M_1 = TR = \begin{pmatrix} 1 & 0 & 0 & 5 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos 90 & -\sin 90 & 0 & 0 \\ \sin 90 & \cos 90 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \cos 90 & -\sin 90 & 0 & 5 \\ \sin 90 & \cos 90 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

학과 _____ 학번 _____ 이름 _____

3) 다음 3차원 아핀변환 행렬 $M_2 = R_z(90)\,T(5, 0, 0)$를 계산하라. (5점)

$$M_2 = RT = \begin{pmatrix} \cos 90 & -\sin 90 & 0 & 0 \\ \sin 90 & \cos 90 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 5 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \cos 90 & -\sin 90 & 0 & 0 \\ \sin 90 & \cos 90 & 0 & 5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

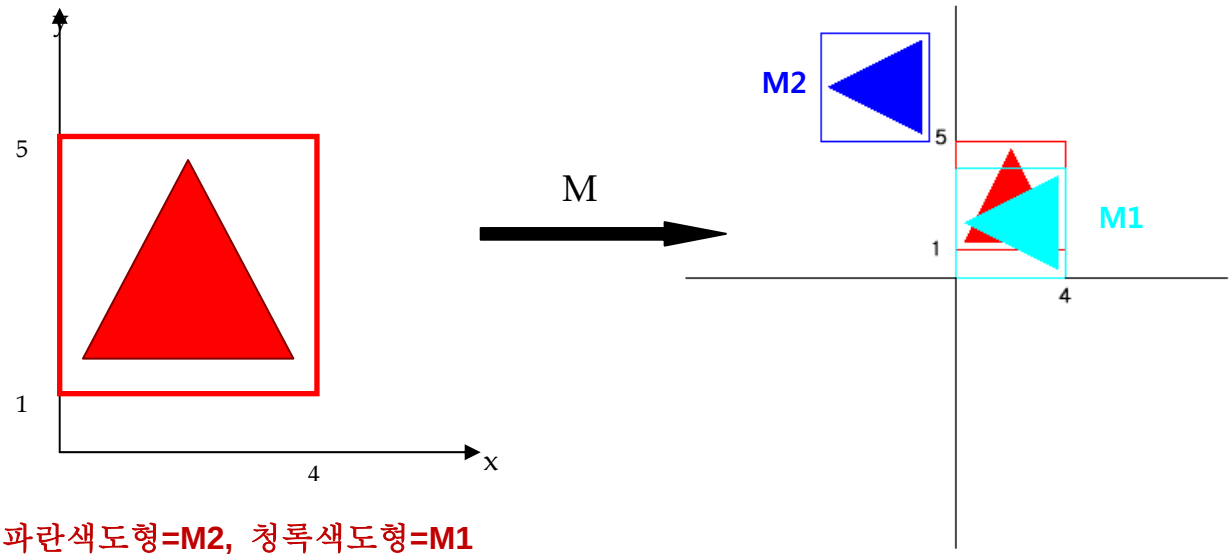
4) 3차원 아핀변환 행렬 $M_1=T(5, 0, 0)\ R_z(90)$을 기본적인 OpenGL 변환 함수(즉, glTranslatef, glRotatef, glScalef)로 표현하라. (5점)

```
glPushMatrix();
glTranslatef(5, 0, 0);
glRotatef(90, 0, 0, 1);
drawObject();
glPopMatrix();
```

5) 3차원 아핀변환 행렬 $M_2 = R_z(90)\ T(5, 0, 0)$를 기본적인 OpenGL 변환 함수(즉, glTranslatef, glRotatef, glScalef)로 표현하라. (5점)

```
glPushMatrix();
glRotatef(90, 0, 0, 1);
glTranslatef(5, 0, 0);
drawObject();
glPopMatrix();
```

6) 다음은 아래 왼쪽 기본 도형에, 위의 문제의 3차원 아핀변환 행렬 M_1과 M_2를 적용하여 나타난 모습을 오른쪽 그림에서 보여주고 있다. 오른쪽 그림의 각 도형에 적용된 아핀변환 행렬을 표시하라. 그리고 변환된 과정을 설명하라. (5점)

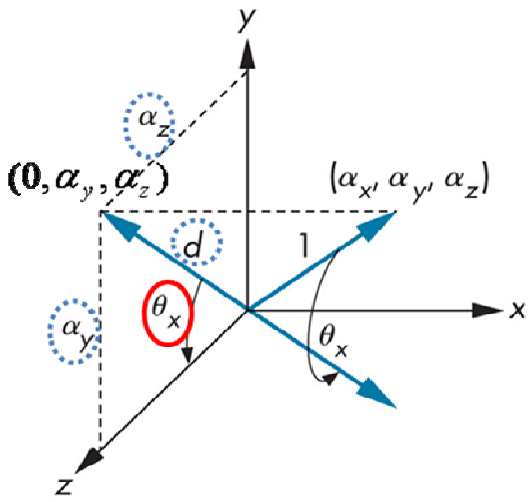


파란색도형=M2, 청록색도형=M1
M2는 x-축으로 5만큼 이동한 후, 90도 회전한 결과
M1은 90도 회전을 먼저 한 후, x-축으로 5만큼 이동한 결과

학과 _____ 학번 _____ 이름 _____

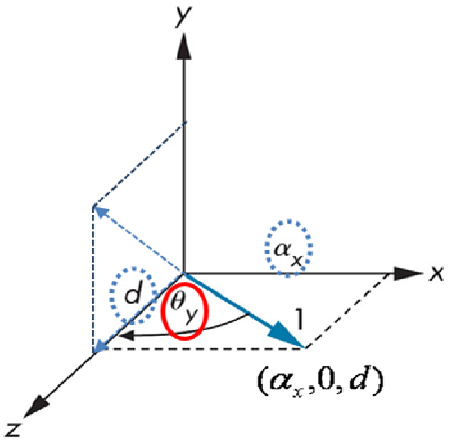
5. 다음은 임의의 축 (arbitrary axis) v=(αx, αy, αz)에 대한 3차원 회전 행렬을 유도하는 문제이다. 아래의 행렬에서 $R_x(\theta_x)$와 $R_y(\theta_y)$를 답하시오. (extra 10점)

$$M = T(P_0)R_x(-\theta_x)R_y(-\theta_y)R_z(\theta)R_y(\theta_y)R_x(\theta_x)T(-P_0)$$



$$R_x(\theta_x) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{\alpha_z}{d} & -\frac{\alpha_y}{d} & 0 \\ 0 & \frac{\alpha_y}{d} & \frac{\alpha_z}{d} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$d = \sqrt{\alpha_y^2 + \alpha_z^2}$$



$$R_y(\theta_y) = \begin{bmatrix} d & 0 & -\alpha_x & 0 \\ 0 & 1 & 0 & 0 \\ \alpha_x & 0 & d & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$d = \sqrt{\alpha_y^2 + \alpha_z^2}$$