Yazhuo Liu
Homework 3

1. (10 points) The projection of a vector A onto a vector B (parallel component) is given by

$$Proj_B\, A = (A\cdot B \,/\, |B|^2)\, B$$

and the component of A that is perpendicular to B is given by

$$Perp_B\, A = A - Proj_B\, A = A - (A\cdot B \,/\, |B|^2)\, B$$

Let A = (2, 2, 1) and B = (1, -2, 0), calculate $Proj_B\, A$ ***and*** $Perp_B\, A$

A.B = 2 – 4 + 0 = -2 $|B|^2 = 5$

$Proj_B$ A = (-0.4, 0.8, 0)

$Perp_B$ A = A – ProjB A = (2.4, 1.2, 1)

2. (10 points) Find the 4x4 transformation matrix corresponding to the OpenGL command:
glFrustum (-1.0, 1.0, -1.0, 1.0, 4, 8);

Then find the composite transformation matrix of a rotation about the z-axis by 30°. followed by the above glFrustum() matrix operation.

l = -1 r = 1 b = -1 t = 1 n = 4 f = 8

According to the formula:

$$R = \begin{matrix} 2 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & -3 & -16 \\ 0 & 0 & -1 & 0 \end{matrix}$$

$$R_z\,(30^o) = \begin{matrix} 0.866 & -0.5 & 0 & 0 \\ 0.5 & 0.866 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{matrix}$$

$$R_z\,.\,R = \begin{matrix} 1.732 & -1 & 0 & 0 \\ 1 & 1.732 & 0 & 0 \\ 0 & 0 & -3 & -16 \\ 0 & 0 & -1 & 0 \end{matrix}$$

3. (10 points) Find the 4x4 transformation matrix of the rotation about the axis passing through points (0, 0, 0) and (1, 1, 0) for 30°. (Hint: Decompose your transformation into a composite of elementary rotations.)
Write a simple OpenGL program to check if your calculation is correct.

$$Rz\,(45^o) = \begin{matrix} 0.707 & -0.707 & 0 & 0 \\ 0.707 & 0.707 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{matrix} \qquad Ry\,(-45^o) = \begin{matrix} -0.707 & 0 & -0.707 & 0 \\ 0 & 1 & 0 & 0 \\ 0.707 & 0 & -0.707 & 0 \\ 0 & 0 & 0 & 1 \end{matrix}$$

$$Rx\ (30^{o}) = \begin{matrix} 1 & 0 & 0 & 0 \\ 0 & 0.866 & -0.5 & 0 \\ 0 & 0.5 & 0.866 & 0 \\ 0 & 0 & 0 & 1 \end{matrix}$$

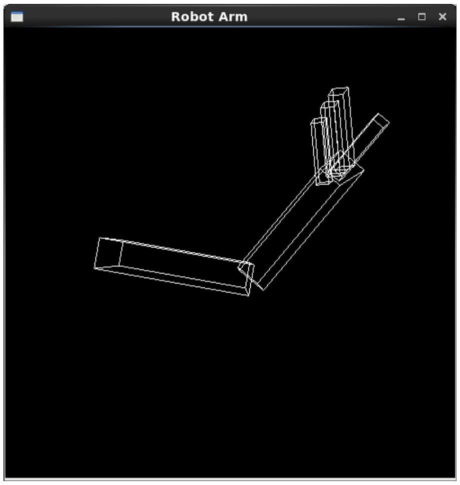
$$Rz\ .\ Ry\ .\ Rx = \begin{matrix} -0.5 & -0.862 & -0.08 & 0 \\ -0.5 & 0.362 & -0.787 & 0 \\ 0.707 & -0.354 & -0.612 & 0 \\ 0 & 0 & 0 & 1 \end{matrix}$$

4. (10 points) Suppose you want to render a scene where all objects are bounded by the box
$-1 \le x \le 2$, $6 \le y \le 8$, and $0 \le z \le 6$.

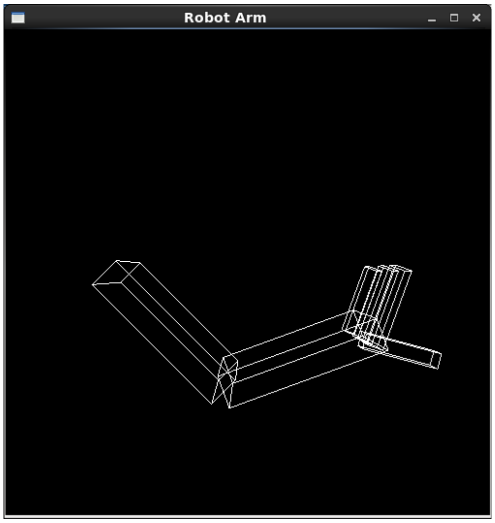
and the viewpoint ('camera') is at (8, 6, 9). Find the angle of the field-of-view for the viewpoint.

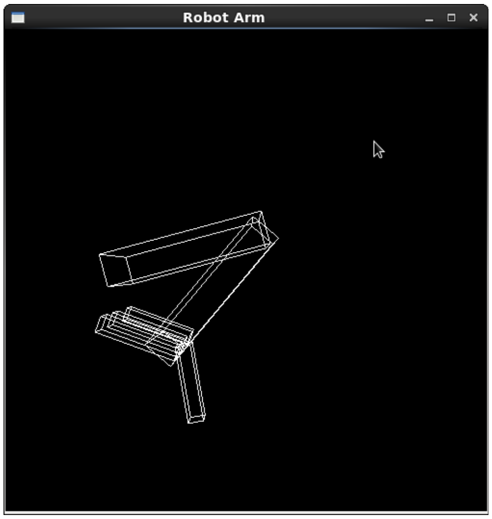
Center = (0.5, 7, 3) corner = (2, 8, 6)
r = sqrt (4.5 +1 + 9) = 3.5 d = sqrt (93.25) = 9.657
$\theta = 2 * \tan^{-1} (r/d) = 39.84^{o}$

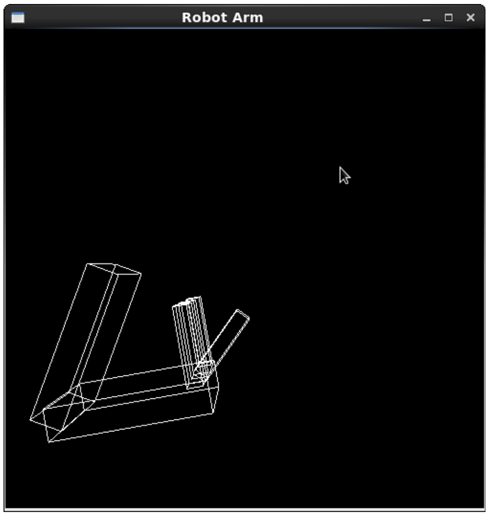
5. (20 points)
a) Write a program to draw the following robot arm.

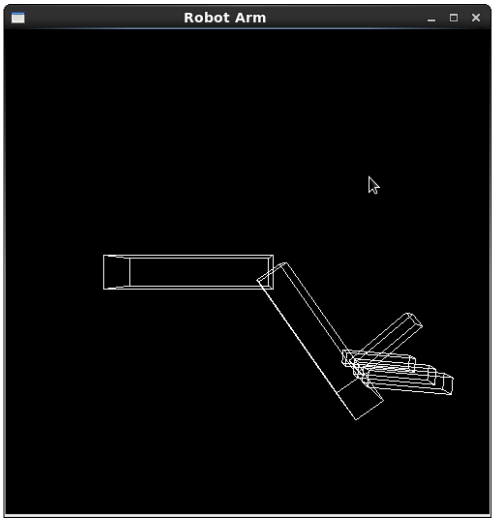


b) Add features to a) so that a user can control the movements of the robot arm by entering commands from the keyboard. The movements include rotate, swing, and translation of the arm and the finger gripping motions.









```
//robot.cpp
#include <GL/glut.h>
#include <stdlib.h>

static int shoulder = 0, elbow = 0, fingers = 0, thumb = 0;

void init(void)
{
  glClearColor (0.0, 0.0, 0.0, 0.0);
  glShadeModel (GL_FLAT);
}

void display(void)
{
  glClear (GL_COLOR_BUFFER_BIT);
  glPushMatrix();                              //Save M0

  glTranslatef (-1.7, 0.0, 0.0);               //M1 = T-1
  glRotatef ((GLfloat) shoulder, 0.0, 0.0, 1.0); //M2 = T-1Rs
  glTranslatef (1.0, 0.0, 0.0);          //M3 = T-1RsT+1
  glPushMatrix();                              //Save M3
  glScalef (2.0, 0.4, 1.0);                    //M4 = T-1RsT+1S
  glutWireCube (1.0);                          //P' =  T-1RsT+1S P
  glPopMatrix();                               //Restore M3 = T-1RsT+1

  glTranslatef (1.0, 0.0, 0.0);          //M5 = T-1RsT+1T+1
  glRotatef ((GLfloat) elbow, 0.0, 0.0, 1.0);  //M6 = T-1RsT+1T+1Re
  glTranslatef (1.0, 0.0, 0.0);          //M7 = T-1RsT+1T+1ReT+1
  glPushMatrix();                              //Save M7
  glScalef (2.0, 0.4, 1.0);                    //M8 = T-1RsT+1T+1ReT+1S
  glutWireCube (1.0);                          //P' = T-1RsT+1T+1ReT+1S P
  glPopMatrix();                               //Restore M7

  glTranslatef (0.7, 0.0, 0.0);
  glRotatef ((GLfloat) thumb, 0.0, 0.0, 1.0);
  glTranslatef (0.7, 0.0, 0.0);
  glPushMatrix();
  glScalef (1.0, 0.2, 0.2);
  glutWireCube (1.0);
  glPopMatrix();

  glTranslatef (-0.5, 0.0, 0.2);
  glRotatef ((GLfloat) fingers, 0.0, 0.0, 1.0);
  glTranslatef (-0.5, 0.0, 0.2);
  glPushMatrix();
  glScalef (1.0, 0.2, 0.2);
  glutWireCube (1.0);
  glPopMatrix();

  glTranslatef (0.0, 0.0, -0.2);
  glTranslatef (0.0, 0.0, -0.2);
  glPushMatrix();
  glScalef (1.0, 0.2, 0.2);
  glutWireCube (1.0);
  glPopMatrix();

  glTranslatef (0.0, 0.0, -0.3);
  glTranslatef (0.0, 0.0, -0.3);
  glPushMatrix();
  glScalef (1.0, 0.2, 0.2);
  glutWireCube (1.0);
  glPopMatrix();

  glPopMatrix();                              //Restore M0
  glutSwapBuffers();
}

void reshape (int w, int h)
{
  glViewport (0, 0, (GLsizei) w, (GLsizei) h);
  glMatrixMode (GL_PROJECTION);
  glLoadIdentity ();
  gluPerspective(65.0, (GLfloat) w/(GLfloat) h, 1.0, 20.0);
  glMatrixMode(GL_MODELVIEW);
  glLoadIdentity();
  glTranslatef (0.0, 0.0, -5.0);
}

void keyboard (unsigned char key, int x, int y)
{
  switch (key) {
    case 'w':
      shoulder = (shoulder + 5) % 360;
      glutPostRedisplay();
      break;
    case 's':
      shoulder = (shoulder - 5) % 360;
      glutPostRedisplay();
      break;
    case 'a':
      elbow = (elbow + 5) % 360;
      glutPostRedisplay();
      break;
    case 'd':
      elbow = (elbow - 5) % 360;
      glutPostRedisplay();
      break;
    case 'f':
      thumb = (thumb + 5) % 360;
      glutPostRedisplay();
      break;
    case 'e':
      fingers = (fingers - 5) % 360;
      glutPostRedisplay();
      break;
    case 27:
      exit(0);
      break;
    default:
      break;
  }
}

int main(int argc, char** argv)
{
  glutInit(&argc, argv);
  glutInitDisplayMode (GLUT_DOUBLE | GLUT_RGB);
  glutInitWindowSize (500, 500);
  glutInitWindowPosition (100, 100);
  glutCreateWindow ("Robot Arm");
  init ();
  glutDisplayFunc(display);
  glutReshapeFunc(reshape);
  glutKeyboardFunc(keyboard);
  glutMainLoop();
  return 0;
}
```

Report:

The first four problems are really straight forward, just plug variables in the formulas. The last one requires some work. I tried a lot of times but it is still not perfect. I successfully finished most of this homework. The last problem could still use some modification and improvement.