Robots, the GLUT and IK-CCD

This document discusses the robot simulator introduced in the chapter on OpenGL programming. The primary purpose here is to present details of the simulator model and how it relates to the real PA0 small industrial robot.

> The files asociated with this project are located in folder GL-RobotSimulator1. The files in folder GL-RobotSimulator2 relate to an extended version of the program that uses a *SensAble Phantom Haptic device* to set the orientation of the end effector of the robot.

A Mitsubishi Industries PA10 robot has a tool head at the end of a chain of rotational links with seven degree of freedom. The Base of the robot can rotate about a vertical axis and there are three other pairs of links, therefore, every configuration that the robot manipulator can adopt is specified by seven parameters.

Figure 1 shows a diagram of an actual robot with the seven rotational angles labeled: S1 S2 S3 E1 E2 W1 W2. A photograph of the real PA10 robot is presented on the right.

Figure 2 shows the functional diagram

Figure 3 shows the relationship between the real robot's axes that give it the seven degrees of freedom and the graphics model.

From operating point of view we think of the hand as an independently orientated work piece. Thus the function of the other links in the arm is to move the wrist to a position where the tool hand can work the tool, this is the point P in figure 3. To move P to the desired location (e.g. target T in figure 3) the articulated linkage has two links controlled by four pivots. The two links can be clearly seen in the functional diagram of figure 2. Each pivot is controlled by two angles occurring in pairs, with each rotation taking place abut a single axis perpendicular to a plane.

The lower link, (the upper arm) pivots about the base (the shoulder) using angles S1 and S2 to point the upper arm in any direction.

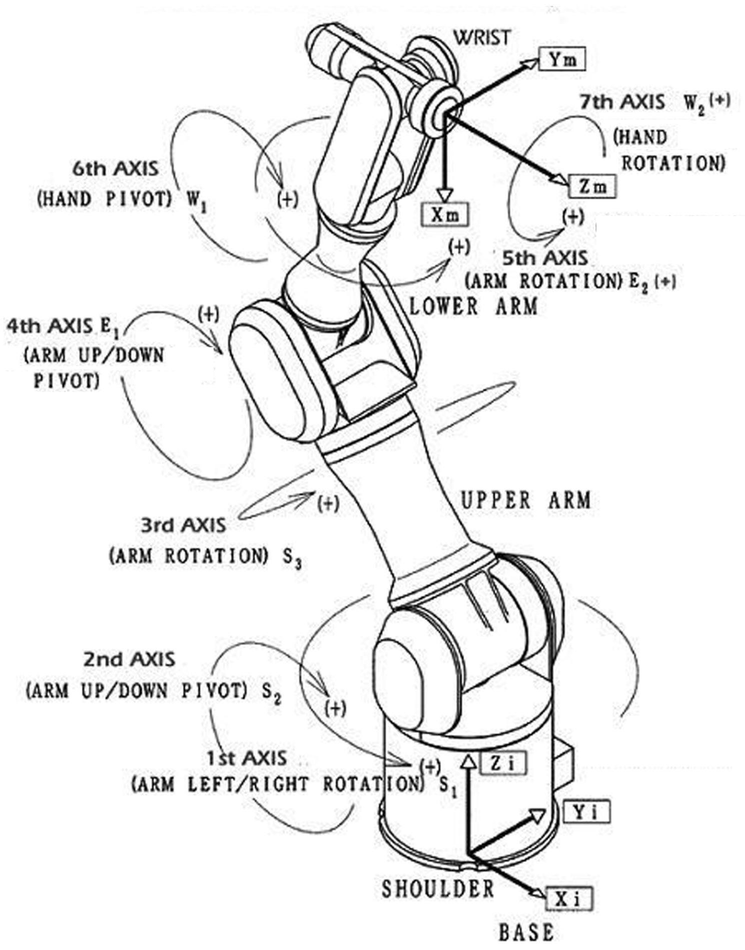
The upper link (the forearm) pivots about the elbow, using rotations S3 and E1.

Using these four rotations the wrist point P can be moved throughout a large volume. Once the wrist is positioned then rotations E2, W1 and W2 may be used to point the tool in any direction and with any orientation.

Before examining the code you need to be become familiar with the operation of the program.

- The left mouse button controls the viewing direction. Click the left button and drag the mouse to change the view of the model.

- The right mouse button controls the magnification, click the right mouse button and drag the mouse (left and right) to zoom-in and zoom-out.



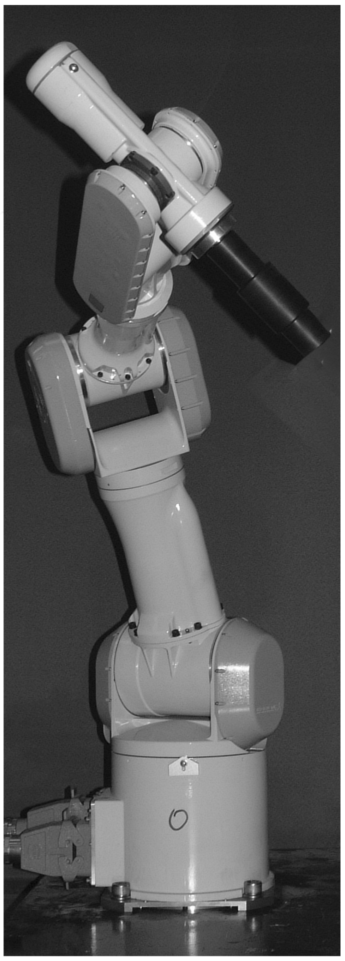


Figure 1: The PA10 Robot showing pivot axes and angles S1, S2, S3 E1, E2, W1, W2 giving it 7 degrees of freedom.

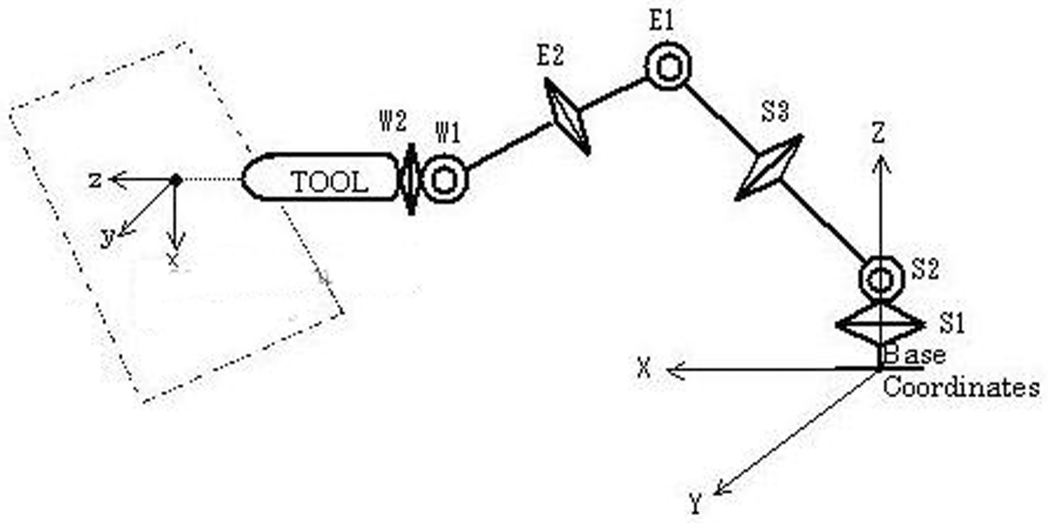


Figure 2: Functional diagram.

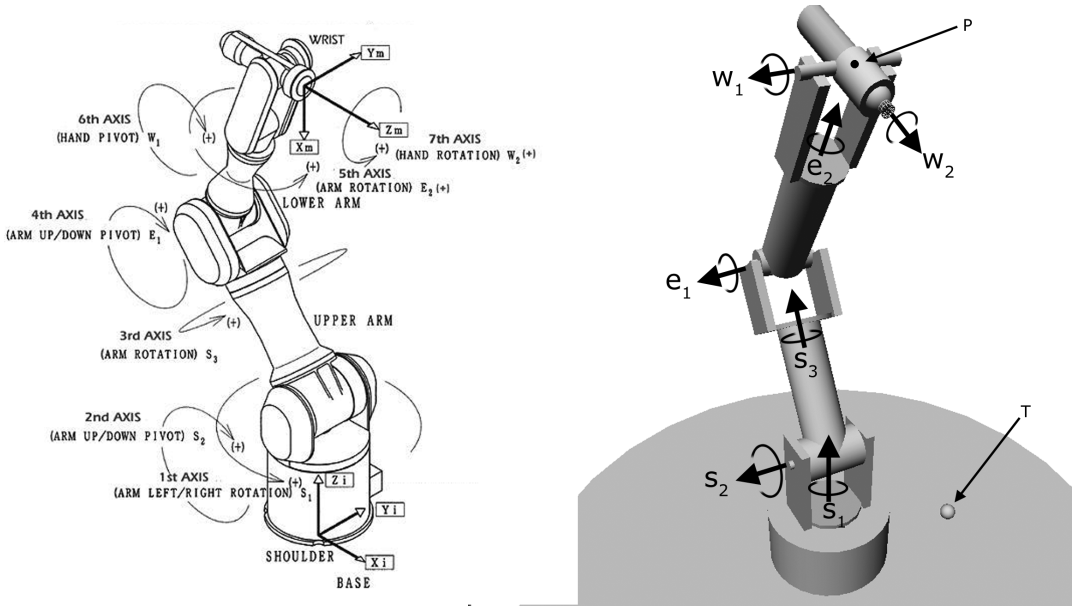

WRIST
Ym
6th AXIS (HAND PIVOT) W1
(+)
Zm
Xm
7th AXIS (HAND ROTATION) W2 (+)
(+)
5th AXIS (ARM ROTATION) E2 (+)
LOWER ARM
4th AXIS (ARM UP/DOWN PIVOT) E1
(+)
3rd AXIS (ARM ROTATION) S3
(+)
UPPER ARM
2nd AXIS (ARM UP/DOWN PIVOT) S2
(+)
1st AXIS (ARM LEFT/RIGHT ROTATION) S1
(+)
Zi
Yi
Xi
SHOULDER
BASE
P
w1
e2
w2
e1
s3
T
s2
s1

Figure 3: The PA10 robot and the OpenGL model built from simple GLUT primitives.

- The center mouse button (the third button) brings up a context menu allowing the user to change settings. The most important settings is *Whether to use Forward Kinematics or Inverse Kinematics to control the pose of the robot.*
- The other context menu commands change:
  1. The viewing axis, along X, Y and Z.
  2. Use the CCD IK algorithm to position the wrist at the target point.
  3. What is displayed:
     - The stylized model. Created from GLUT cylinders and cubes.
     - The axes of the frames of reference attached to each link. See figure 4 which shows the view when the axes are visible.
     - The axes of the frames of reference, as determined by the CCD IK algorithm.
  4. Reset to the rest pose
  5. Print information about the transformations of each links
- Toggle *interactive Target Tracking using IK* on or off:
  - When interactive IK is off (by default) the position of the robot is determined by setting the joint angles. The joint angles are set by the user using the keyboard keys"1" to" 0" and "-" "=" and "q" "w"
  - When the interactive IK algorithm is *on* the robot's wrist tracks the target. The position of the target is set using the Arrow and the Page-Up and Page-Down keys. (Use the ¡Ctrl¿ to modify the rate of target movement (slower.))

  Note that when in interactive mode the orientation of the hand (at the end of the articulated linkage) is maintained in the orientation that is it give by by the user. To set the orientation of the end-effector use the ¡shift¿ key modifier with the Arrow and Page keys.

The program's code is contained in one file and is broken up into functions that are grouped as follows:

- The Cyclic Coordinate Descent algorithm for IK

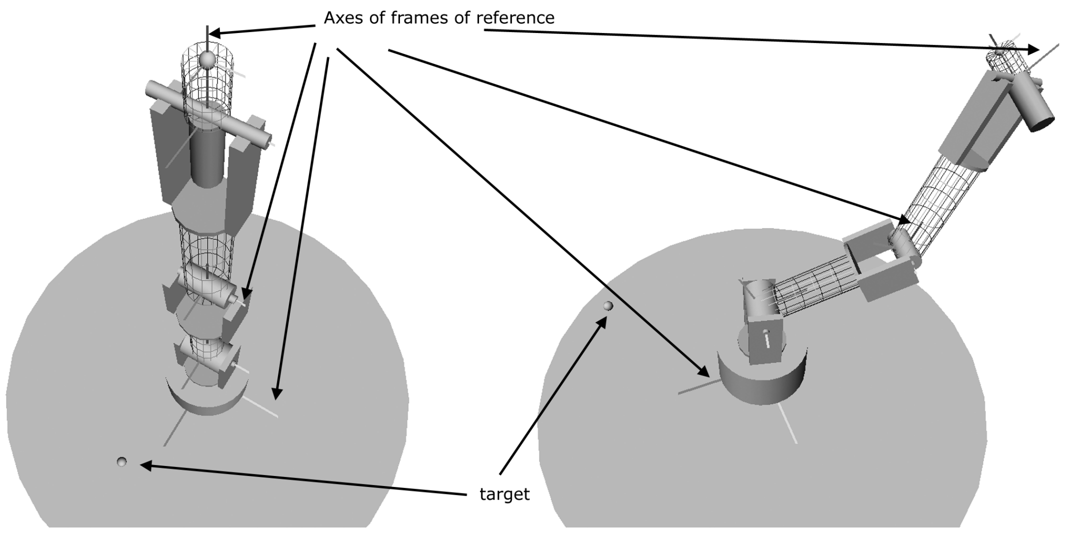


Figure 4: The Axis view shows the axes of the frames of reference attached to each link. By rotation about one of these axes the links in the robot can be positioned into its working pose. The figure also highlights the position of the IK target.

- Function the read the OpenGL transform stack. Used to compare with the program's own transformations.
- Drawing of robot, frames of reference axes and constituent parts.
- User interface: mouse and keyboard message handling.
- Geometry and linear algebra. matrix and vector operations.
- Main entry point to configure and initiate the program using the GLUT.

The most important global variables in the program are defined in listings 1,2,3,4,5,6 and 7.

```
// These angles control the angles of rotation away from the
// rest position for the Robot Arm's joints, they are as indicated
// in the accompanying document.
GLfloat rs1=0.0,rs2=0.0,rs3=0.0,
        re1=0.0,re2=0.0,
        rw1=0.0,rw2=0.0;
```

Listing 1: Global variables

```
// These angles are determined by the CCD algorithm to allow the
//robot to move the centre of its tool
// to the given target position.  (How to arrange that the
// arm moves its working tool so that it approached
// a working points - is a more complicated problem. Ways to
// solve this problem is stil an ongoing research topic.
GLfloat ccd_rs1=0.0,ccd_rs2=0.0,ccd_rs3=0.0,
        ccd_re1=0.0,ccd_re2=0.0,
        ccd_rw1=0.0,ccd_rw2=0.0;
```

Listing 2: Global variables

```
// IK target for the robot's wrist point - when using IK mode
GLfloat p_target[3]={42,0,50};
// These three angles are used to give the direction that the end-effector (the tool)
// at the end of the Robot Arm should be pointing in. (When the CCD algorithm is in use.)
GLfloat hand_pan=0.0,hand_tilt=0.0,hand_twist=0.0;
```

Listing 3: Global variables

```
// IK target for the robot's wrist point - when using IK mode
GLfloat p_target[3]={42,0,50};
// These three angles are used to give the direction that the end-effector (the tool)
// at the end of the Robot Arm should be pointing in. (When the CCD algorithm is in use.)
GLfloat hand_pan=0.0,hand_tilt=0.0,hand_twist=0.0;

// Direction and distance of the camera from the object.
//(Used to set up the viewing transformation )
static GLdouble er=200.0,theta=0.0,phi=0.0;
```

Listing 4: Global variables

```
// Specify the dimensions of the model robot
GLfloat baseHeight=BASEHEIGHT
       ,upperArm=UPPERARM
       ,lowerArm=LOWERARM
       ,Hand=HAND;
```

Listing 5: Global variables

```
// transformations as determined by the OpenGL matrices
// All are (in C element order and  p = Mp transformation type)
// These transformations are all obtained by
// reading them from the OpenGL transform stack.
GLfloat m_camera[4][4],  // Camera transformation matrix
        m_endpoint[4][4]={1,0,0,0, 0,1,0,0, 0,0,1,0, 0,0,0,1},  // endpoint
        m_wristpoint[4][4]={1,0,0,0, 0,1,0,0, 0,0,1,0, 0,0,0,1},// wrist
        m_lowerarm[4][4]={1,0,0,0, 0,1,0,0, 0,0,1,0, 0,0,0,1},
        m_upperarm[4][4]={1,0,0,0, 0,1,0,0, 0,0,1,0, 0,0,0,1},
        p_upperarm[3],   // coordinate of point of upper arm
        p_lowerarm[3],   // coodinate of point of lowe arm
        p_wrist[3];      // coordinate of ROBOT wrist
```

Listing 6: Global variables

```c
// This group of variables are used in the CCD IK algorithm - they mimic the
// forward kinematics variables used to position and draw
// the robot. However the robot is drawn using OpenGL transformation.
// The IK algorithm needs to use its own transformations
// while it is calculating the correct joint angles.
float   c_upperarm[4]={0,0,BASEHEIGHT,1},
        c_lowerarm[4]={0,0,BASEHEIGHT+UPPERARM,1},
        c_wrist[4]={0,0,BASEHEIGHT+UPPERARM+LOWERARM,1},
        c_hand[4]={0,0,BASEHEIGHT+UPPERARM+LOWERARM+HAND,1},
        // this is a point used for the hand to point at, in the algorithm
        c_target[4]={TOFFSET,0,BASEHEIGHT+UPPERARM+LOWERARM,1},
        II[4][4]={1,0,0,0, 0,1,0,0, 0,0,1,0, 0,0,0,1},   // Identity matrix
        B[4][4],  // base transform
        U[4][4],  // upper arm transform
        L[4][4],  // lower arm transform
        H[4][4],  // hand transform
        // composites
        BU[4][4],
        BUL[4][4],
        BULH[4][4],
        // matrix giving orienation of the tool
        ToolO[4][4]={1,0,0,0, 0,1,0,0,  0,0,1,0,  0,0,0,1};
```

Listing 7: Global variables

Note:

The files asociated with this project are located in folder GL-RobotSimulator1. The files in folder GL-RobotSimulator2 relate to an extended version of the program that uses a *SensAble Phantom Haptic device* to set the orientation of the end effector of the robot. This version displays two robots and can use up to two Phantoms to control each of the robots.

Do not use this program unless you have access to a Phantom device and the Open-Haptics developers SDK.