Instituto Tecnológico de Estudios Superiores de Monterrey

Robótica Aplicada

Proyecto Final

“Forwards and Inverse Kinematics of 6 DoF robot”

Erick Osvaldo Castelazo Guerra A01273870 ISDR

Profesor: Jesús Arturo Escobedo Cabello

Abstract

When using a robotic manipulator, knowing the pose of its end effector and the values of each of its joints is useful to be able to control it. When programing a trajectory that the robot will follow, these parameters are needed to do the calculation. Forwards and inverse kinematics solve this problem. Forward’s kinematics obtains the pose of the end effector of a robot relative to the frame of reference of its base given the values of each of its joints. Inverse Kinematics solves the opposite problem, obtaining the joint values given the position and orientation of the end effector. In this project, the forwards an inverse kinematics of a robotic arm will be calculated.

Objectives

Calculate the forwards and inverse kinematics of a 6 degrees of freedom articulated robot. Create a program in MATLAB that automates this procedure. The selected robot for this project is the ABB IRB 140 6kg 0.81m which is available for simulation in the Robot Studio software.

Introduction

In order to make a program to calculate the forwards and backwards kinematic of the selected robot, first a geometrical analysis of the dimensions of the robotic arm will be done to generate a diagram that represent each of the joints, links and frames of reference of the robot. For the forward kinematics, the Denavit and Hartenberg method will be employed to obtain the pose of the end effector relative to the base of the robot. For inverse kinematics, a geometrical method will be employed to obtain the joint values. The program will be coded in Matlab, and its results compared with the results obtained in Robot Studio. Once the results are verified, we can conclude that the calculations for the forwards and inverse kinematics were correct.

Theorical Framework

Forward kinematics

The forward kinematics problem is to obtain the pose of the end effector of a robot. The pose is represented by the homogeneous transform matrix. This matrix contains the information about the translation and the rotation to move from the base frame of reference to the end effector frame of reference. One way to compute the forward kinematics is by using the Denavit and Hartenberg parameters. These parameters describe how to move from one frame of reference to the next one. The 4 parameters are described as follows:

- theta : joint angle, the rotation in the z-axis
- d: joint offset, the translation in the z-axis
- a: link length, the translation in the x-axis
- alpha: link twist, the rotation in the x-axis

The movement to go from the frame 0 to the frame N is described by the following formula:

$$T_{0N} = Rz(\theta_1)Tz(d_1)Tx(a_1)Rx(\alpha_1) + Rz(\theta_2)Tz(d_2)Tx(a_2)Rx(\alpha_2) + \cdots + Rz(\theta_N)Tz(d_N)Tx(a_N)Rx(\alpha_N)$$

Inverse Kinematics

Solving inverse kinematics is a more complex problem since it has an infinite number of solutions. One way to solve it is geometrically, obtaining equations to solve for each joint value.

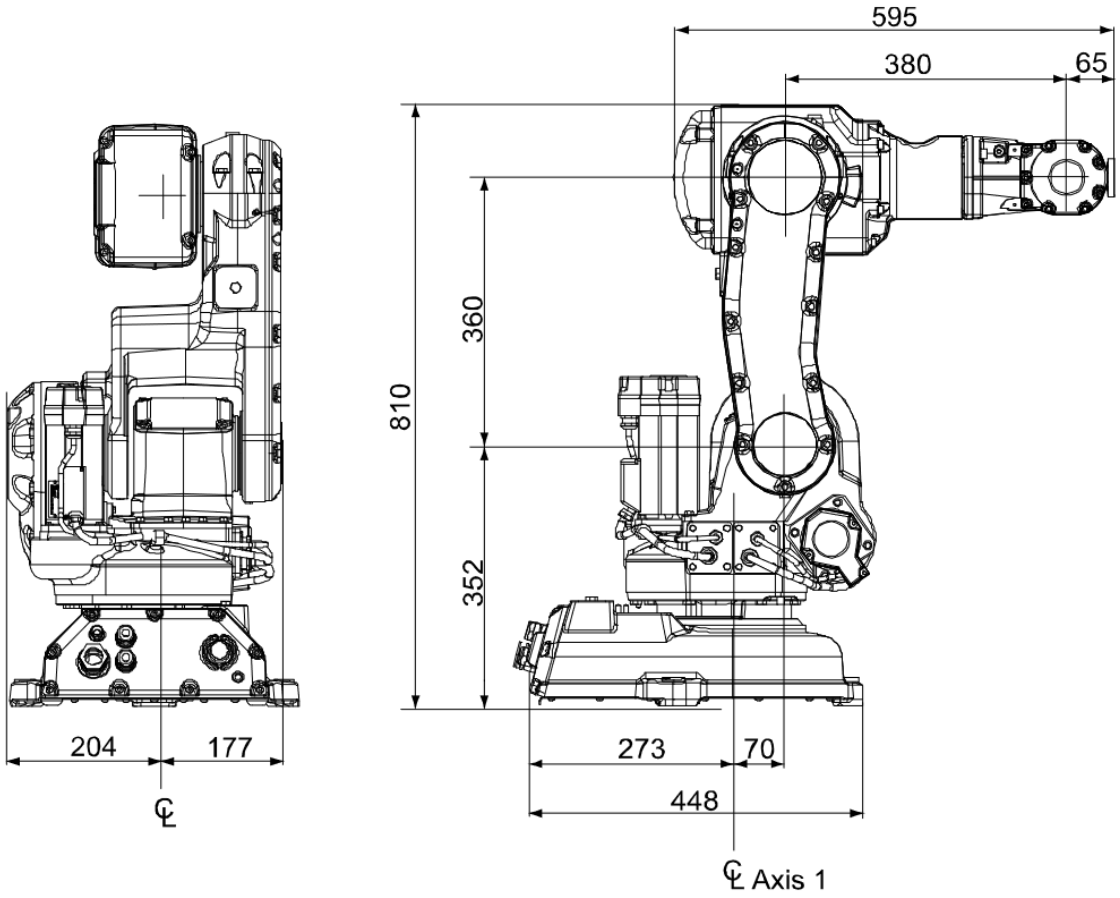
To do Inverse kinematics of a 6 DoF robotic arm we have to make the following assumption: The first three joints are entirely responsible for positioning the end effector, and any additional joints are responsible for orienting the end effector.

The steps to compute the inverse kinematics are described as follows:

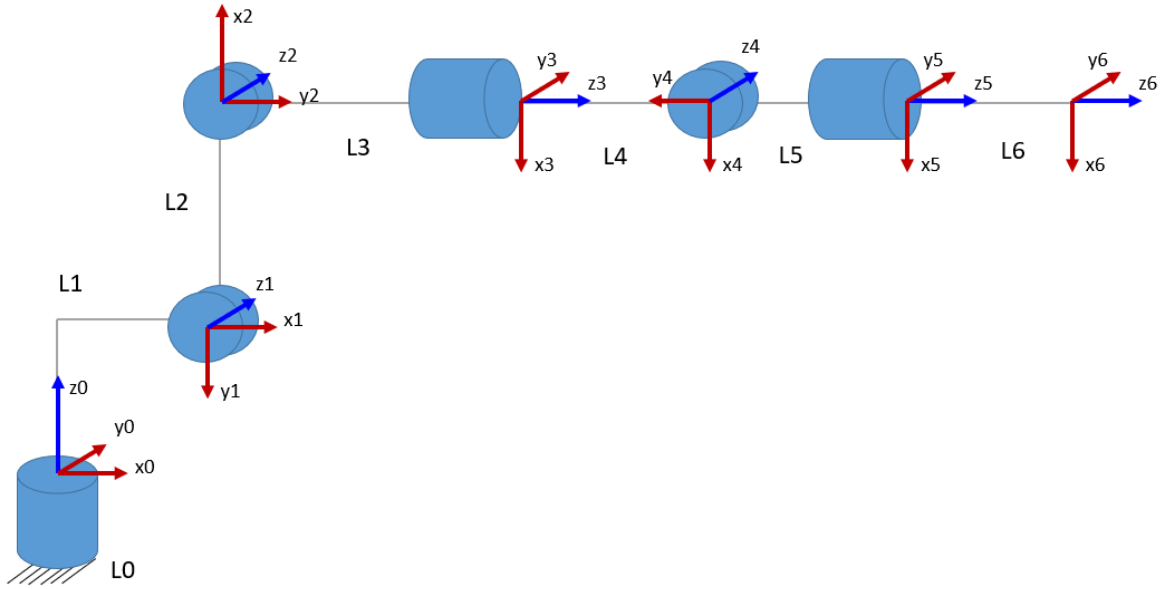
1. Draw a kinematic diagram of only the first 3 joints and do inverse kinematics for position.
2. Do forward kinematics on the first three joints to get the rotation matrix R03.
3. Find the inverse of the RO3 matrix.
4. Do forward kinematics on the last three joints and pull out the rotation part, R36.
5. Specify what you want the rotation matrix R06 to be. R06 will be the orientation the user will input to the program.
6. Given a desired X,Y,Z position, solve for the first three joints using the inverse kinematics equations from Step 1.
7. Plug in those variables and use the rotation matrix to solve for the last three joints.

Development

Dimensions of the ABB IRB 140



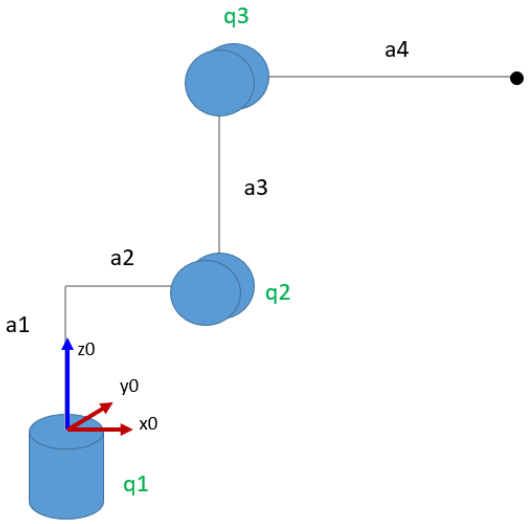
Forward Kinematics Analysis to obtain end effector pose.



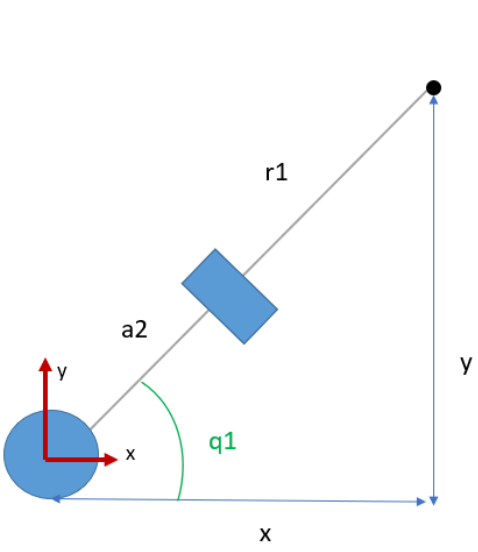
Denavit and Hartenberg parameters

	Theta i	di	ai	Alpha i
1	q1	352	70	-90°
2	q2 – 90°	0	360	0
3	q3+180°	0	0	90°
4	q4	380	0	-90°
5	q5	0	0	90°
6	q6	65	0	0

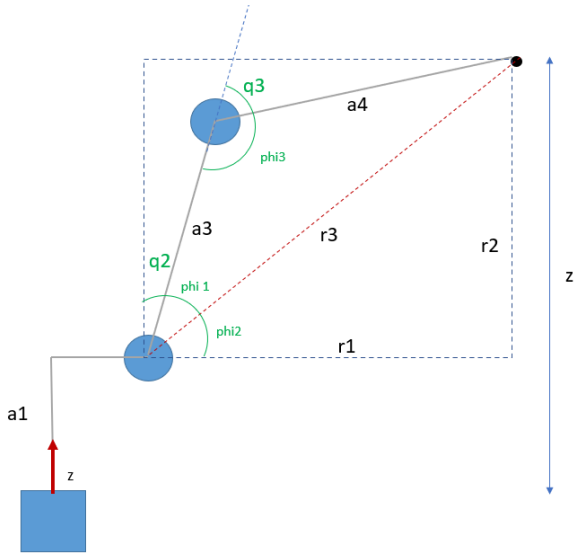
Backwards Kinematics Analysis to obtain position joints (q1, q2, q3)



Backwards Kinematics of first 3 joints



Top view



Side view

$$r1 = \sqrt{x^2 + y^2} - a2$$

$$r2 = z - a1$$

$$r3 = \sqrt{r1^2 + r2^2}$$

$$\phi_1 = cos^{-1}\left(\frac{a4^2 - a3^2 - r3^2}{-2 * a3 * r3}\right)$$

$$\phi_2 = tan^{-1}\left(\frac{r2}{r1}\right)$$

$$\phi_3 = cos^{-1}\left(\frac{r3^2 - a4^2 - a3^2}{-2 * a4 * a3}\right)$$

$$\theta_1 = tan^{-1}\left(\frac{y}{x}\right)$$

$$\theta_2 = 90 - \phi_1 - \phi_2$$

$$\theta_3 = 90 - \phi_3$$

Elbow up configuration.

Backward kinematics analysis to obtain orientation joints (q4,q5,q6)

To obtain the inverse kinematics of the orientation of the arm wrist, we first have to obtain the rotation matrix R03. This can be obtained with the D-H parameters of the first 3 joints that we got from the forward kinematics. Once we have R03 we do the following operations to obtain the joint values:

$$R36 = R03^{-1} * R06$$

R06 is our desired end effector pose.

R36 is the rotation matrix that describes the rotation of the last 3 joints(q4,q5,q6). The following Matlab code was used to obtain R36:

```
syms q4 q5 q6
R34 = rotz(q4) *rotx(-pi/2);
R45 = rotz(q5) *rotx(pi/2);
R56 = rotz(q6) *rotx(0);

R36 = R34*R45*R56;
```

```
R36 =
[cos(q4)*cos(q5)*cos(q6) - sin(q4)*sin(q6), - cos(q6)*sin(q4) - cos(q4)*cos(q5)*sin(q6), cos(q4)*sin(q5)]
[cos(q4)*sin(q6) + cos(q5)*cos(q6)*sin(q4),   cos(q4)*cos(q6) - cos(q5)*sin(q4)*sin(q6), sin(q4)*sin(q5)]
[                                -cos(q6)*sin(q5),                                          sin(q5)*sin(q6),          cos(q5)]
```

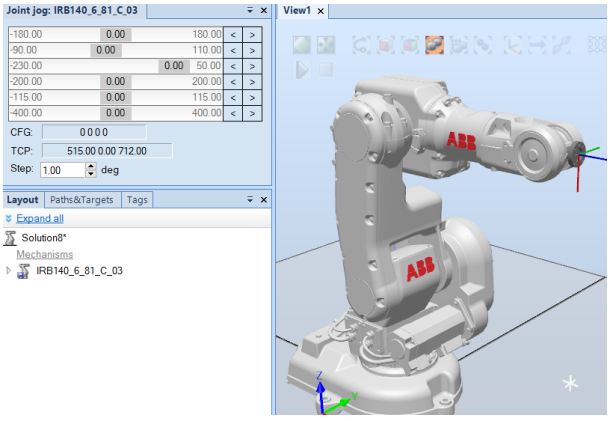
Matlab code to obtain each joint value:

```
q_5 = acosd(R36(3,3));
q_4 = atand(R36(2,3)/R36(1,3));
q_6 = atand(-R36(3,2)/R36(3,1));
```

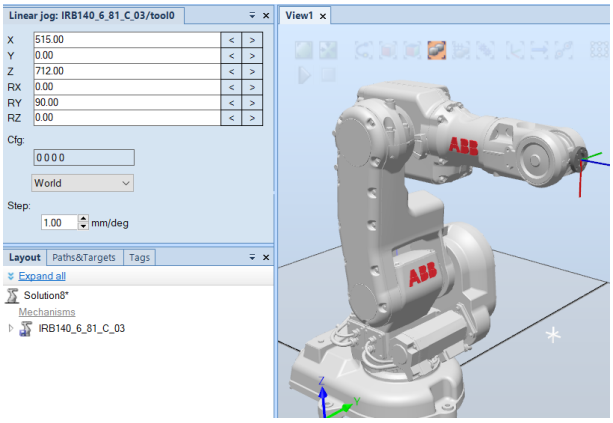
wrist down configuration.

Results

The forward kinematics was obtained using the Denavit and Hartenberg parameters to get the pose of the end effector of the robotic arm. The inverse kinematics was obtained using the elbow-up wrist-down configuration. Both programs were tested and generated the expected results.



Joint jog values



Linear jog values

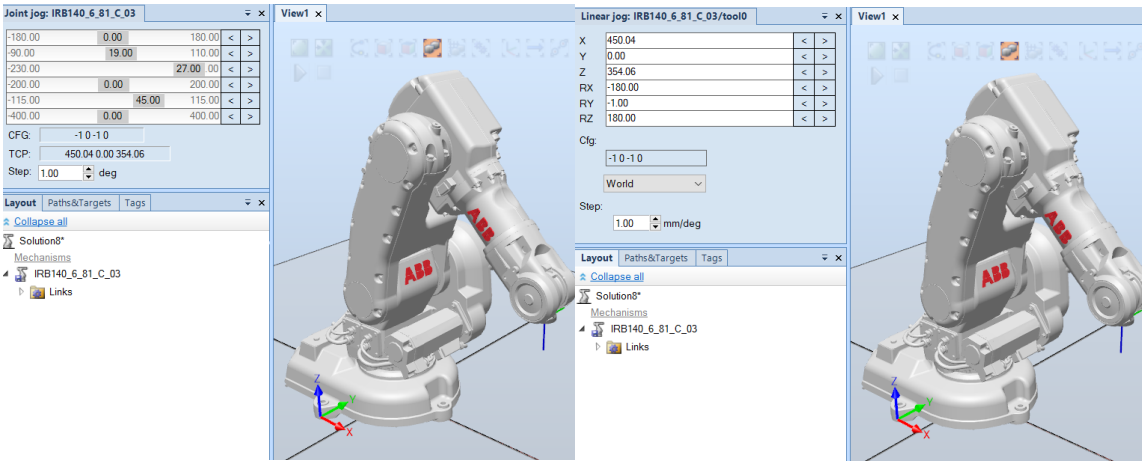
```
v2ProyectoFinal_BackwardsK
Insert end-effector position(x,y,z)
X: 515
Y: 0
Z: 712
Insert end-effector
orientation(rx,ry,rz)
rx: 0
ry: 90
rz: 0
Joint values
q1: 0.00
q2: 0.00
q3: -0.00
q4: 0.00
q5: 0.00
q6: 0.00
```

Inverse Kinematics program output

```
v2ProyectoFinal_ForwardK
Insert value of q1: 0
Insert value of q2: 0
Insert value of q3: 0
Insert value of q4: 0
Insert value of q5: 0
Insert value of q6: 0
Homogeneus transform matrix:
     0     0     1   515
     0     1     0     0
    -1     0     0   712
     0     0     0     1

position vector(x,y,z):
515.00   0.00   712.00
Orientation(rx,ry,rz):
rx: 0.00
ry: 90.00
rz: 0.00
```

Forward Kinematics program output



Joint jog values

Linear jog values

```
v2ProyectoFinal_BackwardsK
Insert end-effector position(x,y,z)
X: 450.04
Y: 0
Z: 354.04
Insert end-effector
orientation(rx,ry,rz)
rx: -180
ry: -1
rz: 180
Joint values
q1: 0.00
q2: 19.00
q3: 27.00
q4: 0.00
q5: 45.00
q6: 0.00
```

```
v2ProyectoFinal_ForwardK
Insert value of q1: 0
Insert value of q2: 19
Insert value of q3: 27
Insert value of q4: 0
Insert value of q5: 45
Insert value of q6: 0
Homogeneus transform matrix:
   -0.9998         0   -0.0175  450.0403
         0    1.0000         0         0
    0.0175         0   -0.9998  354.0475
         0         0         0    1.0000

position vector(x,y,z):
450.04  0.00  354.05
Orientation(rx,ry,rz):
rx: 180.00
ry: -1.00
rz: 180.00
```

Inverse Kinematics program output

Forward Kinematics program output

Conclusions

In this project a practical solution was implemented to obtain the forwards and inverse kinematics of a real robot, the ABB IRB 140. With this project, many of the topics studied during the semester were put to the test. Understanding the forwards and inverse kinematics problem is important since many robotics applications rely on these concepts. The selected robot was analyzed to obtain the necessary equations to compute and program a solution for the problem and the results were compared to the results given by robot studio.

References

Dr. Angela Sodemann, (2018). Robotics 2 U1 (Kinematics) S5 (Inverse Kinematics). Junio 2021, de RoboGrok Sitio web: http://robogrok.com/

Peter Corke. (2014). Convert rotation and translation to homogeneous transform. Junio 2021, de PeterCorke.com Sitio web: http://www.petercorke.com/RTB/r9/html/rt2tr.html

Annexes

Matlab program for forward Kinematics

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%  FORWARDS KINEMATICS   %%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

prompt = 'Insert value of q1: ';q1 = input(prompt);
prompt = 'Insert value of q2: ';q2 = input(prompt);
prompt = 'Insert value of q3: ';q3 = input(prompt);
prompt = 'Insert value of q4: ';q4 = input(prompt);
prompt = 'Insert value of q5: ';q5 = input(prompt);
prompt = 'Insert value of q6: ';q6 = input(prompt);


%       rotz                        tx   tz        rotx
%       theta                       aj   dj        alpha
A1= r2t(rotz(q1,'deg'))    *transl(70,0,352) *r2t(rotx(-
90,'deg'));
A2= r2t(rotz(q2-90,'deg')) *transl(360,0,0)  *r2t(rotx(0,'deg'));
A3= r2t(rotz(q3+180,'deg'))    *transl(0,0,0)
*r2t(rotx(90,'deg'));
A4= r2t(rotz(q4,'deg'))    *transl(0,0,380)  *r2t(rotx(-
90,'deg'));
A5= r2t(rotz(q5,'deg'))    *transl(0,0,0)    *r2t(rotx(90,'deg'));
A6= r2t(rotz(q6,'deg'))    *transl(0,0,65)   *r2t(rotx(0,'deg'));

T06 = A1*A2*A3*A4*A5*A6;

fprintf("Homogeneus transform matrix: \n");
disp(T06);
fprintf("position vector(x,y,z): \n");
fprintf("%.2f  %.2f  %.2f\n",T06(1,4),T06(2,4),T06(3,4));
fprintf("Orientation(rx,ry,rz): \n");

rx= atan2d(T06(3,2),T06(3,3));
ry= atan2d(-T06(3,1),sqrt(T06(3,2)^2 + T06(3,3)^2));
rz= atan2d(T06(2,1),T06(1,1));
fprintf("rx: %.2f\nry: %.2f\nrz: %.2f\n",rx,ry,rz);
```

Matlab program for inverse Kinematics

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%  BACKWARDS KINEMATICS  %%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

fprintf("Insert end-effector position(x,y,z)\n");
prompt = 'X: ';Xt = input(prompt);
prompt = 'Y: ';Yt = input(prompt);
prompt = 'Z: ';Zt = input(prompt);
fprintf("Insert end-effector orientation(rx,ry,rz)\n");
prompt = 'rx: ';rx = input(prompt);
prompt = 'ry: ';ry = input(prompt);
prompt = 'rz: ';rz = input(prompt);
R06 = rotx(rx,'deg')*roty(ry,'deg')*rotz(rz,'deg');


% %End effector coordinates(Xt,Yt,Zt)
% Xt=640;
% Yt=-316;
% Zt=724;
% %End effector orientation(R06)
% R06=[0 0 1;0 1 0;-1 0 0];%rx=0  ry=90  rz=0

%Wrist point coordinates(X,Y,Z)
pw = [Xt; Yt; Zt] + R06*[0;0;-65];
X=pw(1);
Y=pw(2);
Z=pw(3);

%Link Dimensions
a1=352; a2=70; %link 1
a3=360;        %link 2
a4=380;        %link 3

%%Backwards Kinematics Position(q1,q2,3)%%
r1 = sqrt(X^2+Y^2) -a2;
r2 = Z - a1;
r3 = sqrt(r1^2+r2^2);
%elbow up
phi1 = acosd((a4^2-a3^2-r3^2)/(-2*a3*r3));
phi2 = atan2d(r2,r1);
phi3 = acosd((r3^2-a4^2-a3^2)/(-2*a4*a3));

q_1 = atan2d(Y,X);
q_2 = 90 - phi2 - phi1;
q_3 = 90 - phi3;

%%Backwards kinematics Orientation(q4,q5,q6)%%
%Rotation R03
R_A1= rotz(q_1,'deg')    *rotx(-90,'deg');
R_A2= rotz(q_2-90,'deg') *rotx(0,'deg');
R_A3= rotz(q_3+180,'deg')*rotx(90,'deg');
R03 = R_A1*R_A2*R_A3;
```

```
R36 = inv(R03)*R06;

q_5 = acosd(R36(3,3));
q_4 = atand(R36(2,3)/R36(1,3));
q_6 = atand(-R36(3,2)/R36(3,1));

q_4(isnan(q_4)) = 0;
q_6(isnan(q_6)) = 0;

fprintf("Joint values\n");
%position joints
fprintf('q1: %.2f\n',q_1);
fprintf('q2: %.2f\n',q_2);
fprintf('q3: %.2f\n',q_3);
%orientation joints
fprintf('q4: %.2f\n',q_4);
fprintf('q5: %.2f\n',q_5);
fprintf('q6: %.2f\n',q_6);
```