Table of Contents

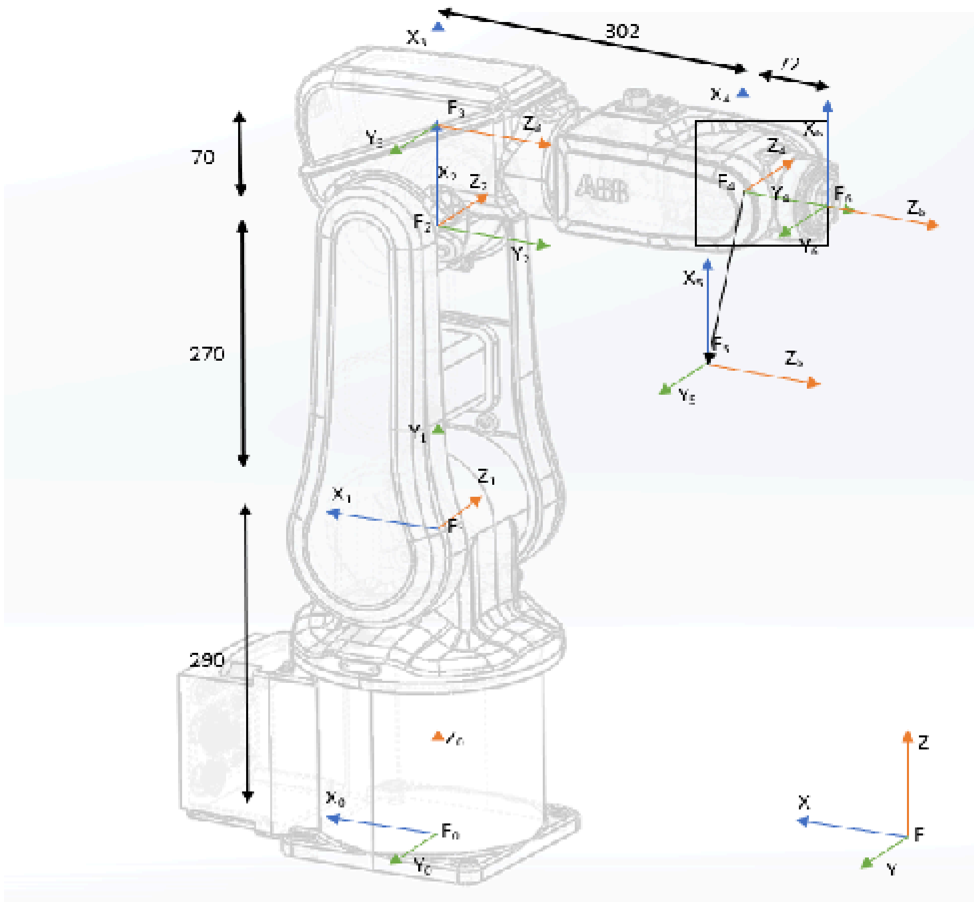
ABB IRB 120 Robot

This program simulates the forward kinematics of an ABB IRB 120 robot. **ALL UNITS USED ARE IN MM AND DEGREES**

```
clc
clear all
syms q1 q2 q3 q4 q5 q6
```

Question 1

On a line drawing of the robot (such as from the figure below of the manual), define and label coordinate frames for the base, each link, and the tip. You should do this using the D-H convention from this class. Clearly show the location of the origin and the direction for each axis of each frame that you define. Be sure to use the axis numbering, base & tool frame, and rotation direction conventions provided for the robot and matching that of the simulation. As previously noted, you are expected to use the D-H convention presented in this class – some other references may have variants in indexing that can cause confusion so be careful with outside references & examples



Declaring various variables. The final transformation matrices for the n dofs with respect to the origin of the manipulator are stored in every successive nth space of the *4x4xn* matrix **Transforms**. Similarly the final transformation matrices for the n dofs with respect to the n-1th frame are stored in every successive nth space of the *4x4xn* matrix **Individual_Transforms**. **ALL UNITS USED ARE IN MM AND DEGREES**

```
global n_dof;
q = [q1 q2 q3 q4 q5 q6].';
[n_dof,m] = size(q);
Individual_Transforms = sym(zeros(4,4,n_dof));
Transforms = sym(zeros(4,4,n_dof));
```

Question 2

Determine the D-H parameters for the robot and show a clearly labeled table. Be sure to include units, identify the joint variables, and check your joint directions. Generate Matlab variables for each of the 4 parameters as nx1 vectors.

The following section shows the DH Parameters in vector form. **ALL UNITS USED ARE IN MM AND DEGREES**

```
thetas = q;
thetas(2,1) = q2 + 90;
thetas
```

```
d = [290 0 0 302 0 72].'
a = [0 270 70 0 0 0].'
alpha = [90 0 -90 90 -90 0].'
```

```
thetas =

      q1
 q2 + 90
      q3
      q4
      q5
      q6


d =

   290
     0
     0
   302
     0
    72


a =

     0
   270
    70
     0
     0
     0


alpha =

    90
     0
   -90
    90
   -90
     0
```

Question 3

a)

Create a Matlab function that takes a set of DH parameters (each of the 4 parameters as an nx1 vector) for a link and returns a 4x4 homogeneous transformation matrix using the following syntax

```
T = dhparam2matrix(theta, d, a, alpha)
```

```
function [trans] = dhparam2matrix(theta, d, a, alpha)
    %DHPARAM2MATRIX This function returns a Homogenous Transformation
 matrix
    % given the input parameters a, alpha, theta and d derived from
 the
    % DH Parameters of the robot.
    %   Please enter all values in degrees and not radians.

    cos_alpha = @cosd;
    sin_alpha = @sind;
    cos_theta = @cosd;
    sin_theta = @sind;

    a11 = cos_theta(theta);
    a12 = -sin_theta(theta)*cos_alpha(alpha);
    a13 = sin_theta(theta)*sin_alpha(alpha);
    a14 = a*cos_theta(theta);
    a21 = sin_theta(theta);
    a22 = cos_theta(theta)*cos_alpha(alpha);
    a23 = -cos_theta(theta)*sin_alpha(alpha);
    a24 = a*sin_theta(theta);
    a31 = 0;
    a32 = sin_alpha(alpha);
    a33 = cos_alpha(alpha);
    a34 = d;
    scl = 1;

    trans = [a11 a12 a13 a14;
             a21 a22 a23 a24;
             a31 a32 a33 a34;
             0   0   0   scl];
end
```

b)

Write a loop that goes through all n links to generate the respective corresponding transformation matrix

A for loop adds the transformation of each frame wrt previous frames to a n dimension 4x4 matrix called Individual_Transforms and the transformation of each frame wrt the base frame to a n dimensional 4x4 matrix called Transforms. **ALL UNITS USED ARE IN MM AND DEGREES**

```
for i = 1:n_dof
    Individual_Transforms(:,:,i) = dhparam2matrix(thetas(i,1), d(i,1),
 a(i,1), alpha(i,1));
    if i == 1
        Transforms(:,:,1) = Individual_Transforms(:,:,1);
    else
        Transforms(:,:,i) =
 Transforms(:,:,i-1)*Individual_Transforms(:,:,i);
    end
end
```

c)

Show the symbolic 4x4 SE(3) transformation matrices corresponding to each of the intermediate transformations. **ALL UNITS USED ARE IN MM AND DEGREES**

```
T01 = Individual_Transforms(:,:,1)
T12 = Individual_Transforms(:,:,2)
T23 = Individual_Transforms(:,:,3)
T34 = Individual_Transforms(:,:,4)
T45 = Individual_Transforms(:,:,5)
T56 = Individual_Transforms(:,:,6)
```

```
T01 =

[ cos((pi*q1)/180), 0,  sin((pi*q1)/180),   0]
[ sin((pi*q1)/180), 0, -cos((pi*q1)/180),   0]
[                0, 1,                 0, 290]
[                0, 0,                 0,   1]


T12 =

[ cos((pi*(q2 + 90))/180), -sin((pi*(q2 + 90))/180), 0,
 270*cos((pi*(q2 + 90))/180)]
[ sin((pi*(q2 + 90))/180),  cos((pi*(q2 + 90))/180), 0,
 270*sin((pi*(q2 + 90))/180)]
[                       0,                        0, 1,
            0]
[                       0,                        0, 0,
            1]


T23 =

[ cos((pi*q3)/180),  0, -sin((pi*q3)/180), 70*cos((pi*q3)/180)]
[ sin((pi*q3)/180),  0,  cos((pi*q3)/180), 70*sin((pi*q3)/180)]
[                0, -1,                 0,                   0]
[                0,  0,                 0,                   1]


T34 =

[ cos((pi*q4)/180), 0,  sin((pi*q4)/180),   0]
[ sin((pi*q4)/180), 0, -cos((pi*q4)/180),   0]
[                0, 1,                 0, 302]
[                0, 0,                 0,   1]


T45 =

[ cos((pi*q5)/180),  0, -sin((pi*q5)/180), 0]
[ sin((pi*q5)/180),  0,  cos((pi*q5)/180), 0]
```

```
[                0,  -1,                 0,  0]
[                0,   0,                 0,  1]


T56 =

[ cos((pi*q6)/180), -sin((pi*q6)/180), 0,  0]
[ sin((pi*q6)/180),  cos((pi*q6)/180), 0,  0]
[                0,                 0, 1, 72]
[                0,                 0, 0,  1]
```

Question 4

Solve for the composite transformation representing the forward kinematics from base to tip (T0n). Be sure to show your work and simplify the solution as much as possible. This will be a large symbolic 4x4 matrix. **ALL UNITS USED ARE IN MM AND DEGREES**

```
T06 = Transforms(:,:,6)


T06 =

[ -
 cos((pi*q6)/180)*(sin((pi*q5)/180)*(cos((pi*q1)/180)*cos((pi*q3)/180)*sin((pi*(q2
 + 90))/180) + cos((pi*q1)/180)*sin((pi*q3)/180)*cos((pi*(q2 +
 90))/180)) + cos((pi*q5)/180)*(sin((pi*q1)/180)*sin((pi*q4)/180)
 - cos((pi*q4)/180)*(cos((pi*q1)/180)*cos((pi*q3)/180)*cos((pi*(q2
 + 90))/180) - cos((pi*q1)/180)*sin((pi*q3)/180)*sin((pi*(q2 +
 90))/180)))) -
 sin((pi*q6)/180)*(sin((pi*q4)/180)*(cos((pi*q1)/180)*cos((pi*q3)/180)*cos((pi*(q2
 + 90))/180) - cos((pi*q1)/180)*sin((pi*q3)/180)*sin((pi*(q2
 + 90))/180)) + cos((pi*q4)/180)*sin((pi*q1)/180)),
 sin((pi*q6)/180)*(sin((pi*q5)/180)*(cos((pi*q1)/180)*cos((pi*q3)/180)*sin((pi*(q2
 + 90))/180) + cos((pi*q1)/180)*sin((pi*q3)/180)*cos((pi*(q2 +
 90))/180)) + cos((pi*q5)/180)*(sin((pi*q1)/180)*sin((pi*q4)/180)
 - cos((pi*q4)/180)*(cos((pi*q1)/180)*cos((pi*q3)/180)*cos((pi*(q2
 + 90))/180) - cos((pi*q1)/180)*sin((pi*q3)/180)*sin((pi*(q2 +
 90))/180)))) -
 cos((pi*q6)/180)*(sin((pi*q4)/180)*(cos((pi*q1)/180)*cos((pi*q3)/180)*cos((pi*(q2
 + 90))/180) - cos((pi*q1)/180)*sin((pi*q3)/180)*sin((pi*(q2
 + 90))/180)) + cos((pi*q4)/180)*sin((pi*q1)/180)),
 sin((pi*q5)/180)*(sin((pi*q1)/180)*sin((pi*q4)/180) -
 cos((pi*q4)/180)*(cos((pi*q1)/180)*cos((pi*q3)/180)*cos((pi*(q2
 + 90))/180) - cos((pi*q1)/180)*sin((pi*q3)/180)*sin((pi*(q2 +
 90))/180))) -
 cos((pi*q5)/180)*(cos((pi*q1)/180)*cos((pi*q3)/180)*sin((pi*(q2
 + 90))/180) + cos((pi*q1)/180)*sin((pi*q3)/180)*cos((pi*(q2
 + 90))/180)), 270*cos((pi*q1)/180)*cos((pi*(q2 + 90))/180)
 + 72*sin((pi*q5)/180)*(sin((pi*q1)/180)*sin((pi*q4)/180) -
 cos((pi*q4)/180)*(cos((pi*q1)/180)*cos((pi*q3)/180)*cos((pi*(q2
 + 90))/180) - cos((pi*q1)/180)*sin((pi*q3)/180)*sin((pi*(q2 +
 90))/180))) -
```

```
72*cos((pi*q5)/180)*(cos((pi*q1)/180)*cos((pi*q3)/180)*sin((pi*(q2
+ 90))/180) + cos((pi*q1)/180)*sin((pi*q3)/180)*cos((pi*(q2 +
90))/180)) + 70*cos((pi*q1)/180)*cos((pi*q3)/180)*cos((pi*(q2 +
90))/180) - 302*cos((pi*q1)/180)*cos((pi*q3)/180)*sin((pi*(q2 +
90))/180) - 302*cos((pi*q1)/180)*sin((pi*q3)/180)*cos((pi*(q2 +
90))/180) - 70*cos((pi*q1)/180)*sin((pi*q3)/180)*sin((pi*(q2 +
90))/180)]
[ -
cos((pi*q6)/180)*(sin((pi*q5)/180)*(cos((pi*q3)/180)*sin((pi*q1)/180)*sin((pi*(q2
+ 90))/180) + sin((pi*q1)/180)*sin((pi*q3)/180)*cos((pi*(q2 +
90))/180)) -
cos((pi*q5)/180)*(cos((pi*q4)/180)*(cos((pi*q3)/180)*sin((pi*q1)/180)*cos((pi*(q2
+ 90))/180) - sin((pi*q1)/180)*sin((pi*q3)/180)*sin((pi*(q2
+ 90))/180)) + cos((pi*q1)/180)*sin((pi*q4)/180))) -
sin((pi*q6)/180)*(sin((pi*q4)/180)*(cos((pi*q3)/180)*sin((pi*q1)/180)*cos((pi*(q2
+ 90))/180) - sin((pi*q1)/180)*sin((pi*q3)/180)*sin((pi*(q2
+ 90))/180)) - cos((pi*q1)/180)*cos((pi*q4)/180)),
sin((pi*q6)/180)*(sin((pi*q5)/180)*(cos((pi*q3)/180)*sin((pi*q1)/180)*sin((pi*(q2
+ 90))/180) + sin((pi*q1)/180)*sin((pi*q3)/180)*cos((pi*(q2 +
90))/180)) -
cos((pi*q5)/180)*(cos((pi*q4)/180)*(cos((pi*q3)/180)*sin((pi*q1)/180)*cos((pi*(q2
+ 90))/180) - sin((pi*q1)/180)*sin((pi*q3)/180)*sin((pi*(q2
+ 90))/180)) + cos((pi*q1)/180)*sin((pi*q4)/180))) -
cos((pi*q6)/180)*(sin((pi*q4)/180)*(cos((pi*q3)/180)*sin((pi*q1)/180)*cos((pi*(q2
+ 90))/180) - sin((pi*q1)/180)*sin((pi*q3)/180)*sin((pi*(q2
+ 90))/180)) - cos((pi*q1)/180)*cos((pi*q4)/180)), -
cos((pi*q5)/180)*(cos((pi*q3)/180)*sin((pi*q1)/180)*sin((pi*(q2
+ 90))/180) + sin((pi*q1)/180)*sin((pi*q3)/180)*cos((pi*(q2 +
90))/180)) -
sin((pi*q5)/180)*(cos((pi*q4)/180)*(cos((pi*q3)/180)*sin((pi*q1)/180)*cos((pi*(q2
+ 90))/180) - sin((pi*q1)/180)*sin((pi*q3)/180)*sin((pi*(q2
+ 90))/180)) + cos((pi*q1)/180)*sin((pi*q4)/180)),
270*sin((pi*q1)/180)*cos((pi*(q2 + 90))/180) -
72*cos((pi*q5)/180)*(cos((pi*q3)/180)*sin((pi*q1)/180)*sin((pi*(q2
+ 90))/180) + sin((pi*q1)/180)*sin((pi*q3)/180)*cos((pi*(q2 +
90))/180)) -
72*sin((pi*q5)/180)*(cos((pi*q4)/180)*(cos((pi*q3)/180)*sin((pi*q1)/180)*cos((pi*
+ 90))/180) - sin((pi*q1)/180)*sin((pi*q3)/180)*sin((pi*(q2
+ 90))/180)) + cos((pi*q1)/180)*sin((pi*q4)/180)) +
70*cos((pi*q3)/180)*sin((pi*q1)/180)*cos((pi*(q2 + 90))/180) -
302*cos((pi*q3)/180)*sin((pi*q1)/180)*sin((pi*(q2 + 90))/180) -
302*sin((pi*q1)/180)*sin((pi*q3)/180)*cos((pi*(q2 + 90))/180) -
70*sin((pi*q1)/180)*sin((pi*q3)/180)*sin((pi*(q2 + 90))/180)]
[


 cos((pi*q6)/180)*(sin((pi*q5)/180)*(cos((pi*q3)/180)*cos((pi*(q2
+ 90))/180) - sin((pi*q3)/180)*sin((pi*(q2 + 90))/180)) +
cos((pi*q4)/180)*cos((pi*q5)/180)*(cos((pi*q3)/180)*sin((pi*(q2
+ 90))/180) + sin((pi*q3)/180)*cos((pi*(q2 + 90))/180))) -
sin((pi*q4)/180)*sin((pi*q6)/180)*(cos((pi*q3)/180)*sin((pi*(q2
+ 90))/180) + sin((pi*q3)/180)*cos((pi*(q2 + 90))/180)),
```

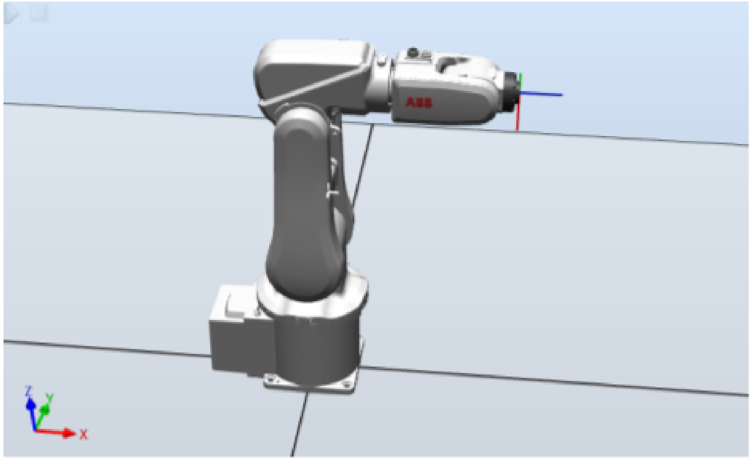
```
-
sin((pi*q6)/180)*(sin((pi*q5)/180)*(cos((pi*q3)/180)*cos((pi*(q2
+ 90))/180) - sin((pi*q3)/180)*sin((pi*(q2 + 90))/180)) +
cos((pi*q4)/180)*cos((pi*q5)/180)*(cos((pi*q3)/180)*sin((pi*(q2
+ 90))/180) + sin((pi*q3)/180)*cos((pi*(q2 + 90))/180))) -
cos((pi*q6)/180)*sin((pi*q4)/180)*(cos((pi*q3)/180)*sin((pi*(q2 +
90))/180) + sin((pi*q3)/180)*cos((pi*(q2 + 90))/180)),

cos((pi*q5)/180)*(cos((pi*q3)/180)*cos((pi*(q2
+ 90))/180) - sin((pi*q3)/180)*sin((pi*(q2 + 90))/180)) -
cos((pi*q4)/180)*sin((pi*q5)/180)*(cos((pi*q3)/180)*sin((pi*(q2
+ 90))/180) + sin((pi*q3)/180)*cos((pi*(q2 + 90))/180)),

270*sin((pi*(q2 + 90))/180) + 302*cos((pi*q3)/180)*cos((pi*(q2
+ 90))/180) + 70*cos((pi*q3)/180)*sin((pi*(q2 +
90))/180) + 70*sin((pi*q3)/180)*cos((pi*(q2 + 90))/180)
- 302*sin((pi*q3)/180)*sin((pi*(q2 + 90))/180) +
72*cos((pi*q5)/180)*(cos((pi*q3)/180)*cos((pi*(q2 +
90))/180) - sin((pi*q3)/180)*sin((pi*(q2 + 90))/180)) -
72*cos((pi*q4)/180)*sin((pi*q5)/180)*(cos((pi*q3)/180)*sin((pi*(q2 +
90))/180) + sin((pi*q3)/180)*cos((pi*(q2 + 90))/180)) + 290]
[

0,

0,

0,

1]
```

Question 5

Plug into the answer for Problem 4 the home/zero joint configuration

```
[0, 0, 0, 0, 0, 0]
```

as shown in the figure below. Show the numeric 4x4 homogeneous transformation matrix representing the 6-DOF pose tip frame with respect to the base frame. Check that the rotation matrix and the x,y,z offset directions match what you would have expected by inspection of the figure below. **ALL UNITS USED ARE IN MM AND DEGREES**



The following function solves this equation while constraining the equation to the joint limits. **ALL UNITS USED ARE IN MM AND DEGREES**

```
vals = [0 0 0 0 0 0].';
solution1 = double(simplify(fk_solve(Transforms, q, vals, 0)));
solution1(:,:,6)
% f(1) = drawManip(solution1, 'Home')
```

```
ans =

     0     0    -1  -374
     0     1     0     0
     1     0     0   630
     0     0     0     1
```

Question 6

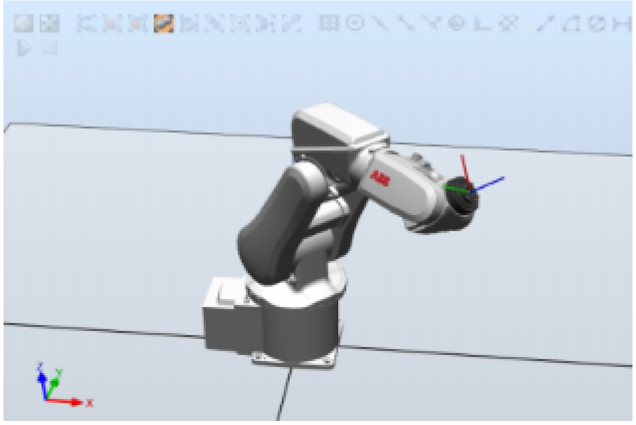
For the following joint configuration

```
[-45, 30, -30, -30, -45, 180],
```

determine the forward kinematics of the end effector (configuration shown below). These values represent the angles with respect to the previous link as we typically use in class.

a)

Determine the numeric 4x4 homogeneous transformation matrix representing the tool's pose (tip frame) with respect to the base frame for this joint configuration.



ALL UNITS USED ARE IN MM AND DEGREES

```
vals = [-45, 30, -30, -30, -45, 180].';
solution2 = double(simplify(fk_solve(Transforms, q, vals, 0)));
solution2(:,:,6)

% b) Clearly state:
%
% # The Cartesian position of the robot tip, including units
%
% *ALL UNITS USED ARE IN MM AND DEGREES*
%
P06 = solution2(1:3,4,6)
%
% # The unit vector representing the approach vector of the robot
% this configuration
%
% *ALL UNITS USED ARE IN MM AND DEGREES*
%
A06 = solution2(1:3,3,6)

% f(2) = drawManip(solution2, 'Rotated Orientation')
```

```
ans =

   -0.2500   -0.6124   -0.7500 -363.0057
    0.7500   -0.6124    0.2500  327.0057
   -0.6124   -0.5000    0.6124  637.9177
         0         0         0    1.0000


P06 =
```

```
 -363.0057
  327.0057
  637.9177


A06 =

   -0.7500
    0.2500
    0.6124
```

Published with MATLAB® R2019a