Robot Morphology

Grup 11 - C

Pol Sancho Bugés

Héctor Suárez Arcos

Link: https://drive.mathworks.com/sharing/8f532fe9-a59a-4ace-b5cd-d70fbe2d9893

Table of Contents

6R Robot. Puma 560

Before start the exercise see the videos:

https://youtu.be/ArzP7rh4_9Q

Call the Wired Robot object and plot it

```
close
clear
mdl_puma560 % Invoque the puma object from the RTB
p560.plot(qz) % qr is the joint vector 1x6. Try qr, qn, any within the limits
```

Work with the wire model and change the point of view.

See: https://es.mathworks.com/help/matlab/creating_plots/setting-the-viewpoint-with-azimuth-and-elevation.html

```
close all
p560.plot(qz)
view([-42.61 46.25])
```

Play with the teach

Modify the joint angle [q1q2 q3 q4 q5 q6]). It is a kind of Joystick.

Pay attention to [x y z].

[ax ay az] are no relevant for the exercise.

```
p560.teach('approach')
```

Moving the Robot

```
clear all
close all
mdl_puma560
```

Declare a joint motion by adding rows

```
Q=zeros(100,6); % at the moment no motion
```

See the Joint 1 limits

```
q1_limits=(p560.links(1, 1).qlim )
```

Build the joint's motion. Firts only Joint #1

```
q1=linspace(q1_limits(1),q1_limits(2),100)';
Q=[q1 Q(:,2:6)]
p560.plot(Q,'jaxes')
```

Play with the plot options

Start getting familiar with options of the RTB tools/functions

Type: help plot \ Other functions named plot \ SerialLink/plot

and visit: robot.pdf file: https://atenea.upc.edu/mod/url/view.php?id=4184725

or at: https://petercorke.com/toolboxes/robotics-toolbox/

And search for: ETS2.plot

Moving two joints. See above

```
q2_limits=p560.links(1, 2).qlim
q2=linspace(q2_limits(1),q2_limits(2),100)';
Q12=[q1 q2 Q(:,3:6)];
```

Options: Add a trail to see the trajectory, display the join axis, make biger or smaller the robot

```
close all
mdl_puma560
```

```
p560.plot(Q12,'trail','--','jaxes','zoom',2) %% Play outside the mlx file to see
it: copy the setence in Command Window
```

Play with other options to get familiar with. You must! becouse all along the course it will be necesary

Recovering End efector position

Use function 'fkine' for recovering the finger tips of the robot.

In next sessionI I will explain and derive this function

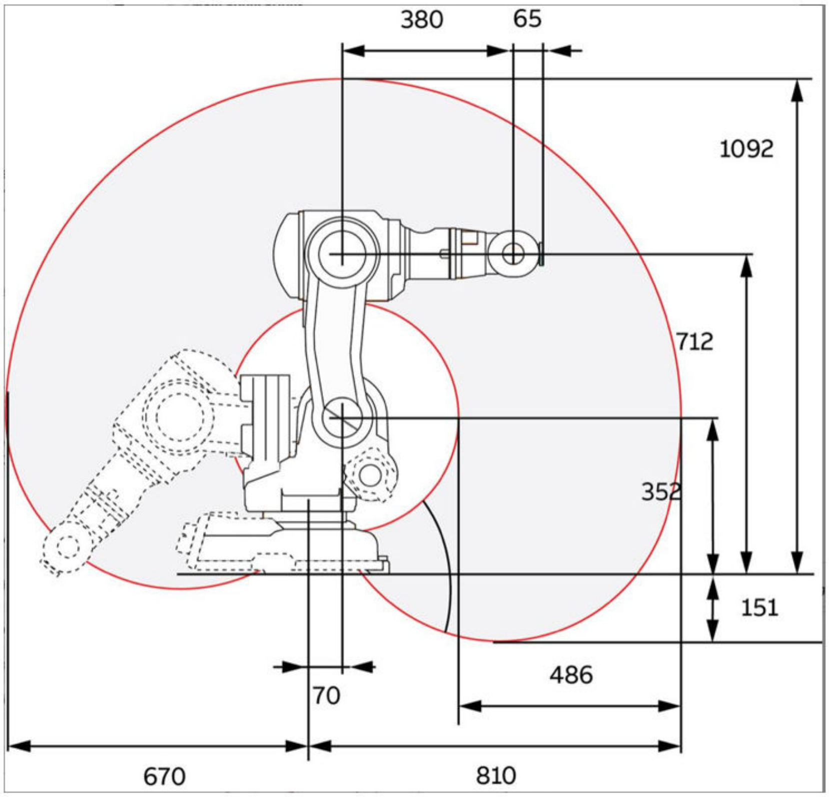
```
T=p560.fkine(Q12); % Forward Kinematics to be explained. Given Theta's (q's) obtain
the robot EE position
ft=[T.t] % to get only the position
figure
plot3(ft(1,:),ft(2,:), ft(3,:))
view(0,40)
```

Working area

```
clear all
close all
mdl_puma560
q2_limits=p560.links(1, 2).qlim
q2=linspace(q2_limits(1),q2_limits(2),100)';
Q= [zeros(100,1) linspace(q2_limits(1),q2_limits(2),100)' zeros(100,4) ]
figure
hold on
v = [-1 -0.1501 -1 ; 1 -0.1501 -1 ; 1 -0.1501 1; -1 -0.1501 1];
f = [1 2 3 4];
patch('Faces',f,'Vertices',v,'FaceColor','blue','FaceAlpha',.3)
title ('Antonio-G11')
p560.plot(Q,'trail','--','jaxes','zoom',2,'movie','2M.mp4')
T=p560.fkine(Q);
ft=[T.t]
```

IRB140 exercise

Get a joint sequence movement to recover the work space as shown in the figure. See video rb140_WS_Solution.mp4.



Deliverable to be uploaded in ATENEA:

1.- Make a movie showing the workspace of the IRB140 ABB robot manipulator.

2.- The link of your Robotica_24_25_1Q shared folder. Keep organize your folders.

Notice:

The title of the figure ought contain the names and group you belong.

Step 1: Invoque IRB140

```
clear
close all
mdl_irb140
```

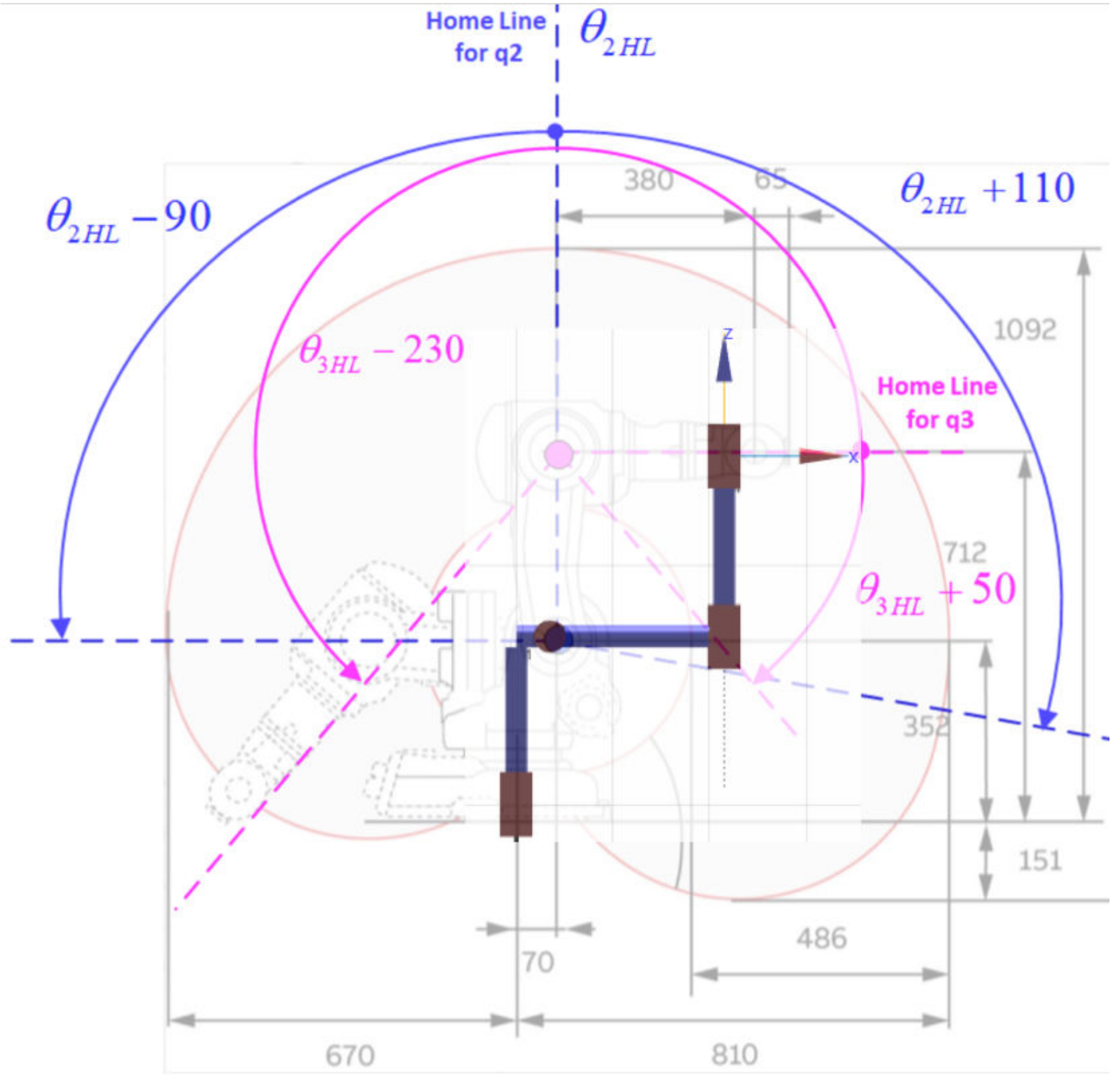
```
robot = 

IRB 140 [ABB]:: 6 axis, RRRRRR, stdDH, slowRNE
+---+-----------+-----------+-----------+-----------+-----------+
| j |     theta |         d |         a |     alpha |    offset |
+---+-----------+-----------+-----------+-----------+-----------+
|  1|         q1|      0.352|       0.07|    -1.5708|          0|
|  2|         q2|          0|       0.36|          0|          0|
|  3|         q3|          0|          0|     1.5708|          0|
|  4|         q4|       0.38|          0|    -1.5708|          0|
|  5|         q5|          0|          0|     1.5708|          0|
|  6|         q6|          0|          0|          0|          0|
+---+-----------+-----------+-----------+-----------+-----------+
```

Step 2: Plot the IRB 140

```
irb140.plot(qd,'zoom',2, 'view',[0 0])
```

Pay attention to home position of the ABB Drawing and the wired model of the RTB

To think about

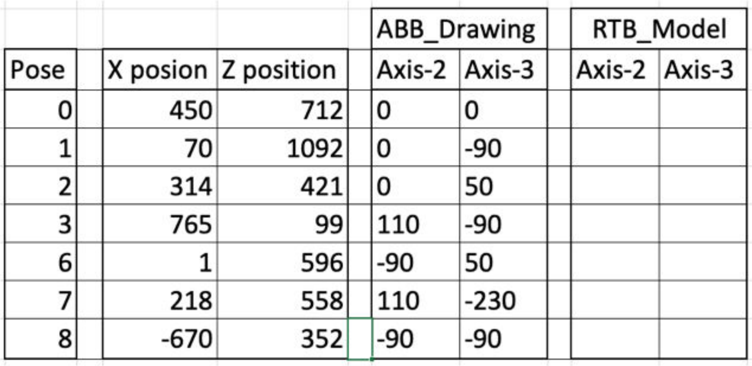


Step 3: Use the teach

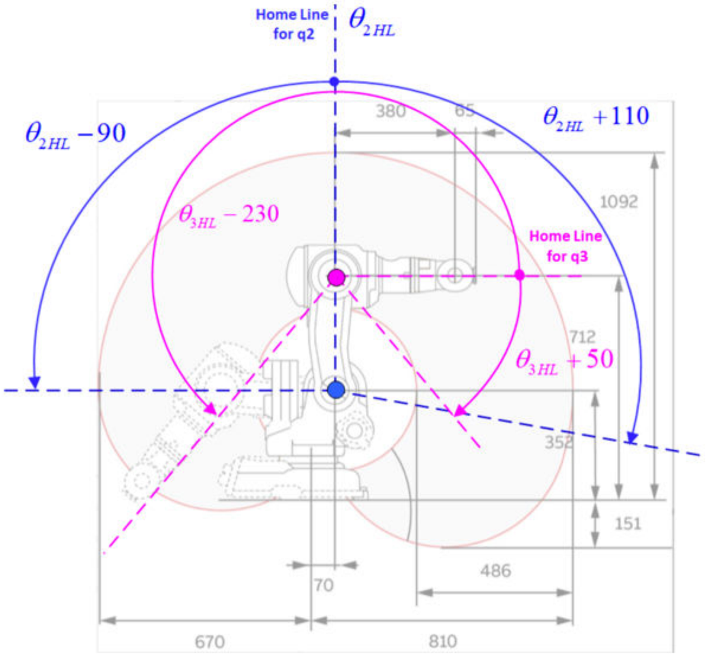
```
irb140.teach('approach')
```

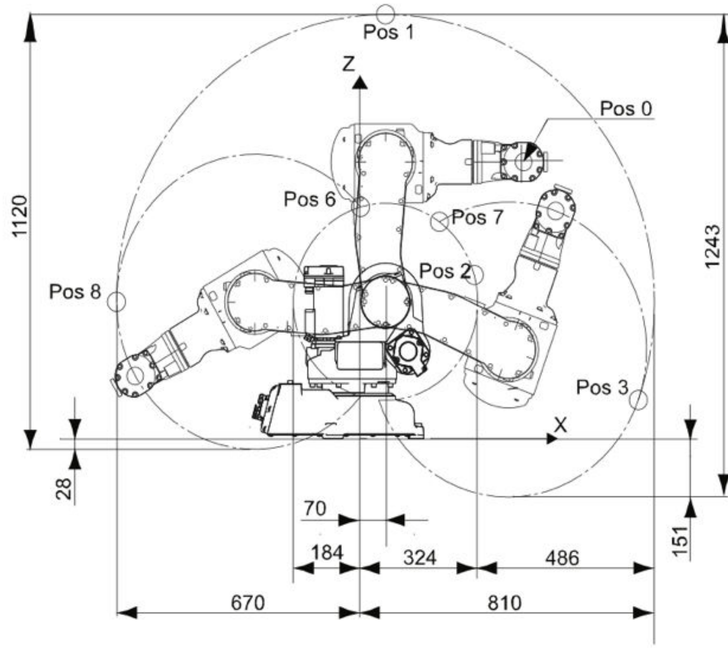
Step 4: Fill the table

Understand the numbers that appears in the following table and fill/create a matrix with the right angle sequence of irb140RTB's and fill the Joint limits for Axis 2 & 3



			ABB_Drawing		RTB_Model	
Pose	X posion	Z position	Axis-2	Axis-3	Axis-2	Axis-3
0	450	712	0	0		
1	70	1092	0	-90		
2	314	421	0	50		
3	765	99	110	-90		
6	1	596	-90	50		
7	218	558	110	-230		
8	-670	352	-90	-90		





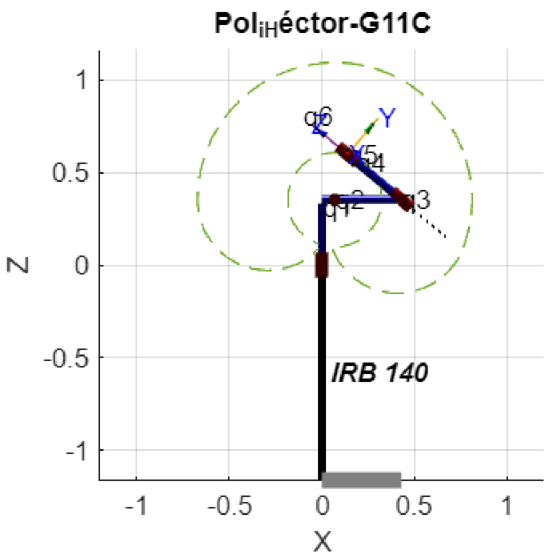
```
Q = [0   0    0 0 0 0; ...
     0   0  -90 0 0 0; ...
     0   0   50 0 0 0; ...
     0 110  -90 0 0 0; ...
     0 -90   50 0 0 0; ...
     0 110 -230 0 0 0; ...
     0 90   -90 0 0 0  ...
];

Q = rad2deg(qd) + Q;
```

Finding the workspace of irb140

```
Q_0 = [                 ...
    0 -90   50 0 0 0; ...
    0 110   50 0 0 0; ...
    0 110  -90 0 0 0; ...
    0 -90  -90 0 0 0; ...
    0 -90 -230 0 0 0; ...
    0  90 -230 0 0 0];

Q_0 = deg2rad(Q_0) + qd;
[n, m] = size(Q_0);

k = 100;

Q = zeros(k * (n - 1), m);
for i = 1 : (n - 1)
    for j = 1 : m
    row = (i - 1) * k + 1: i * k;
    Q(row, j) = linspace(Q_0(i, j), Q_0(i + 1, j), k)';
    end
end
```

```
title('Pol_i_Héctor-G11C')
irb140.plot(Q, 'trail', '--', 'jaxes', 'zoom', 2, 'view', [0
0],'movie','Entregable.mp4');
```

```
Animate: saving video --> Entregable.mp4 with profile 'MPEG-4'
```


$\text{Pol}_{\text{iH}}\text{éctor-G11C}$
q6
Z
Y
X
q5
q4
q3
q2
q1
IRB 140
Z
X