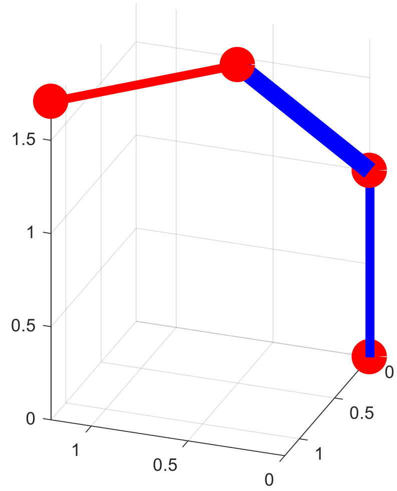
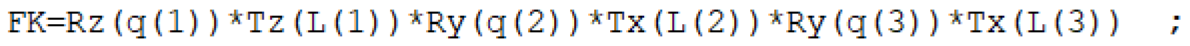
Jose Corona

Home Task 4

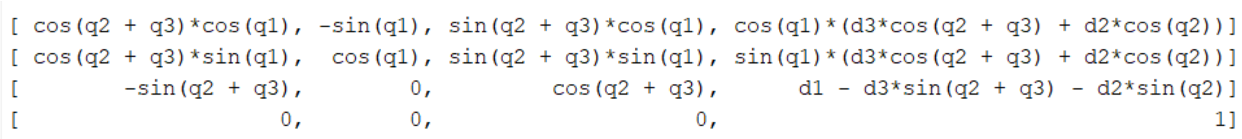
To run the file with name “Program.m “ in MATLAB.

1. Calculate Jacobian (numeric method)



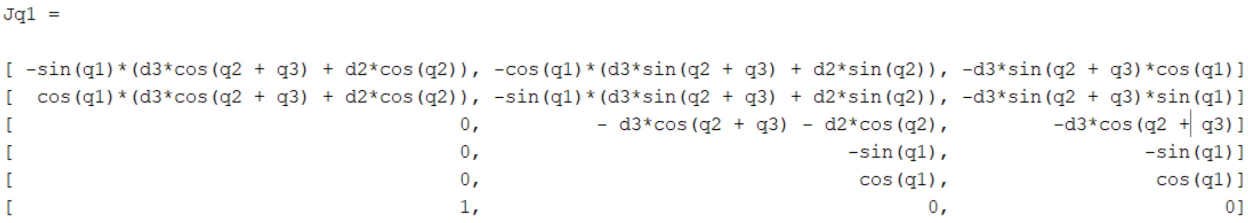


```
FK=Rz(q(1))*Tz(L(1))*Ry(q(2))*Tx(L(2))*Ry(q(3))*Tx(L(3))   ;
```



```
[ cos(q2 + q3)*cos(q1), -sin(q1), sin(q2 + q3)*cos(q1), cos(q1)*(d3*cos(q2 + q3) + d2*cos(q2))]
[ cos(q2 + q3)*sin(q1),  cos(q1), sin(q2 + q3)*sin(q1), sin(q1)*(d3*cos(q2 + q3) + d2*cos(q2))]
[        -sin(q2 + q3),        0,         cos(q2 + q3),      d1 - d3*sin(q2 + q3) - d2*sin(q2)]
[                    0,        0,                    0,                                      1]
```

I used the numeric method to get the jacobian.



```
Jq1 =

[ -sin(q1)*(d3*cos(q2 + q3) + d2*cos(q2)), -cos(q1)*(d3*sin(q2 + q3) + d2*sin(q2)), -d3*sin(q2 + q3)*cos(q1)]
[  cos(q1)*(d3*cos(q2 + q3) + d2*cos(q2)), -sin(q1)*(d3*sin(q2 + q3) + d2*sin(q2)), -d3*sin(q2 + q3)*sin(q1)]
[                                       0,          - d3*cos(q2 + q3) - d2*cos(q2),         -d3*cos(q2 + q3)]
[                                       0,                                -sin(q1),                 -sin(q1)]
[                                       0,                                 cos(q1),                  cos(q1)]
[                                       1,                                       0,                        0]
```

2. Joint trajectory q(t) from q(0) = (0, 0, 0) to q(2) = (2, 3, 4) with null initial and final velocities and accelerations. (polynomial)

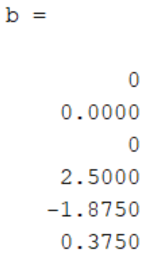
A polynomial solution for each joint is solved as like in presentation, since it has 6 constrains, is needed a polynomial of fifth solution.

$$\theta(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5$$

$$\begin{bmatrix} \theta_0 \\ \omega_0 \\ \alpha_0 \\ \theta_f \\ \omega_f \\ \alpha_f \end{bmatrix} = \begin{bmatrix} 1 & t_0 & t_0^2 & t_0^3 & t_0^4 & t_0^5 \\ 0 & 1 & 2t_0 & 3t_0^2 & 4t_0^3 & 5t_0^4 \\ 0 & 0 & 2 & 6t_0 & 12t_0^2 & 20t_0^3 \\ 1 & t_f & t_f^2 & t_f^3 & t_f^4 & t_f^5 \\ 0 & 1 & 2t_f & 3t_f^2 & 4t_f^3 & 5t_f^4 \\ 0 & 0 & 2 & 6t_f & 12t_f^2 & 20t_f^3 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{bmatrix}$$

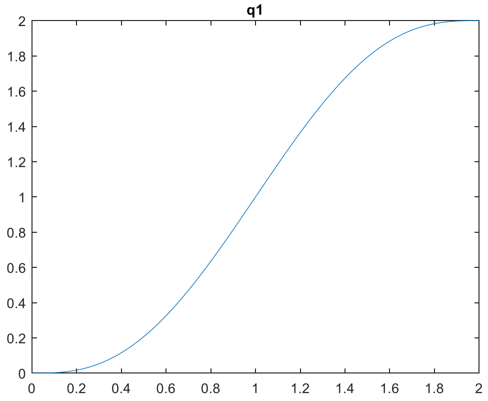
For the joint 1:

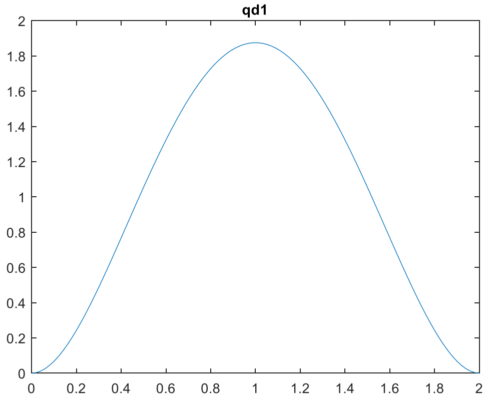
So the coefficients for the polynomial are:

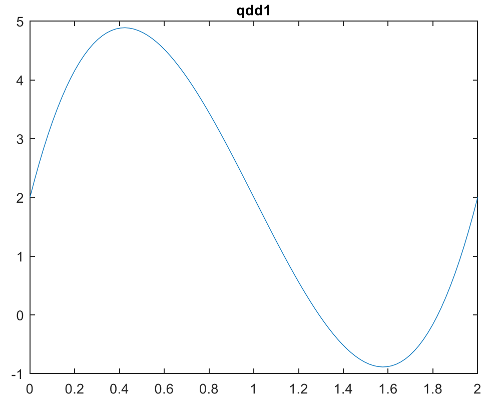


```
b =

         0
    0.0000
         0
    2.5000
   -1.8750
    0.3750
```

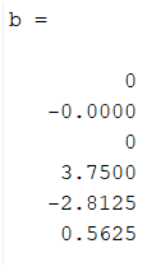
q1= (3*t^5)/8 - (15*t^4)/8 + (5*t^3)/2 + t/562949953421312





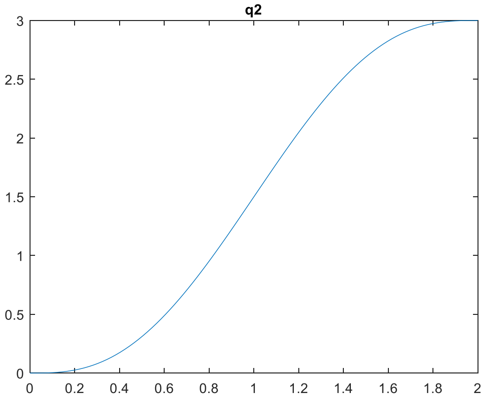


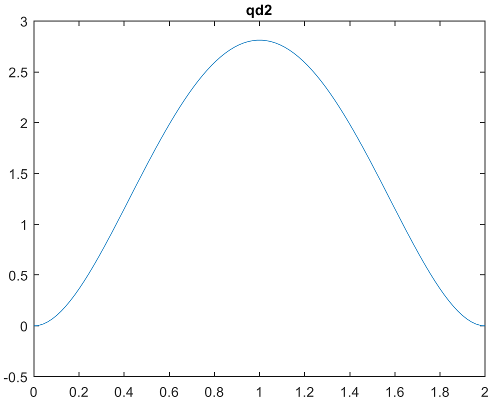
For the joint 2:

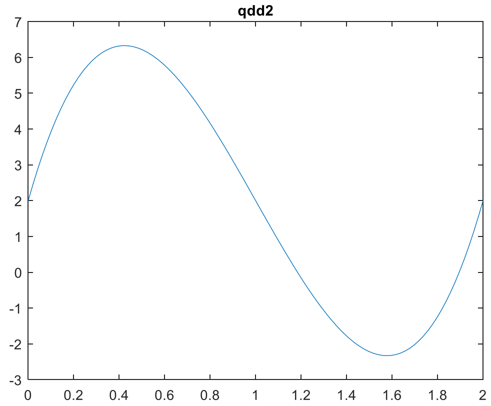


```
b =


         0
   -0.0000
         0
    3.7500
   -2.8125
    0.5625
```

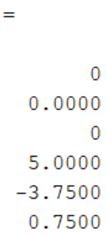
q2= (9*t^5)/16 - (45*t^4)/16 + (15*t^3)/4 - t/562949953421312





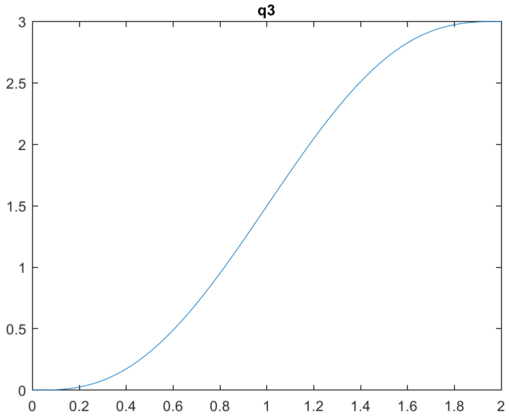


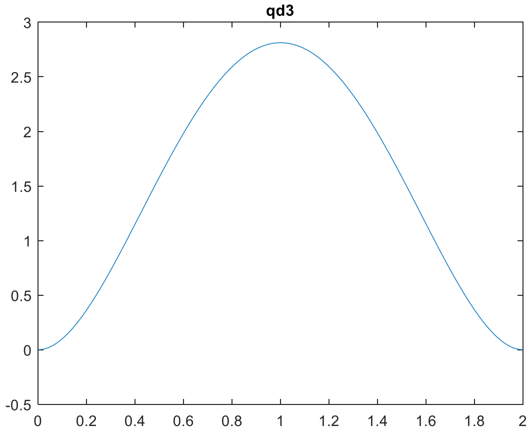
For the joint 3:

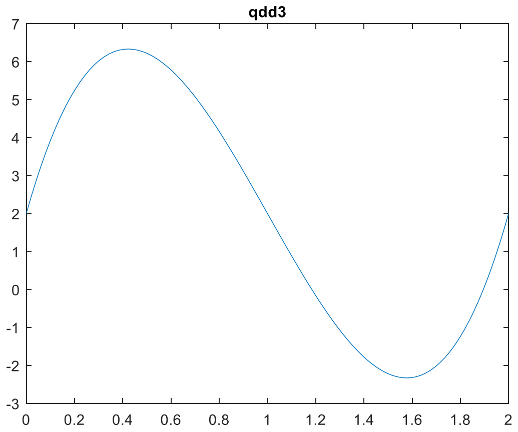


```
b =

         0
    0.0000
         0
    5.0000
   -3.7500
    0.7500
```

q3= (3*t^5)/4 - (15*t^4)/4 + 5*t^3 + t/281474976710656



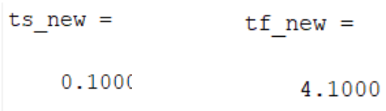




3. Joint trajectory for the following commands: PTP – q1 = (0, 0, 0) to q2 = (2, 3, 4) (trapezoidal)

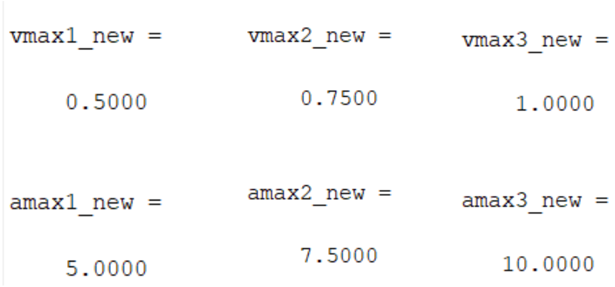
Fist we have to calculate the times “ts” and “tf” for all trapezoidal velocity for each joints, taking into account the controller command interpretation frequency of 10hz.

So, the values of “ts” and “tf” for all the joins is :



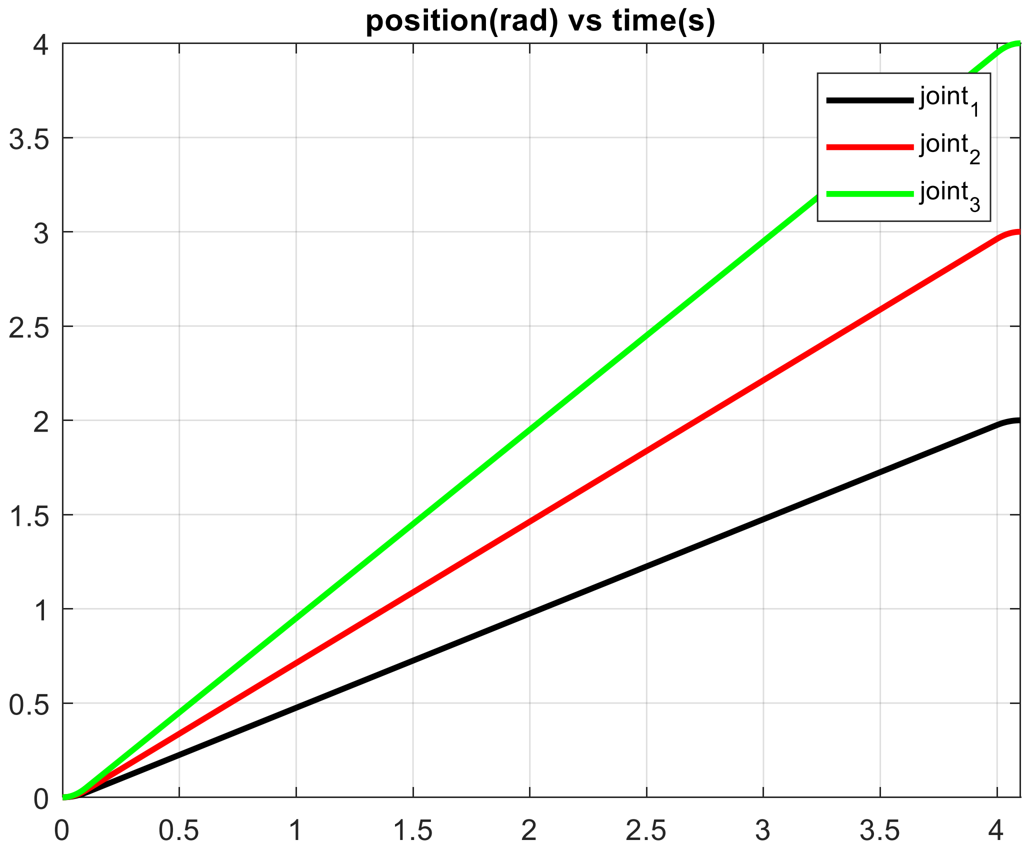
```
ts_new =            tf_new =

    0.100(              4.1000
```

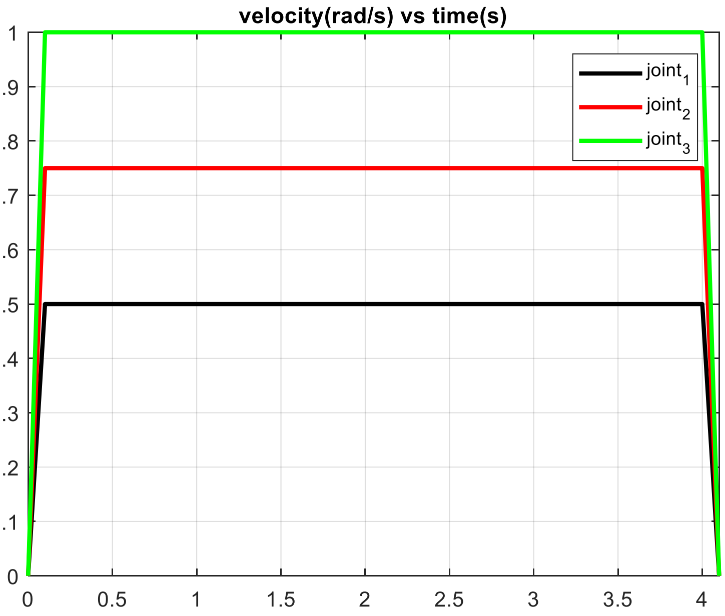
And the new values of max velocity and acceleration for each joint are:

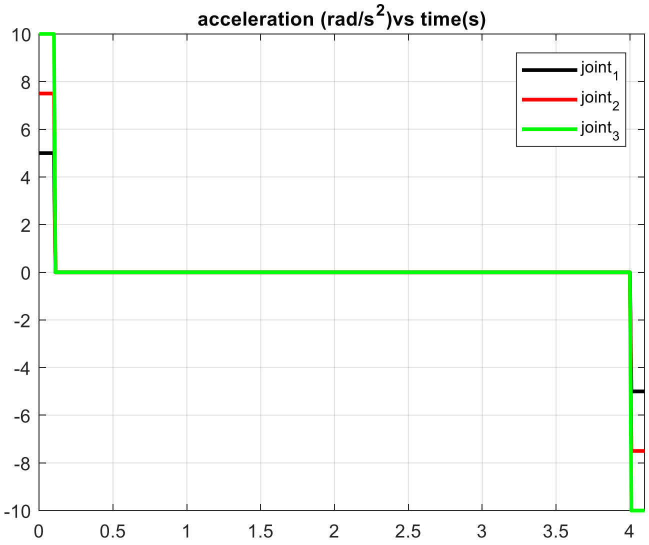


```
vmax1_new =        vmax2_new =        vmax3_new =

    0.5000             0.7500             1.0000


amax1_new =        amax2_new =        amax3_new =

    5.0000             7.5000            10.0000
```

And the plot of the position, velocity and aceleration are:

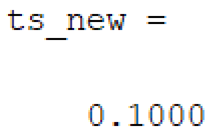

position(rad) vs time(s)
0
0.5
1
1.5
2
2.5
3
3.5
4
0
0.5
1
1.5
2
2.5
3
3.5
4
joint$_1$
joint$_2$
joint$_3$


velocity(rad/s) vs time(s)
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
0.5
1
1.5
2
2.5
3
3.5
4
joint$_1$
joint$_2$
joint$_3$

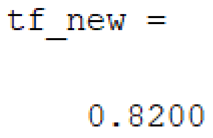


4. Joint trajectory for the following commands: LIN – p1 = (1, 0, 1) to p2 = (√2/2, √2/2, 1.2) (trapezoidal)

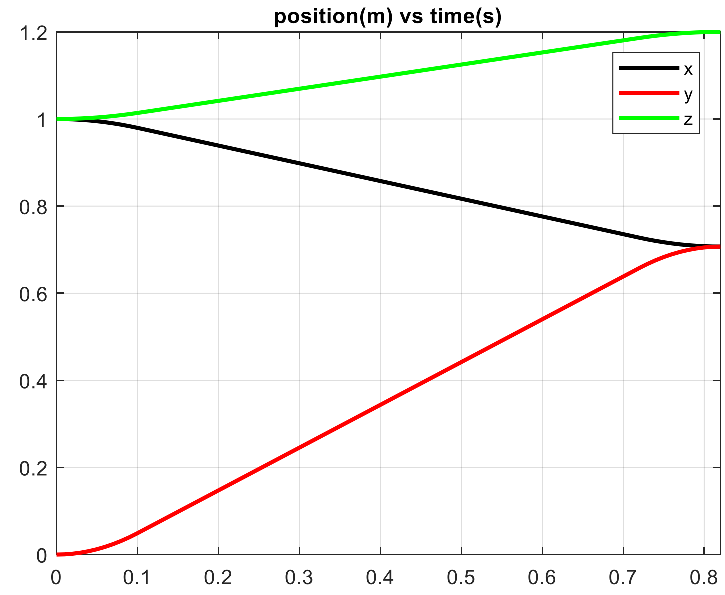
4a. Trapezoidal trajectory in task space (without constrains in joints space)

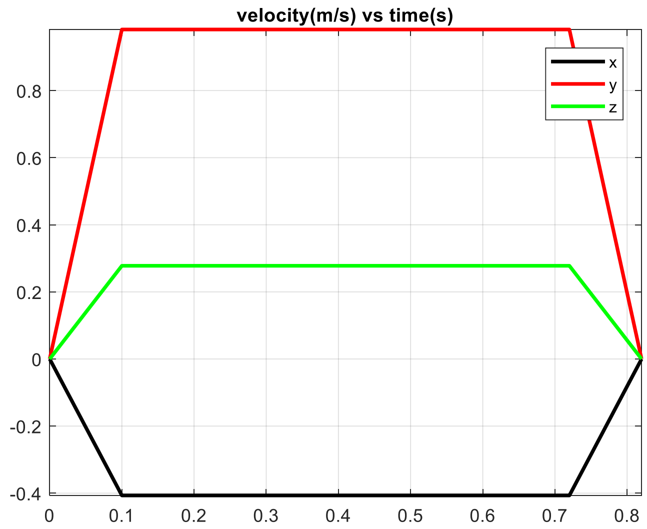


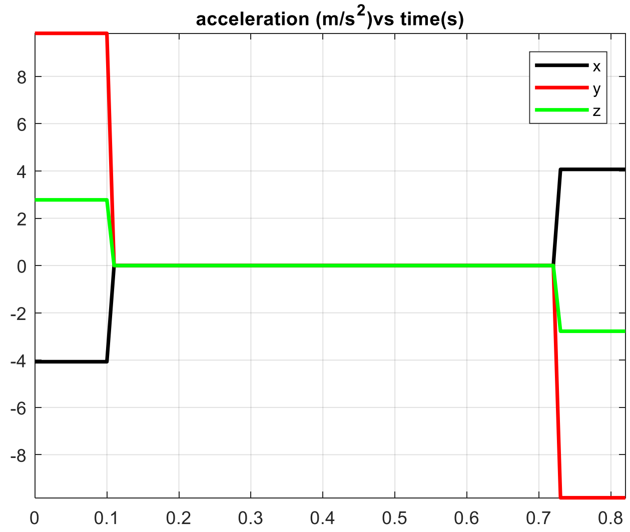
```
ts_new =

    0.1000
```



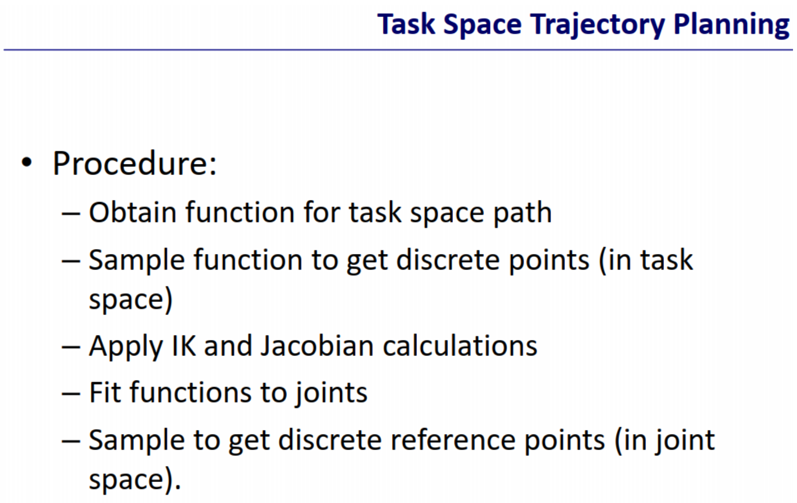
```
tf_new =

    0.8200
```


position(m) vs time(s)


velocity(m/s) vs time(s)



To get the joints position, velocity and acceleration, since que have the formulas for velocity in x,y,z for each section of the trajectory, we have to multiply it with the inverse of the Jacobean, to get the function of velocity in the joints. Then we could try derivate it to get the function of acceleration in the joints.

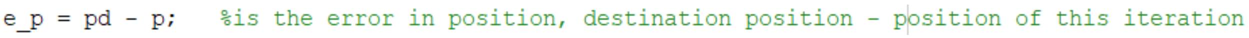


Task Space Trajectory Planning

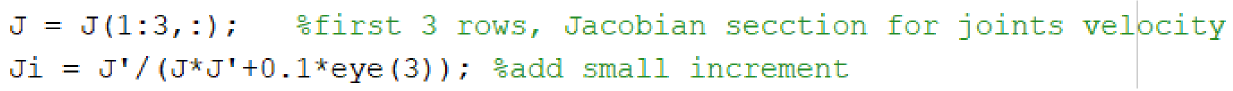
- Procedure:
  - Obtain function for task space path
  - Sample function to get discrete points (in task space)
  - Apply IK and Jacobian calculations
  - Fit functions to joints
  - Sample to get discrete reference points (in joint space).

4b. Trapezoidal trajectory in task space and constrains in joints space too.

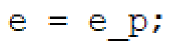
Fist we have to calculate the joints position for p1 and p2, so we use the inverse kinematics. I develop a function for IK (file named “IK.m”).



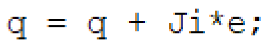
```
e_p = pd - p;   %is the error in position, destination position - position of this iteration
```



```
J = J(1:3,:);    %first 3 rows, Jacobian secction for joints velocity
Ji = J'/(J*J'+0.1*eye(3)); %add small increment
```

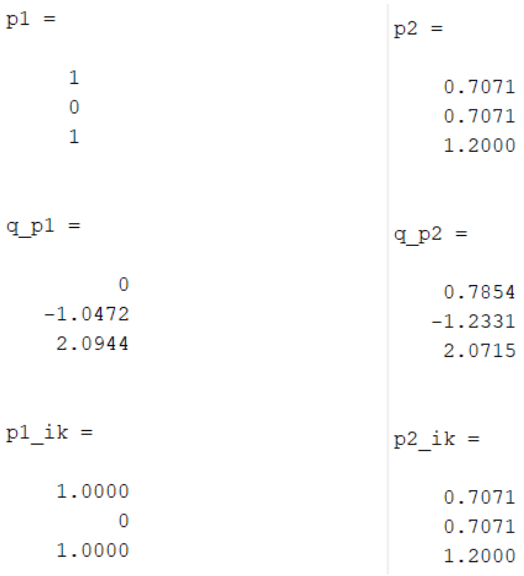


```
e = e_p;
```



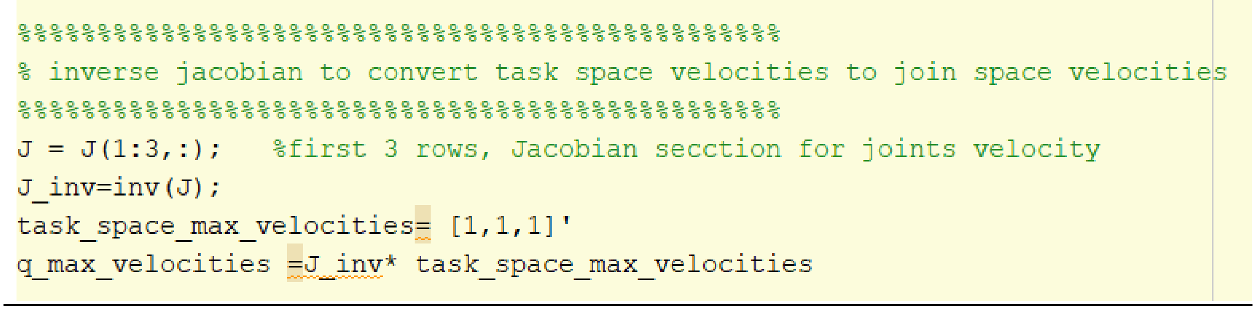
```
q = q + Ji*e;
```

So, for point p1 and p2, I get the joins position and checked they were correct, with my forward kinematics:



```
p1 =                    p2 =

     1                      0.7071
     0                      0.7071
     1                      1.2000


q_p1 =                  q_p2 =

         0                  0.7854
   -1.0472                 -1.2331
    2.0944                  2.0715


p1_ik =                 p2_ik =

    1.0000                  0.7071
         0                  0.7071
    1.0000                  1.2000
```

Then,



```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% inverse jacobian to convert task space velocities to join space velocities
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
J = J(1:3,:);   %first 3 rows, Jacobian secction for joints velocity
J_inv=inv(J);
task_space_max_velocities= [1,1,1]'
q_max_velocities =J_inv* task_space_max_velocities
```

Link Github:

https://github.com/Jose-R-Corona/HomeTaks4