Sumeet Shanbhag

Robot Manipulator form is given by

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \tau$$

Consider a cubic polynomial for Joint 1.

$$q(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3$$

$$\dot{q}(t) = a_1 + 2 a_2 t' + 3 a_3 t^2$$

Repeating the same steps for joint, we can get the following form with initial & final conditions t_0 & t_f, Combining the equations

$$\underbrace{\begin{bmatrix} 1 & t_0 & t_0^2 & t_0^3 \\ 0 & 1 & 2t_0 & 3t_0^2 \\ 1 & t_f & t_f^2 & t_f^3 \\ 0 & 1 & 2t_f & 3t_f^2 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}}_{a} = \underbrace{\begin{bmatrix} q_0 \\ \dot{q}_0 \\ q_f \\ \dot{q}_f \end{bmatrix}}_{B}$$

After putting in MATLAB, we can derive a using

a = inv(A) * B

Lets define a virtual input v and choose control input τ according to

$$\tau = M(q)\,v(t) + C(q,\dot{q})\dot{q} + g(q)$$

Results in n decoupled linear systems :-

$$\ddot{q} = v(t) \quad , \quad v = -[\underset{n\times n}{K_p} \quad \underset{n\times n}{K_d}]\, x$$

$$v = -K_p(q - q_d) - K_d(\dot{q} - \dot{q}_d) + \ddot{q}_d \ , \ K_p > 0 \ , \ K_d > 0$$

Putting this in the control law, we get

$$\tau = M(q)[-K_p(q - q_d) - K_d(\dot{q} - \dot{q}_d) + \ddot{q}_d] + C(q,\dot{q})\dot{q} + G(q)$$

Now this theory will be used in MATLAB

Sumeet Shanbhag

Robot Control Assignment 3 Report

a) Trajectory Generation:

Cubic Polynomial

A comprises of cubic polynomials. And B1 and B2 comprises the angles and their derivatives. J1 and J2 is the coefficient of the cubic polynomial which are used to generate a trajectory of the joints.

```
J1 = 4×1            J2 = 4×1
      3.1416              1.5708
           0                   0
     -0.0942             -0.0471
      0.0063              0.0031
```

Joint1 =

$$\pi - \frac{3\,\pi\,t^2}{100} + \frac{\pi\,t^3}{500}$$

Joint2 =

$$\frac{\pi}{2} - \frac{3\,\pi\,t^2}{200} + \frac{\pi\,t^3}{1000}$$

vel1 =

$$-\frac{3\,\pi\,t}{50} + \frac{3\,\pi\,t^2}{500}$$

vel2 =

$$-\frac{3\,\pi\,t}{100} + \frac{3\,\pi\,t^2}{1000}$$

acc1 =

$$-\frac{3\,\pi}{50} + \frac{3\,\pi\,t}{250}$$

acc2 =

$$-\frac{3\,\pi}{100} + \frac{3\,\pi\,t}{500}$$

b) **Manipulator equation**

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \tau$$

The equations of motion from programming assignment 1 are

T1 = theta1_ddot*(I1 + I2 + (m1*(2*r1^2*cos(theta1)^2 + 2*r1^2*sin(theta1)^2))/2 + (m2*(2*(r2*cos(theta2 + theta1) + l1*cos(theta1))^2 + 2*(r2*sin(theta2 + theta1) + l1*sin(theta1))^2))/2) + theta2_ddot*(I2 + (m2*(2*r2*sin(theta2 + theta1)*(r2*sin(theta2 + theta1) + l1*sin(theta1)) + 2*r2*cos(theta2 + theta1)*(r2*cos(theta2 + theta1) + l1*cos(theta1))))/2) + (m2*theta2_dot*(2*(cos(theta1 + theta2)*r2*theta2_dot + cos(theta1 + theta2)*r2*theta1_dot)*(r2*sin(theta2 + theta1) + l1*sin(theta1)) - 2*(sin(theta1 + theta2)*r2*theta2_dot + sin(theta1 + theta2)*r2*theta1_dot)*(r2*cos(theta2 + theta1) +

l1*cos(theta1)) - 2*r2*sin(theta2 + theta1)*(theta1_dot*(r2*cos(theta2 + theta1) + l1*cos(theta1)) + r2*theta2_dot*cos(theta2 + theta1)) + 2*r2*cos(theta2 + theta1)*(theta1_dot*(r2*sin(theta2 + theta1) + l1*sin(theta1)) + r2*theta2_dot*sin(theta2 + theta1))))/2 - g*m2*(r2*sin(theta2 + theta1) + l1*sin(theta1)) - g*m1*r1*sin(theta1);

T2 = (m2*(2*(sin(theta1 + theta2)*r2*theta2_dot + sin(theta1 + theta2)*r2*theta1_dot)*(theta1_dot*(r2*cos(theta2 + theta1) + l1*cos(theta1)) + r2*theta2_dot*cos(theta2 + theta1)) - 2*(cos(theta1 + theta2)*r2*theta2_dot + cos(theta1 + theta2)*r2*theta1_dot)*(theta1_dot*(r2*sin(theta2 + theta1) + l1*sin(theta1)) + r2*theta2_dot*sin(theta2 + theta1))))/2 + theta2_ddot*(I2 + (m2*(2*r2^2*cos(theta2 + theta1)^2 + 2*r2^2*sin(theta2 + theta1)^2))/2) + theta1_ddot*(I2 + (m2*(2*r2*sin(theta2 + theta1)*(r2*sin(theta2 + theta1) + l1*sin(theta1)) + 2*r2*cos(theta2 + theta1)*(r2*cos(theta2 + theta1) + l1*cos(theta1))))/2) - (m2*theta2_dot*(2*r2*sin(theta2 + theta1)*(theta1_dot*(r2*cos(theta2 + theta1) + l1*cos(theta1)) + r2*theta2_dot*cos(theta2 + theta1)) - 2*r2*cos(theta2 + theta1)*(theta1_dot*(r2*sin(theta2 + theta1) + l1*sin(theta1)) + r2*theta2_dot*sin(theta2 + theta1)) - 2*r2*sin(theta2 + theta1)*(cos(theta1 + theta2)*r2*theta2_dot + cos(theta1 + theta2)*r2*theta1_dot) + 2*r2*cos(theta2 + theta1)*(sin(theta1 + theta2)*r2*theta2_dot + sin(theta1 + theta2)*r2*theta1_dot)))/2 - g*m2*r2*sin(theta2 + theta1);

G =

[- (8829*sin(theta2 + theta1))/2000 - (28449*sin(theta1))/2000, -(8829*sin(theta2 + theta1))/2000]

M =

[(81*cos(theta1)^2)/400 + (81*sin(theta1)^2)/400 + ((9*cos(theta2 + theta1))/20 + cos(theta1))^2 + ((9*sin(theta2 + theta1))/20 + sin(theta1))^2 + sym(21/125), (9*cos(theta2 + theta1)*((9*cos(theta2 + theta1))/20 + cos(theta1)))/20 + (9*sin(theta2 + theta1)*((9*sin(theta2 + theta1))/20 + sin(theta1)))/20 + sym(21/250); (9*cos(theta2 + theta1)*((9*cos(theta2 + theta1))/20 + cos(theta1)))/20 + (9*sin(theta2 + theta1)*((9*sin(theta2 + theta1))/20 + sin(theta1)))/20 + sym(21/250), (81*cos(theta2 + theta1)^2)/400 + (81*sin(theta2 + theta1)^2)/400 + sym(21/250)]

C =

[-(theta2_dot*((9*sin(theta2 + theta1)*((9*cos(theta2 + theta1)*theta2_dot)/20 + ((9*cos(theta2 + theta1))/20 + cos(theta1))*theta1_dot))/10 + ((9*cos(theta2 + theta1))/20 + cos(theta1))*((9*sin(theta2 + theta1)*theta2_dot)/10 + (9*sin(theta2 + theta1)*theta1_dot)/10) - ((9*cos(theta2 + theta1)*theta2_dot)/10 + (9*cos(theta2 + theta1)*theta1_dot)/10)*((9*sin(theta2 + theta1))/20 + sin(theta1)) - (9*cos(theta2 + theta1)*((9*sin(theta2 + theta1)*theta2_dot)/20 + ((9*sin(theta2 + theta1))/20 + sin(theta1))*theta1_dot))/10))/2; (((9*sin(theta2 + theta1)*theta2_dot)/10 + (9*sin(theta2 + theta1)*theta1_dot)/10)*((9*cos(theta2 + theta1)*theta2_dot)/20 + ((9*cos(theta2 + theta1))/20 + cos(theta1))*theta1_dot))/2 - (theta2_dot*((9*sin(theta2 + theta1)*((9*cos(theta2 + theta1)*theta2_dot)/20 + ((9*cos(theta2 + theta1))/20 + cos(theta1))*theta1_dot))/10 - (9*cos(theta2 + theta1)*((9*sin(theta2 +

theta1)*theta2_dot)/20 + ((9*sin(theta2 + theta1))/20 + sin(theta1))*theta1_dot))/10 - (9*sin(theta2 + theta1)*((9*cos(theta2 + theta1)*theta2_dot)/20 + (9*cos(theta2 + theta1)*theta1_dot)/20))/10 + (9*cos(theta2 + theta1)*((9*sin(theta2 + theta1)*theta2_dot)/20 + (9*sin(theta2 + theta1)*theta1_dot)/20))/10))/2 - (((9*sin(theta2 + theta1)*theta2_dot)/20 + ((9*sin(theta2 + theta1))/20 + sin(theta1))*theta1_dot)*((9*cos(theta2 + theta1)*theta2_dot)/10 + (9*cos(theta2 + theta1)*theta1_dot)/10))/2]

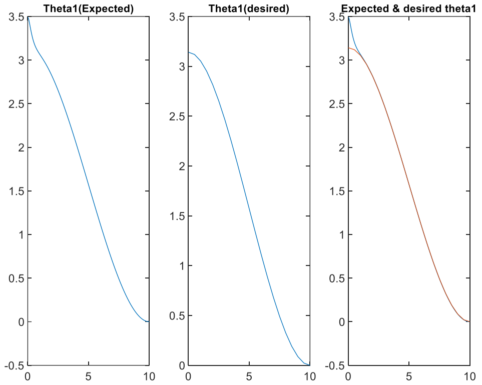
c)

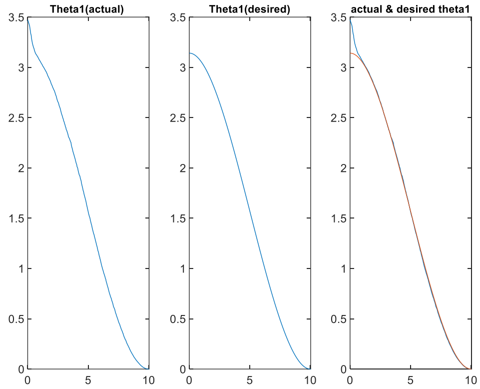
```
A = 4x4

     0     0     1     0
     0     0     0     1
     0     0     0     0
     0     0     0     0

B = 4x2

     0     0
     0     0
     1     0
     0     1

lamdas = 1x4

   -10    -8    -7    -6

K = 2x4

   42.0000         0   13.0000         0
         0   80.0000         0   18.0000
```

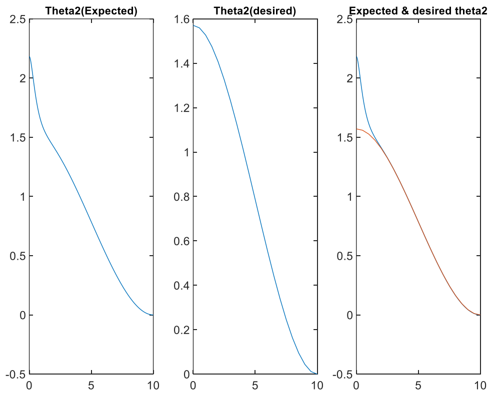
v = - (K * ([theta1 ; theta2; theta1_dot1 ;theta2_dot1] - [Joint1 ; Joint2 ; Vel1; Vel2])) + total_acc;

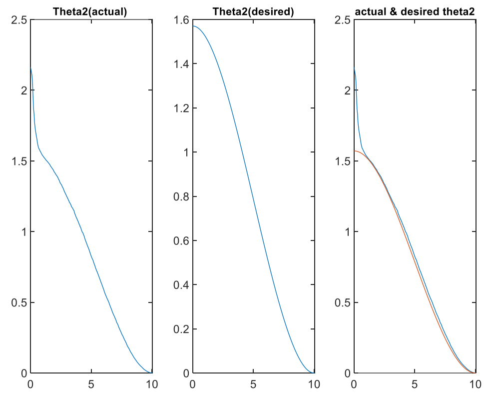
Control Law = M* v + C + G

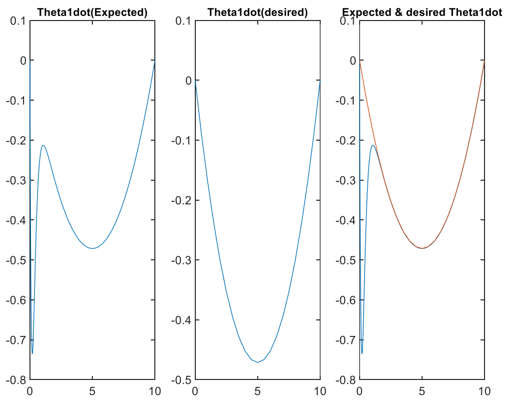
MATLAB plots vs ROS plots

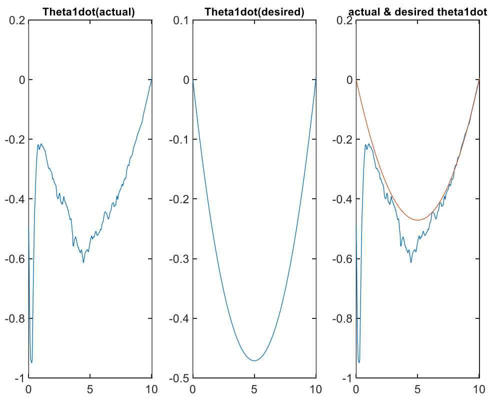




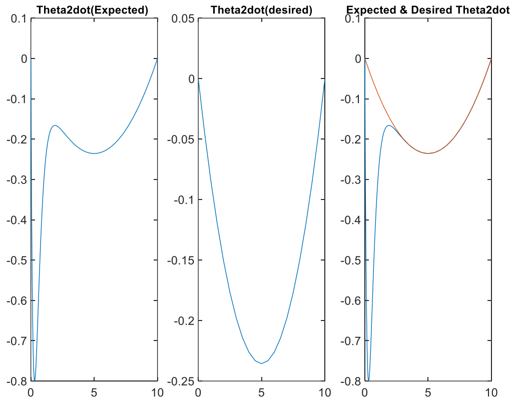


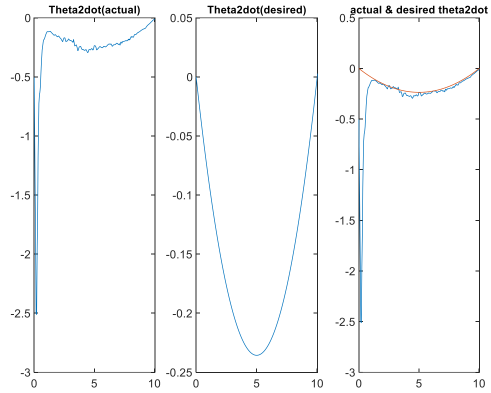


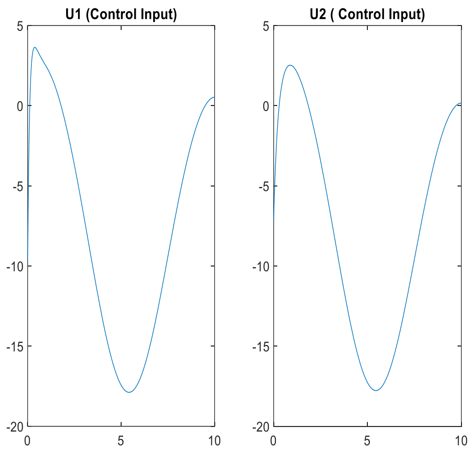


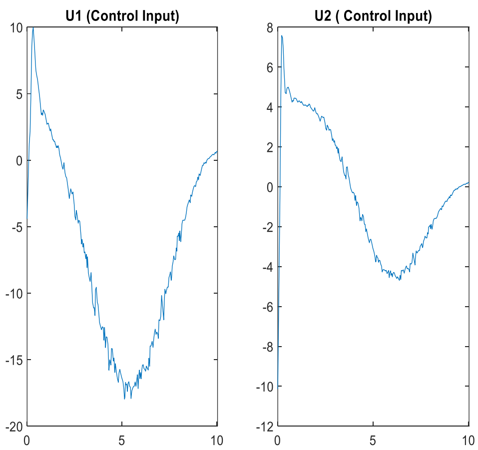


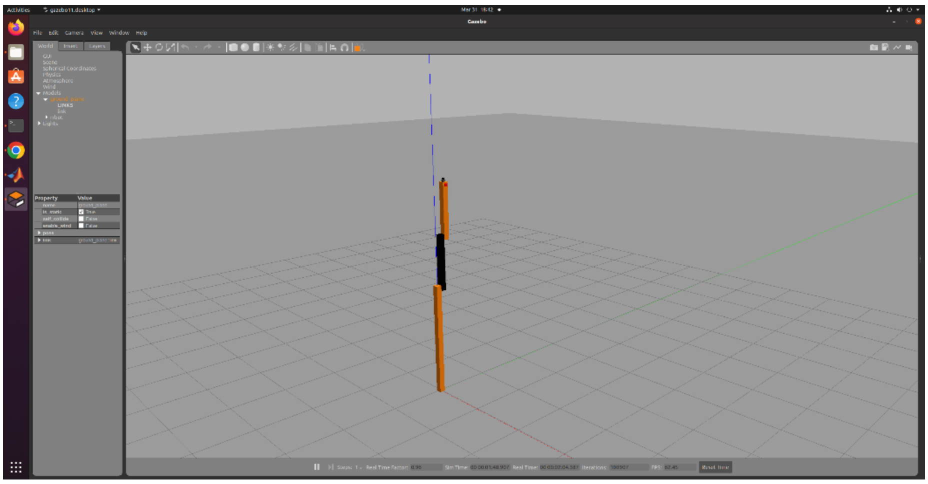
MATLAB plots vs ROS plots











Due to gazebo having physics applied to the robot, parameters such as friction and other disturbances affect the graph of the systems. These parameters are however absent in MATLAB which leads to a smoother plot. Hence, we can see that there are a lot of oscillations and irregularities in Gazebo and not in MATLAB.