Programming Assignment 2

a) The Equilibrium Points

By equating the state-space function to zero, we get theta1_dot and theta2_dot to be zeros.
We use the EOM and *subs*titute for m1, m2, l1, l2, r1, r2, I1, I2, g, theta1_dot, theta2_dot, theta1_ddot, theta2_ddot.
The substituted equations are *solve*d for theta1 and theta 2 to get the following equilibrium points.
[Theta1, Theta2] = [0,0], [0, pi], [pi, 0], [pi, pi]

b) Jacobian Linearization

Since the equation is non-linear we linearize it around the equilibrium points. For this, we first find the Jacobians, A and B matrices.
A = jacobian(X_dot, X);
B = jacobian(X_dot, u);

c) Stability

The Eigen values obtained at the different Equilibrium points are:
eigen_1 = 7.1676, 2.7129, -7.1676, -2.7129
eigen_2 = -3.8995 + 0.0000i, 3.8995 + 0.0000i, 0.0000 + 4.9864i, 0.0000 - 4.9864i
eigen_3 = -0.0000 + 7.1676i, -0.0000 - 7.1676i, 0.0000 + 2.7129i, 0.0000 - 2.7129i
eigen_4 = -4.9864 + 0.0000i, 0.0000 + 3.8995i, 0.0000 - 3.8995i, 4.9864 + 0.0000i
We can see that only at **(pi, 0)** the system is **marginally stable** as the real parts of the eigen values are zero.

d) The Controllability

For upward position, **theta1 = theta2 = theta1_dot = theta2_dot = 0**
The controllability matrix:

ctrb(A1,B1)

The rank of the controllability matrix is ***4***.

r = rank(ctrb(A1,B1));

The controllability matrix has full rank and hence the system is controllable at the upward position.

e) State-Feedback Control

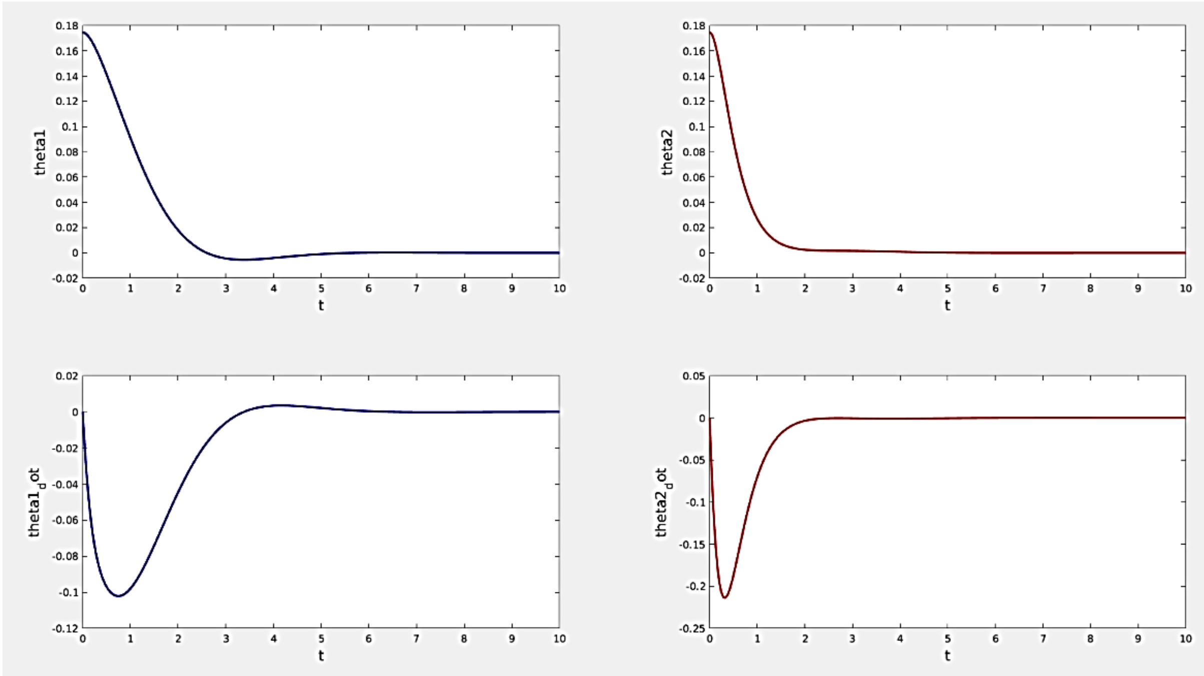
The eigen values are set to **[-4, -2, -1+i, -1-i]**
The gain matrix obtained,

K = [24.7907, 12.7142, 5.7498, 6.0241; 6.3156, 7.4114, 1.7584, 2.1889]

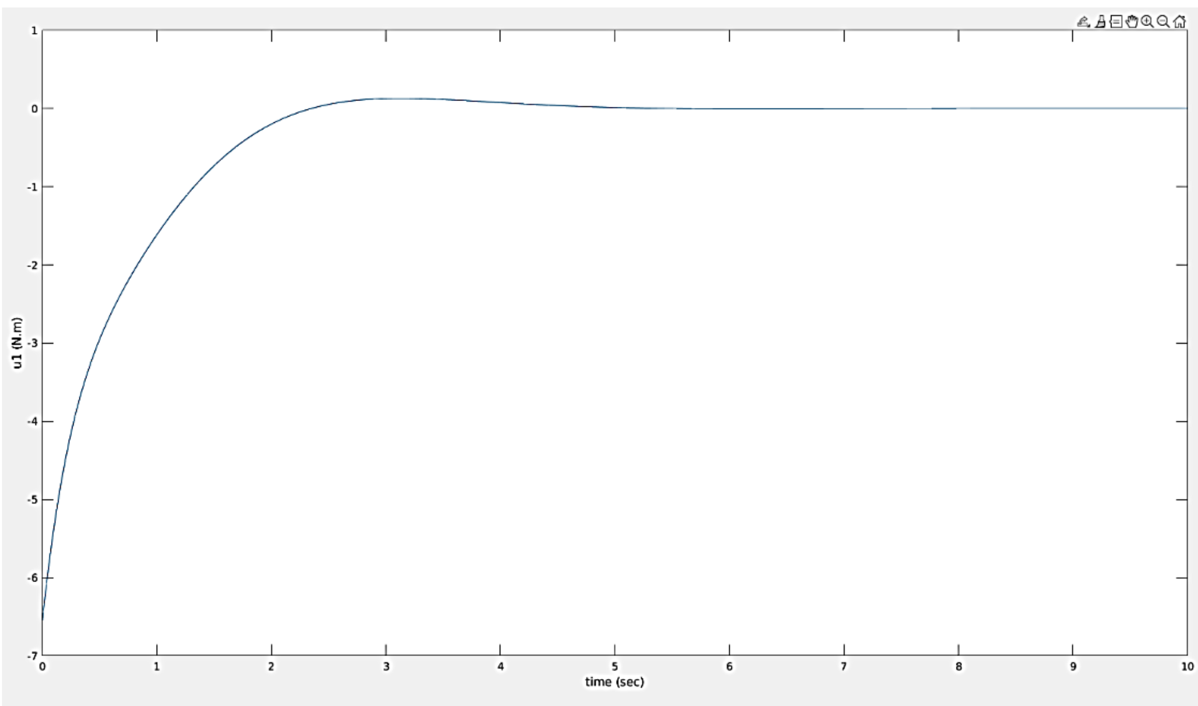
f) By substituting for the efforts in ode function as

U = -kX

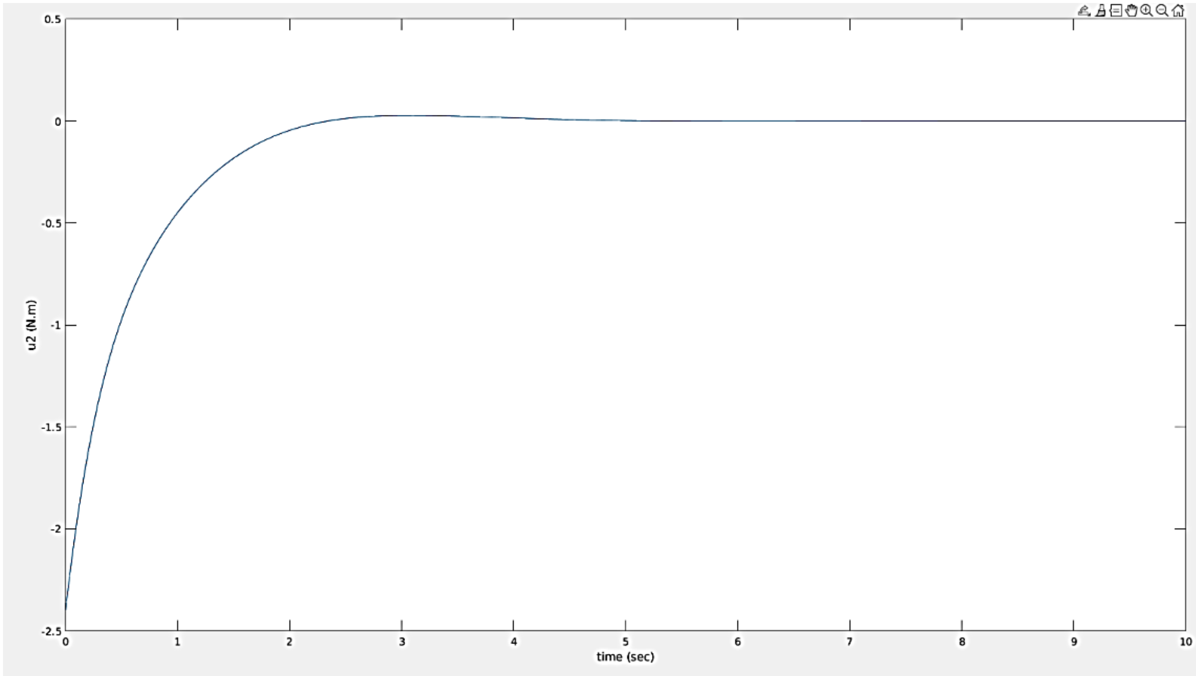
And running the simulation for 10s we get the following results,



Position and Velocity as a function of time



Effort (u1) as a function of time.



Effort (u2) as a function of time.

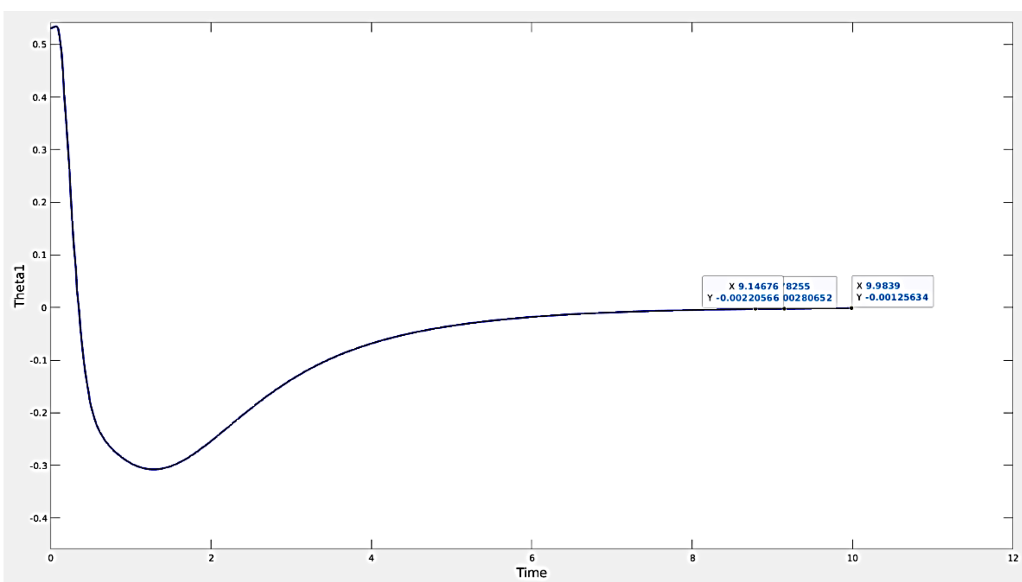
All the states, theta1, theta2, theta1_dot, theta2_dot as well as the efforts u1 and u2 eventually converge to zero.

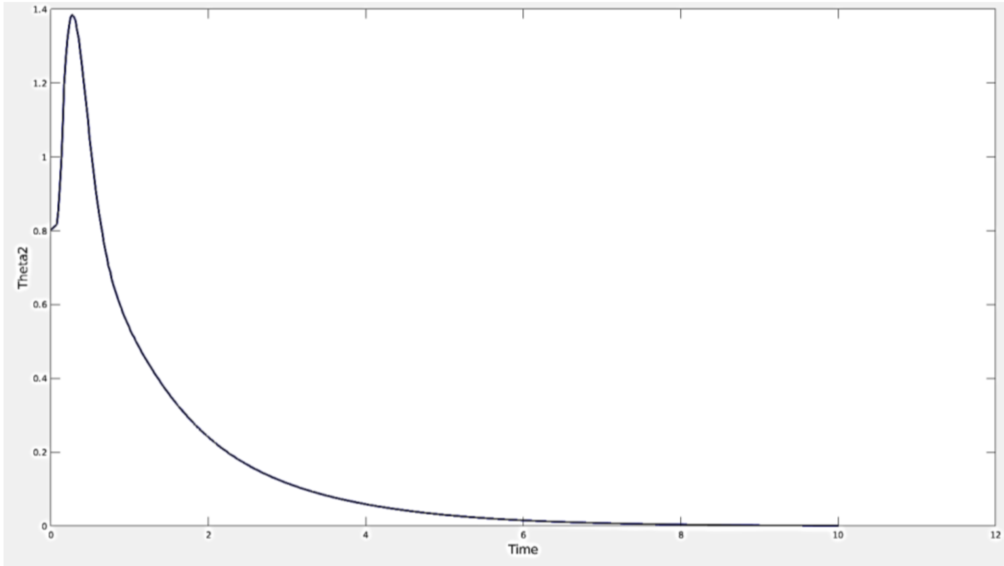
Results from Gazebo Simulation

We apply the developed control law to the RRbot in Gazebo.

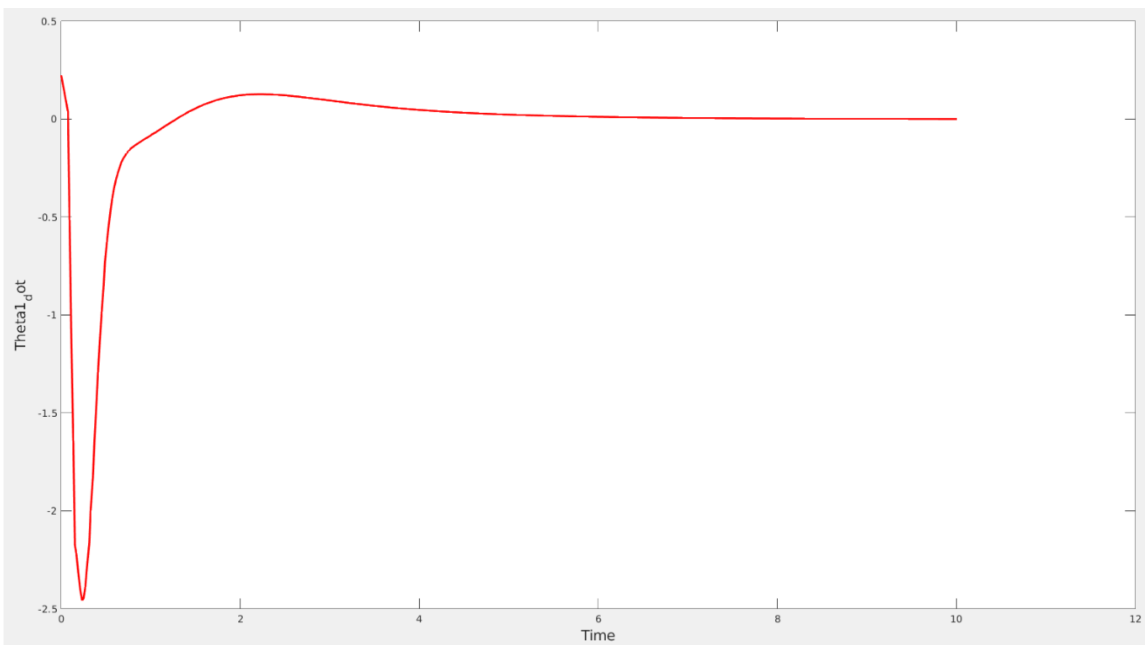
By subscribing to joint states we are able to collect the positions (theta1 and theta2), velocities(theta1_dot and theta2_dot) as well as the efforts (U1 and U2) for a period of 10s.

The results are plotted below.

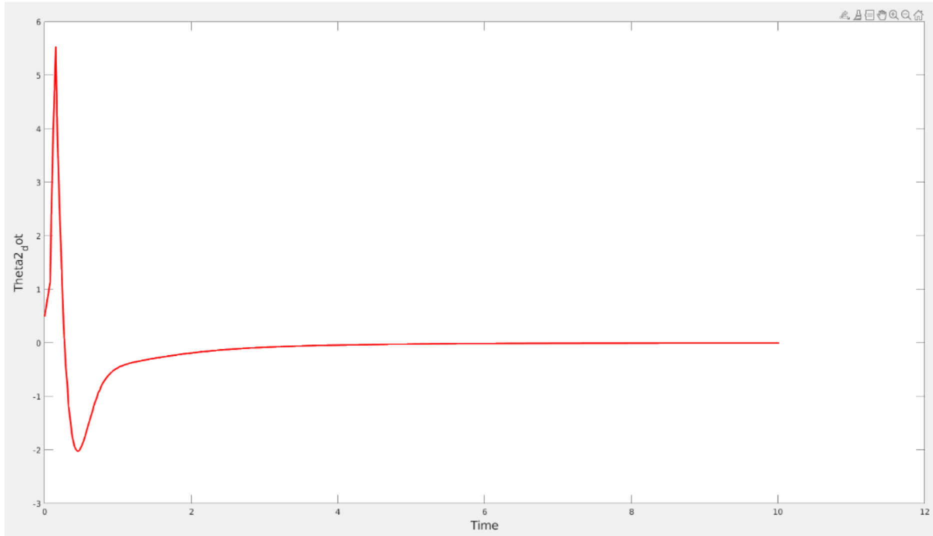


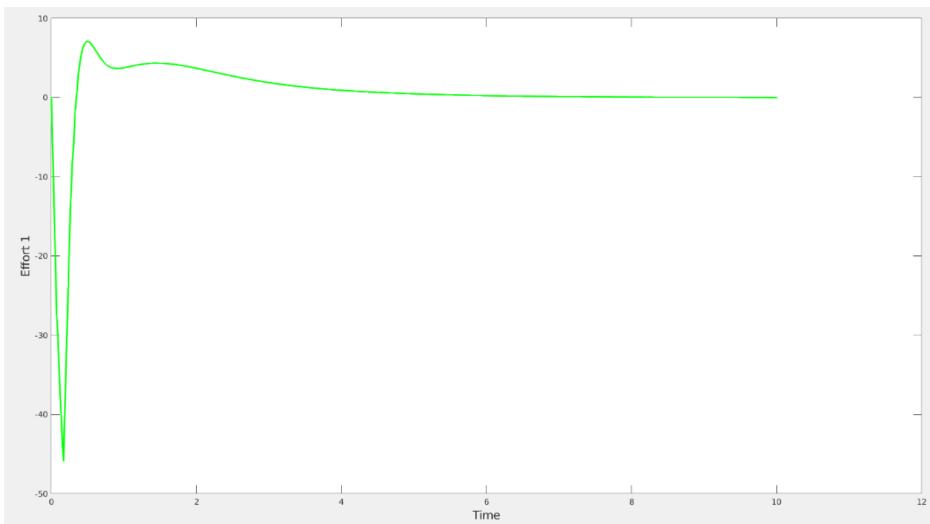


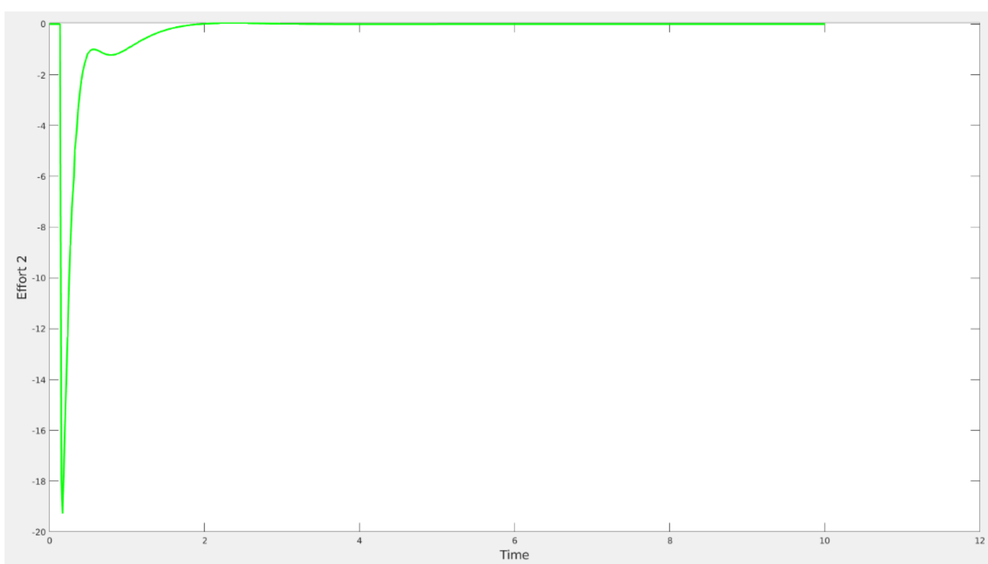
Theta2
Time



Theta1_dot
Time







Both the simulation in Gazebo and Matlab yield satisfactory results as the Control law is able to converge the states and efforts to the equilibrium point. We do see differences in the data plots mainly because of how in

Gazebo the RRbot starts from the upward position and falls due to gravity and has to be re picked up to its starting position for the simulation to begin .

Finally the robot stabilized in the upward position,