State Feedback Control of the RRBot Robotic Arm

a) ***(2 points)*** Using the symbolic equations of motion derived in Programming Assignment 1, determine the equilibrium points of the robot in MATLAB.
Hint: It might be easier to use the original *second-order* EoMs for this purpose, and not the state-space representation. MATLAB functions `subs` and `solve` may come in handy!

⇒ For calculating the equilibrium points,

$$\dot{x} = f(x, u) = 0$$

where, $\dot{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} \dot{\theta_1} \\ \dot{\theta_2} \\ \ddot{\theta_1} \\ \ddot{\theta_2} \end{bmatrix}$

→ x_3 & x_4 are represented by EoM, calculated in the previous assignment.

→ Hence,

$$x^* = (0, 0, 0, 0) \text{ or, } (\pi, 0, 0, 0) \text{ or, } (0, \pi, 0, 0).$$

⇒ FROM MATLAB ,

```
Equilibirum points of the robot:

theta 1 =
 0
pi
 0


theta 2 =
 0
 0
pi


theta1_dot = 0

theta2_dot = 0
```

b) ***(2 points)*** Derive the Jacobian linearization of the symbolic state-space representation (derived in Programming Assignment 1) in MATLAB to obtain the symbolic representation of the state matrix A and the input matrix B. MATLAB function `jacobian` can be used for linearization.
Substitute the physical parameters of the robot (listed below) in the state and input matrices.

$$m_1 = m_2 = 1\ (kg), \quad l_1 = l_2 = 1\ (m), \quad r_1 = r_2 = 0.45\ (m)$$
$$I_1 = I_2 = 0.084\ (kg \cdot m^2), \quad g = 9.81\ (m/s^2)$$

⇒ The Jacobian linearization is performed using x, $\dot{x}$ & u ;

→ $A = \text{Jacobian}(\dot{x}, x) \in R^{4\times4}$

$B = \text{Jacobian}(\dot{x}, u) \in R^{4\times2}$

```
A:
[ 0, 0, 1, 0]
[0, 0, 0, 1]
[ ((597429*cos(theta1))/500000 + (715149*cos(theta1 + theta2)^2*cos(theta1))/800000 + (2304369*sin(theta1 + theta2)^2*cos(theta1))/800000 - (79461*cos(theta1 + theta2)*sin(theta1 +
theta2)*sin(theta1))/40000)/((1701*cos(theta1 + theta2)^2)/100000 + (1701*sin(theta1 + theta2)^2)/100000 + (10101*cos(theta1)^2)/100000 + (10101*sin(theta1)^2)/100000 + (6561*cos(theta1 +
theta2)^2*cos(theta1)^2)/160000 + (38961*cos(theta1 + theta2)^2*sin(theta1)^2)/160000 + (38961*sin(theta1 + theta2)^2*cos(theta1)^2)/160000 + (6561*sin(theta1 + theta2)^2*sin(theta1)^2)/
160000 - (81*cos(theta1 + theta2)*sin(theta1 + theta2)*cos(theta1)*sin(theta1))/200 + 441/62500),
(((81*cos(theta1 + theta2)*sin(theta1 + theta2)*sin(theta1)^2)/200 + (81*cos(theta1 + theta2)^2*cos(theta1)*sin(theta1))/200 - (81*sin(theta1 + theta2)^2*cos(theta1)*sin(theta1))/200 - (81*cos(theta1
+ theta2)*sin(theta1 + theta2)*cos(theta1)^2)/200)*((21*t1)/250 - (21*t2)/250 + (597429*sin(theta1))/500000 + (2304369*cos(theta1 + theta2)^2*sin(theta1))/800000 + (715149*sin(theta1 +
theta2)^2*sin(theta1))/800000 + (81*t1*cos(theta1 + theta2)^2)/400 - (81*t2*cos(theta1 + theta2)^2)/400 + (81*t1*sin(theta1 + theta2)^2)/400 - (81*t2*sin(theta1 + theta2)^2)/400 - (9*t2*cos(theta1
+ theta2)*cos(theta1))/20 - (9*t2*sin(theta1 + theta2)*sin(theta1))/20 - (79461*cos(theta1 + theta2)*sin(theta1 + theta2)*cos(theta1))/40000))/((1701*cos(theta1 + theta2)^2)/100000 +
(1701*sin(theta1 + theta2)^2)/100000 + (10101*cos(theta1)^2)/100000 + (10101*sin(theta1)^2)/100000 + (6561*cos(theta1 + theta2)^2*cos(theta1)^2)/160000 + (38961*cos(theta1 +
theta2)^2*sin(theta1)^2)/160000 + (38961*sin(theta1 + theta2)^2*cos(theta1)^2)/160000 + (6561*sin(theta1 + theta2)^2*sin(theta1)^2)/160000 - (81*cos(theta1 + theta2)*sin(theta1 +
theta2)*cos(theta1)*sin(theta1))/200 + 441/62500)^2 - ((79461*cos(theta1 + theta2)^2*cos(theta1))/40000 - (79461*sin(theta1 + theta2)^2*cos(theta1))/40000 + (9*t2*cos(theta1 +
theta2)*sin(theta1))/20 - (9*t2*sin(theta1 + theta2)*cos(theta1))/20 + (79461*cos(theta1 + theta2)*sin(theta1 + theta2)*sin(theta1))/20000)/((1701*cos(theta1 + theta2)^2)/100000 + (1701*sin(theta1 +
theta2)^2)/100000 + (10101*cos(theta1)^2)/100000 + (10101*sin(theta1)^2)/100000 + (6561*cos(theta1 + theta2)^2*cos(theta1)^2)/160000 + (38961*cos(theta1 + theta2)^2*sin(theta1)^2)/160000 +
(38961*sin(theta1 + theta2)^2*cos(theta1)^2)/160000 + (6561*sin(theta1 + theta2)^2*sin(theta1)^2)/160000 - (81*cos(theta1 + theta2)*sin(theta1 + theta2)*cos(theta1)*sin(theta1))/200 +
441/62500), 0, 0]
[-((597429*cos(theta1))/500000 - (185409*cos(theta1 + theta2))/500000 + (874071*cos(theta1 + theta2)*cos(theta1)^2)/800000 + (715149*cos(theta1 + theta2)^2*cos(theta1))/800000 -
(4246749*cos(theta1 + theta2)*sin(theta1)^2)/800000 + (2304369*sin(theta1 + theta2)^2*cos(theta1))/800000 - (79461*cos(theta1 + theta2)*sin(theta1 + theta2)*sin(theta1))/40000 +
(256041*sin(theta1 + theta2)*cos(theta1)*sin(theta1))/40000)/((1701*cos(theta1 + theta2)^2)/100000 + (1701*sin(theta1 + theta2)^2)/100000 + (10101*cos(theta1)^2)/100000 + (10101*sin(theta1)^2)/
100000 + (6561*cos(theta1 + theta2)^2*cos(theta1)^2)/160000 + (38961*cos(theta1 + theta2)^2*sin(theta1)^2)/160000 + (38961*sin(theta1 + theta2)^2*cos(theta1)^2)/160000 + (6561*sin(theta1 +
theta2)^2*sin(theta1)^2)/160000 - (81*cos(theta1 + theta2)*sin(theta1 + theta2)*cos(theta1)*sin(theta1))/200 + 441/62500), ((185409*cos(theta1 + theta2))/500000 + (4246749*cos(theta1 +
theta2)*cos(theta1)^2)/800000 + (79461*cos(theta1 + theta2)^2*cos(theta1))/40000 - (874071*cos(theta1 + theta2)*sin(theta1)^2)/800000 - (79461*sin(theta1 + theta2)^2*cos(theta1))/40000 -
(9*t1*cos(theta1 + theta2)*sin(theta1))/20 + (9*t1*sin(theta1 + theta2)*cos(theta1))/20 + (9*t2*cos(theta1 + theta2)*sin(theta1))/10 - (9*t2*sin(theta1 + theta2)*cos(theta1))/10 + (79461*cos(theta1 +
theta2)*sin(theta1 + theta2)*sin(theta1))/20000 + (256041*sin(theta1 + theta2)*cos(theta1)*sin(theta1))/40000)/((1701*cos(theta1 + theta2)^2)/100000 + (1701*sin(theta1 + theta2)^2)/100000 +
(10101*cos(theta1)^2)/100000 + (10101*sin(theta1)^2)/100000 + (6561*cos(theta1 + theta2)^2*cos(theta1)^2)/160000 + (38961*cos(theta1 + theta2)^2*sin(theta1)^2)/160000 + (38961*sin(theta1 +
theta2)^2*cos(theta1)^2)/160000 + (6561*sin(theta1 + theta2)^2*sin(theta1)^2)/160000 - (81*cos(theta1 + theta2)*sin(theta1 + theta2)*cos(theta1)*sin(theta1))/200 + 441/62500) - (((81*cos(theta1 +
theta2)*sin(theta1 + theta2)*sin(theta1)^2)/200 + (81*cos(theta1 + theta2)^2*cos(theta1)*sin(theta1))/200 - (81*sin(theta1 + theta2)^2*cos(theta1)*sin(theta1))/200 - (81*cos(theta1 +
theta2)*sin(theta1 + theta2)*cos(theta1)^2)/200)*((21*t1)/250 - (21*t2)/125 - (185409*sin(theta1 + theta2))/500000 + (597429*sin(theta1))/500000 - (481*t2*cos(theta1)^2)/400 -
(481*t2*sin(theta1)^2)/400 - (4246749*sin(theta1 + theta2)*cos(theta1)^2)/800000 + (2304369*cos(theta1 + theta2)^2*sin(theta1))/800000 + (874071*sin(theta1 + theta2)*sin(theta1)^2)/800000 +
(715149*sin(theta1 + theta2)^2*sin(theta1))/800000 + (81*t1*cos(theta1 + theta2)^2)/400 - (81*t2*cos(theta1 + theta2)^2)/400 + (81*t1*sin(theta1 + theta2)^2)/400 - (81*t2*sin(theta1 + theta2)^2)/
400 + (9*t1*cos(theta1 + theta2)*cos(theta1))/20 - (9*t2*cos(theta1 + theta2)*cos(theta1))/10 + (9*t1*sin(theta1 + theta2)*sin(theta1))/20 - (9*t2*sin(theta1 + theta2)*sin(theta1))/10 -
(79461*cos(theta1 + theta2)*sin(theta1 + theta2)*cos(theta1))/40000 + (256041*cos(theta1 + theta2)*cos(theta1)*sin(theta1))/40000))/((1701*cos(theta1 + theta2)^2)/100000 + (1701*sin(theta1 +
theta2)^2)/100000 + (10101*cos(theta1)^2)/100000 + (10101*sin(theta1)^2)/100000 + (6561*cos(theta1 + theta2)^2*cos(theta1)^2)/160000 + (38961*cos(theta1 + theta2)^2*sin(theta1)^2)/160000 +
(38961*sin(theta1 + theta2)^2*cos(theta1)^2)/160000 + (6561*sin(theta1 + theta2)^2*sin(theta1)^2)/160000 - (81*cos(theta1 + theta2)*sin(theta1 + theta2)*cos(theta1)*sin(theta1))/200 +
441/62500)^2, 0, 0]
```

```
B:
[0,  0]
[0,  0]
[ ((81*cos(theta1 + theta2)^2)/400 + (81*sin(theta1 + theta2)^2)/400 + 21/250)/((1701*cos(theta1 + theta2)^2)/100000 + (1701*sin(theta1 + theta2)^2)/100000 + (10101*cos(theta1)^2)/100000 +
(10101*sin(theta1)^2)/100000 + (6561*cos(theta1 + theta2)^2*cos(theta1)^2)/160000 + (38961*cos(theta1 + theta2)^2*sin(theta1)^2)/160000 + (38961*sin(theta1 + theta2)^2*cos(theta1)^2)/160000
+ (6561*sin(theta1 + theta2)^2*sin(theta1)^2)/160000 - (81*cos(theta1 + theta2)*sin(theta1 + theta2)*cos(theta1)*sin(theta1))/200 + 441/62500),                         -((9*sin(theta1 +
theta2)*sin(theta1))/20 + (81*cos(theta1 + theta2)^2)/400 + (81*sin(theta1 + theta2)^2)/400 + (9*cos(theta1 + theta2)*cos(theta1))/20 + 21/250)/((1701*cos(theta1 + theta2)^2)/100000 +
(1701*sin(theta1 + theta2)^2)/100000 + (10101*cos(theta1)^2)/100000 + (10101*sin(theta1)^2)/100000 + (6561*cos(theta1 + theta2)^2*cos(theta1)^2)/160000 + (38961*cos(theta1 +
theta2)^2*sin(theta1)^2)/160000 + (38961*sin(theta1 + theta2)^2*cos(theta1)^2)/160000 + (6561*sin(theta1 + theta2)^2*sin(theta1)^2)/160000 - (81*cos(theta1 + theta2)*sin(theta1 +
theta2)*cos(theta1)*sin(theta1))/200 + 441/62500)]
[-((9*sin(theta1 + theta2)*sin(theta1))/20 + (81*cos(theta1 + theta2)^2)/400 + (81*sin(theta1 + theta2)^2)/400 + (9*cos(theta1 + theta2)*cos(theta1))/20 + 21/250)/((1701*cos(theta1 + theta2)^2)/
100000 + (1701*sin(theta1 + theta2)^2)/100000 + (10101*cos(theta1)^2)/100000 + (10101*sin(theta1)^2)/100000 + (6561*cos(theta1 + theta2)^2*cos(theta1)^2)/160000 + (38961*cos(theta1 +
theta2)^2*sin(theta1)^2)/160000 + (38961*sin(theta1 + theta2)^2*cos(theta1)^2)/160000 + (6561*sin(theta1 + theta2)^2*sin(theta1)^2)/160000 - (81*cos(theta1 + theta2)*sin(theta1 +
theta2)*cos(theta1)*sin(theta1))/200 + 441/62500), ((9*sin(theta1 + theta2)*sin(theta1))/10 + (81*cos(theta1 + theta2)^2)/400 + (81*sin(theta1 + theta2)^2)/400 + (481*cos(theta1)^2)/400 +
(481*sin(theta1)^2)/400 + (9*cos(theta1 + theta2)*cos(theta1))/10 + 21/125)/((1701*cos(theta1 + theta2)^2)/100000 + (1701*sin(theta1 + theta2)^2)/100000 + (10101*cos(theta1)^2)/100000 +
(10101*sin(theta1)^2)/100000 + (6561*cos(theta1 + theta2)^2*cos(theta1)^2)/160000 + (38961*cos(theta1 + theta2)^2*sin(theta1)^2)/160000 + (38961*sin(theta1 + theta2)^2*cos(theta1)^2)/160000
+ (6561*sin(theta1 + theta2)^2*sin(theta1)^2)/160000 - (81*cos(theta1 + theta2)*sin(theta1 + theta2)*cos(theta1)*sin(theta1))/200 + 441/62500)]
```

c) ***(2 points)*** Investigate the stability properties of the linearized system around each of the equilibrium points found in Step (a).

⇒ For checking the stability of the linearized system, the eigen values of Matrix A at each of the equilibrium point is calculated.

⇒ The system is,

→ Asymptotically stable ⇒ $Re(\lambda i) < 0$

→ Marginally stable ⇒ $Re(\lambda i) \leq 0$

→ Unstable ⇒ $Re(\lambda i) > 0$.

⇒ Result of MATLAB Calculations,

```
Stability around: (0,0,0,0)
Eigenvalues are: 7.168, 2.713, -7.168, -2.713
The linearized system is unstable around the equilibirum point (0,0,0,0)

Stability around: (pi,0,0,0)
Eigenvalues are: -0.000+7.168j, -0.000-7.168j, -0.000+2.713j, -0.000-2.713j
The linearized system is marginally stable around the equilibirum point (pi,0,0,0)

Stability around: (0,pi,0,0)
Eigenvalues are: -3.900+0.000j, 3.900+0.000j, 0.000+4.986j, 0.000-4.986j
The linearized system is unstable around the equilibirum point (0,pi,0,0)
```

d) ***(0.5 points)*** Investigate the controllability of the linearized system around the equilibrium point corresponding to the "upward" configuration.

⇒ The linearized system is controllable if Rank(C) = n.

where, C = Controllability Matrix

n ⇒ $A \in R^{n \times n}$

→ so, for the "upward" configuration

$x^* = (0, 0, 0, 0)$

⇒ Result of MATLAB Calculation.

```
Rank of Controllability Matrix: 4
As Rank of C equals to n, the linearized system is controllable for Upward Configuration
```

e) ***(2.5 points)*** Design a state-feedback control to stabilize the system around the upward equilibrium point. Feel free to use MATLAB function `place`. Include the selected eigenvalues and the resulting gains in your report.

```
% ---------- State-Feedback Controller ------------ %

% choosing the desired eigenvalues
desiredEigen = [-1.1, -2.2, -3.3, -4.4];
disp("Selected Eigenvalues are: [-1.1, -2.2, -3.3, -4.4]");
disp(" ");
% placing the desired eigenvalues to obtain the gain matrix
K = place(A,B,desiredEigen);
disp("Calculated Gain Matrix: ");
display(K);
```

⇒ RESULTS :-

```
Selected Eigenvalues are: [-1.1, -2.2, -3.3, -4.4]

Calculated Gain Matrix:

K =

   29.1901    4.1846   11.4551    2.1855
    7.3871    4.9028    3.3496    1.0601
```

f) ***(6 points)*** Use `ode45` and the `ode` function developed in Programming Assignment 1 to construct a simulation of the system in MATLAB with the initial conditions $\theta_1(0) = 30°$, $\theta_2(0) = 45°$, $\dot{\theta}_1(0) = 0$ and $\dot{\theta}_2(0) = 0$ and a time-span of 10 seconds.
Do not forget to implement your feedback control law (designed in Step (e)) inside the `ode` function. Plot the state trajectories (of the state space form) and the control inputs trajectories to evaluate the performance.
If the performance is not satisfactory (e.g. the system does not converge to the "upward equilibrium point), go back to Step (e) and change the location of the assigned eigenvalues.
Optional: Try to choose the eigenvalues such that the joint torques (i.e. control inputs) stay within the bounds of:

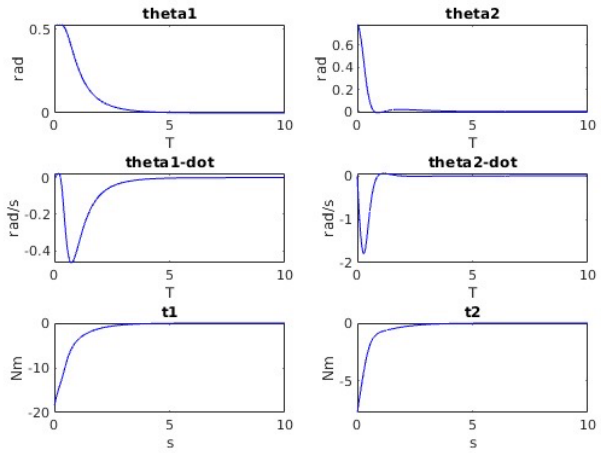
$$-20 \leq u_1 \leq 20\ (Nm), \qquad -10 \leq u_2 \leq 10\ (Nm)$$

⇒ Calculated torques based on the selected eigen values.

```
Eigenvalues are selected such that:

Torque1: -18.570 < u1 < -0.000

Torque2: -7.719 < u2 < -0.000
```

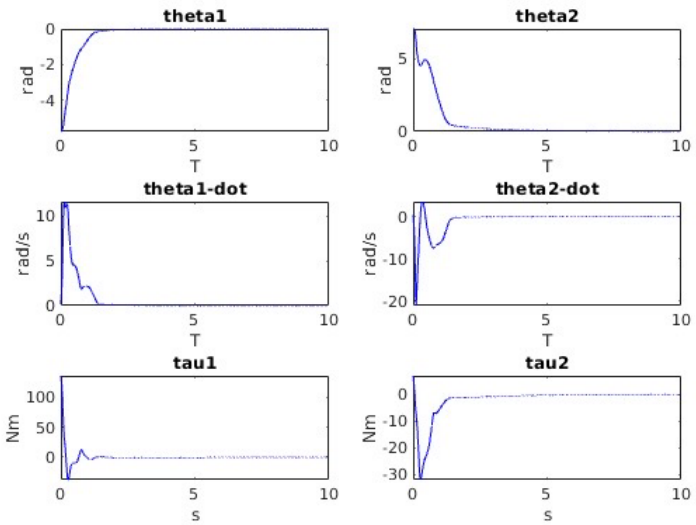


State and Control Inputs Trajectories (ODE 45 Simulation)

g) ***(5 points)*** Copy and paste the following code into a new MATLAB script named `rrbot_control.m`.

Note: as an alternative, you can use the ROS Python script posted on Canvas for this purpose.

Complete the code inside the `while` loop to control the RRbot robot in Gazebo for 10 seconds using your state-feedback controller designed in Step (f). Sample the data at each loop to be plotted at the end. Feel free to define new functions and variables in your program if needed. Compare the resulting trajectories and control inputs in Gazebo with those obtained in Step (f) in MATLAB. Discuss your findings in your final report.

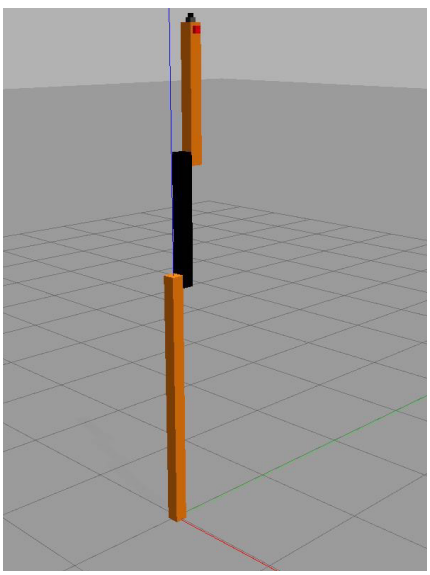


State and Control Inputs Trajectories (Gazebo Simulation)

=) The ODE45 simulation starts directly from the initial conditions in absence of physics engine. On the other hand, in Gazebo simulation, both the links free fall because of Gravity & then to reach the desired "Upward" Configuration. the system needs to apply more joint torques. Starting from close to 100 Nm for Joint-1 & -30 Nm for Joint-2 based on the selected eigen values.

=) However, the state-feedback controller is designed efficiently to achieve the desired configuration within the short amount of time as seen in the figure above.

→ In Gazebo simulation, the dynamics of link-2 affects the motion of link-1 as well and I believe that explains the small bumps visible in the control input trajectory for Task 1.



RRbot in Upward configuration using the state feedback controller