School of Engineering and Material Science

DENM011/DEN408 Robotics

Computer-Aided Simulation Tutorial Exercise 2021

Author:
Name: Vishal Kashyap
Student number: 210684838

Table of Contents

1. Objective

A control system is an integral part of any robot manipulator. It directs the movement of actuators to achieve a desired position or configuration of the robot. A controller works by controlling the velocity or force exerted on the actuators. It usually has a feedback loop that looks at the system's output and modifies the controller's behaviour as required. This exercise will look at a two-link planar anthropomorphic manipulator with a computed torque controller. The equations of motion for this manipulator will be derived and then validated using MATLAB SIMULINK. The stability and robustness of this controller will also be looked at in the presence of parameter uncertainty and modelling errors. Finally, comments will be made on its practical usability and scope for further improvement.

2. Background theory

This section goes through the problem formulation and gives a brief overview of how we arrive at the dynamic equations of motion for a 2-link manipulator. In Section 2A, we derive the equations of motion using the Lagrange formulation. The computed torque controller is then introduced in section 2B. Finally, section 2C proves that the computed torque controller gives us zero tracking error asymptotically .

A. Lagrange formulation

We want to obtain an anthropomorphic two-link manipulator's dynamic equation of motion, as shown in figure 1. This equation will relate torque applied at joints to their respective angular acceleration. We will work in the $(\theta 1, \theta 2)$ coordinate system (see figure 1) and use the Euler Lagrange equations to derive this equation.

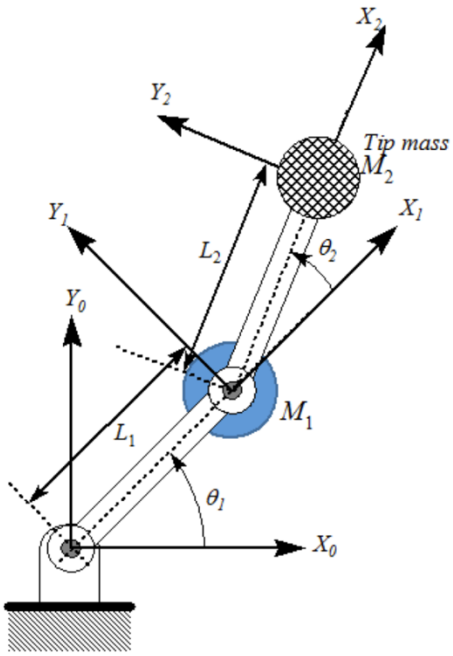

Figure 1. Two link manipulator

The two link manipulator is modelled as shown in the figure above. The links have lengths L1 and L2 respectively and are connected using a revolute joint. The end effector and joint between links 1 and 2 are modelled as point masses. We index link 1, the joint between link 1 and link 2, link 2 and end-effector as

1, 2, 3 and 4, respectively. To derive the potential and kinetic energies of these links, we can write their position in the (X_0, Y_0) reference frame:

$$\begin{bmatrix} X_1 \\ Y_1 \\ X_2 \\ Y_2 \\ X_3 \\ Y_3 \\ X_4 \\ Y_4 \end{bmatrix} = \begin{bmatrix} L_{1cg}\cos\theta_1 \\ L_{1cg}\sin\theta_1 \\ L_1\cos\theta_1 + L_{2cg}\cos(\theta_1+\theta_2) \\ L_1\sin\theta_1 + L_{2cg}\sin(\theta_1+\theta_1) \\ L_1\cos\theta_1 \\ L_1\sin\theta_1 \\ L_1\cos\theta_1 + L_2\cos(\theta_1+\theta_2) \\ L_1\sin\theta_1 + L_2\sin(\theta_1+\theta_1) \end{bmatrix} \tag{1}$$

Differentiating these positions w.r.t time will give us their horizontal and vertical velocities.

$$\begin{bmatrix} \dot{X}_1 \\ \dot{Y}_1 \\ \dot{X}_2 \\ \dot{Y}_2 \\ \dot{X}_3 \\ \dot{Y}_3 \\ \dot{X}_4 \\ \dot{Y}_4 \end{bmatrix} = \begin{bmatrix} -L_{1cg}\sin\theta_1\,\dot{\theta}_1 \\ L_{1cg}\cos\theta_1\,\dot{\theta}_1 \\ -L_1\sin\theta_1\,\dot{\theta}_1 - L_{2cg}\sin(\theta_1+\theta_2)(\dot{\theta}_1+\dot{\theta}_2) \\ L_1\cos\theta_1\,\dot{\theta}_1 + L_{2cg}\cos(\theta_1+\theta_2)(\dot{\theta}_1+\dot{\theta}_1) \\ -L_1\sin\theta_1\,\dot{\theta}_1 \\ L_1\cos\theta_1\,\dot{\theta}_1 \\ -L_1\sin\theta_1\,\dot{\theta}_1 - L_2\sin(\theta_1+\theta_2)(\dot{\theta}_1+\dot{\theta}_2) \\ L_1\cos\theta_1\,\dot{\theta}_1 + L_2\cos(\theta_1+\theta_2)(\dot{\theta}_1+\dot{\theta}_1) \end{bmatrix} \tag{2}$$

The Lagrangian is defined as the difference between kinetic and potential energy. To get the Lagrangian, we need to obtain the system's kinetic and potential energy, which is the sum of kinetic and potential energies of the individual links. Kinetic energy, K_i is defined as $\frac{1}{2}m_i\left(\dot{X}_i^2 + \dot{Y}_i^2\right) + \frac{1}{2}m_i k_i^2 \omega_i^2$. Here, k_i and ω_i are the radius of gyration and angular velocity of link i. We can write the potential energy P_i as $m_i g Y_i$.

$$\begin{bmatrix} K_1 \\ P_1 \\ K_2 \\ P_2 \\ K_3 \\ P_3 \\ K_4 \\ P_4 \end{bmatrix} = \begin{bmatrix} 0.5m_1(L_{1cg}^2\dot{\theta}_1^2 + k_1\dot{\theta}_1^2) \\ m_1 g L_{1cg}\sin\theta_1 \\ 0.5m_2\left[L_1^2\dot{\theta}_1^2 + L_{2cg}^2\left(\dot{\theta}_1+\dot{\theta}_2\right)^2 + 2L_1L_{2cg}\dot{\theta}_1(\dot{\theta}_1+\dot{\theta}_1)\cos\theta_2 + k_2\left(\dot{\theta}_1+\dot{\theta}_2\right)^2\right] \\ m_2 g(L_1\sin\theta_1 + L_{2cg}\sin(\theta_1+\theta_2) \\ 0.5M_1L_1^2\dot{\theta}_1^2 \\ M_1 g L_1\sin\theta_1 \\ 0.5M_2\left[L_1^2\dot{\theta}_1^2 + L_2^2\left(\dot{\theta}_1+\dot{\theta}_2\right)^2 + 2L_1L_2\dot{\theta}_1(\dot{\theta}_1+\dot{\theta}_1)\cos\theta_2\right] \\ M_2 g(L_1\sin\theta_1 + L_2\sin(\theta_1+\theta_2) \end{bmatrix} \tag{3}$$

The Lagrangian can now be defined as,

$$L = K_1 + K_2 + K_3 + K_4 - P_1 - P_2 - P_3 - P_4 \tag{4}$$

Substituting expressions for kinetic and potential energy we get,

$$L = \frac{1}{2} I_{11} \dot{\theta}_1^2 + \frac{1}{2} I_{22} \left(\dot{\theta}_1 + \dot{\theta}_2\right)^2 + I_{21} \dot{\theta}_1 \left(\dot{\theta}_1 + \dot{\theta}_2\right) \cos\theta_2 - \Gamma_{11} g \sin\theta_1 - \Gamma_{22} g \sin(\theta_1 + \theta_2) \tag{5.1}$$

We have defined $I_{11},\ I_{22},\ I_{21}, \Gamma_{11}\ and\ \Gamma_{22}$ to simply the expression,

$$\begin{aligned}
&I_{11} = m_1\left(L_{1cg}^2 + k_{1cg}^2\right) + M_1 L_1^2 + (m_2 + M_2) L_1^2 \\
&I_{21} = \left(m_2 L_{2cg} + M_2 L_2\right) L_1 = \Gamma_{22} L_1 \\
&I_{22} = m_2\left(L_{2cg}^2 + k_{2cg}^2\right) + M_2 L_2^2 \\
&\Gamma_1 = \Gamma_{11} \cos\theta_1 + \Gamma_{22} \cos(\theta_1 + \theta_2),\ \Gamma_2 = \Gamma_{22} \cos(\theta_1 + \theta_2) \\
&\Gamma_{11} = \left(m_1 L_{1cg} + m_2 L_1 + M_1 L_1 + M_2 L_1\right), \Gamma_{22} = (m_2 L_{2cg} + M_2 L_2)
\end{aligned} \tag{5.2}$$

The Euler-Lagrange equations then are,

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\theta}_i} - \frac{\partial L}{\partial \theta_i} = T_i \tag{6}$$

Applying these to Lagrangian in equation 5.1, we get

$$\begin{bmatrix} I_{11} + I_{21}\cos\theta_2 & I_{21}\cos\theta_2 + I_{22} \\ I_{21}\cos\theta_2 & I_{22} \end{bmatrix} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_1 + \ddot{\theta}_2 \end{bmatrix} + I_{21} \sin\theta_2 \begin{bmatrix} 2\dot{\theta}_1\dot{\theta}_2 - \dot{\theta}_2^2 \\ \dot{\theta}_1^2 \end{bmatrix} + g \begin{bmatrix} \Gamma_1 \\ \Gamma_2 \end{bmatrix} = \begin{bmatrix} T_1 \\ T_2 \end{bmatrix} \tag{7}$$

In the special case when, $L_1 = L_2 = 2L_{1cg} = 2L_{2cg} = L$, $m_1 = m_2 = m$, $M_1 = \mu_1 m$, $M_2 = \mu_2 m$ and $k_{1cg}^2 = k_{2cg}^2 = \frac{L^2}{12}$. Also, defining $\dot{\theta}_1 = \omega_1$ and $\dot{\theta}_2 = \omega_2 - \omega_1$ we can write the equations of motion of the manipulator in state space form as,

$$\begin{bmatrix} \dot{\omega}_1 \\ \dot{\omega}_2 \end{bmatrix} = I_{robot}^{-1} \left\{ \frac{1}{mL^2} \begin{bmatrix} T_1 \\ T_2 \end{bmatrix} - \left(\frac{1}{2} + \mu_2\right) \sin\theta_2 \begin{bmatrix} \omega_1^2 - \omega_2^2 \\ \omega_1^2 \end{bmatrix} - \frac{g}{L} \begin{bmatrix} \left(\frac{3}{2} + \mu_1 + \mu_2\right) \\ \left(\frac{1}{2} + \mu_2\right) \end{bmatrix} \right\} \tag{8}$$

B. Computed torque controller

Given the robot dynamic equations in the previous section, we have a direct relation between torque and state-space variables. In an ideal world, we could use this equation to generate a torque based on the desired $\theta, \dot{\theta}\ and\ \ddot{\theta}$ at any time t. The torque generated in this way would guide our links to the final value of θ.

The situation in the real world is never this simple as our dynamic model of the robot system is not perfect and the system parameters might be inaccurate. This results in error accumulation over time and failure to reach the final desired value of θ. To rectify this error, we use controllers that use the difference in desired and actual variables at any time t to generate corrective forces. There are many controllers out there which could be used for our two-link manipulator, but we focus on the computed torque controller, which is a type of feedforward control that uses both dynamic equations of motion and error between desired and actual state variables to find control torques. Next, we derive the control torques using this controller for the two link manipulator

If we get $\theta_d, \dot{\theta}_d\ and\ \ddot{\theta}_d$ from the trajectory generator, we can find the required torque at any point in time using the following general equation of motion (Lynch, 2017):

$$\tau(t) = M[\theta_d(t)]\ddot{\theta}_d(t) + h[\dot{\theta}_d(t), \theta_d(t)] \tag{9}$$

We add the PD control to this model of robot dynamics. A PD control has the form

$$u(t) = K_p\theta_e + K_d\dot{\theta}_e \tag{10}$$

Where $\theta_e = \theta - \theta_d$, which is the error between the actual and desired trajectory, and K_p and K_d are the proportional and derivative gains. Adding this control to equation 10 we get an equation of torque control

$$\tau = M(\theta)\left[\ddot{\theta}_d + K_p\theta_e + K_d\dot{\theta}_e\right] + h(\dot{\theta}_d, \theta_d) \tag{11}$$

In the above equation, $M\ddot{\theta}_d$ generates the torque required for the planned trajectory while using feedback $(K_p\theta_e + K_d\dot{\theta}_e)$ to make sure θ_e goes to zero. The final $h(\dot{\theta}_d, \theta_d)$ term takes care of non-linear dynamics. This general equation is known as the computed torque controller. The following equations show how this controller can be applied to the two-link manipulator.

For the given two-link manipulator,

$$M(\theta) = \begin{bmatrix} I_{11} + 2I_{21}\cos\theta_2 + I_{22} & I_{21}\cos\theta_2 + I_{22} \\ I_{21}\cos\theta_2 + I_{22} & I_{22} \end{bmatrix} \tag{12}$$

$$h(\dot{\theta}_d, \theta_d) = I_{21}\sin\theta_2 \begin{bmatrix} 2\dot{\theta}_1\dot{\theta}_2 - \dot{\theta}_2^2 \\ \dot{\theta}_1^2 \end{bmatrix} + g\begin{bmatrix} \Gamma_1 \\ \Gamma_2 \end{bmatrix} \tag{13}$$

Therefore the equation for control torque is

$$\begin{bmatrix} T_{1c} \\ T_{2c} \end{bmatrix} = \begin{bmatrix} I_{11} + 2I_{21}\cos\theta_2 + I_{22} & I_{21}\cos\theta_2 + I_{22} \\ I_{21}\cos\theta_2 + I_{22} & I_{22} \end{bmatrix}\begin{bmatrix} \ddot{\theta}_{d1} + K_p\theta_{e1} + K_d\dot{\theta}_{e1} \\ \ddot{\theta}_{d2} + K_p\theta_{e2} + K_d\dot{\theta}_{e2} \end{bmatrix} + I_{21}\sin\theta_2\begin{bmatrix} 2\dot{\theta}_1\dot{\theta}_2 - \dot{\theta}_2^2 \\ \dot{\theta}_1^2 \end{bmatrix} + g\begin{bmatrix} \Gamma_1 \\ \Gamma_2 \end{bmatrix} \tag{14}$$

C. Error Dynamics

In the previous section, we derived the computed torque controller for the two link manipulator. In this section, we will see how the tracking error in our system changes with time. Tracking error in the system is defined as the difference between desired and actual trajectories. Substituting the equation for control torque (equation 14) in the dynamics equation (equation 7), we get,

$$\begin{bmatrix} I_{11} + 2I_{21}\cos\theta_2 + I_{22} & I_{21}\cos\theta_2 + I_{22} \\ I_{21}\cos\theta_2 + I_{22} & I_{22} \end{bmatrix}\begin{bmatrix} \ddot{\theta}_{e1} + K_p\theta_{e1} + K_d\dot{\theta}_{e1} \\ \ddot{\theta}_{e2} + K_p\theta_{e2} + K_d\dot{\theta}_{e2} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{15}$$

The equations for tracking error reduce to the following form,

$$\ddot{\theta}_e + K_d\dot{\theta}_e + K_p\theta_e = 0 \tag{16}$$

The above equation in standard second-order form can be written as

$$\ddot{\theta}_e + 2\zeta\omega_n\dot{\theta}_e + \omega_n^2\theta_e = 0 \tag{17}$$

Here, ζ is the damping ratio and, ω_n is the natural frequency.

$$\omega_n = \sqrt{K_p} \quad and \quad \zeta = \frac{K_d}{2\sqrt{K_p}} \tag{18}$$

For stability, K_p and K_d must be chosen positive. Also, we can impose the critical damping condition on K_p and K_d to have no overshoot and fast response (Lynch, 2017). Then our standard second-order form will become

$$\ddot{\theta}_e + 2\omega_n\dot{\theta}_e + \omega_n^2\theta_e = 0 \quad and. \quad K_d = 2\sqrt{K_p} \tag{19}$$

Thus, a computed torque controller ensures that our system is well behaved. Looking at equation 15, we can see the terms of the non-linear dynamic cancel out, and for the simulation **we can put $\boldsymbol{g} = \mathbf{0}$.** This can also be seen if we substitute the value of control torques from equation 14 for the value of torques in equation 8. Due to the design of our controller the angular accelerations no longer depend on non-linear terms.

3. Control Loop

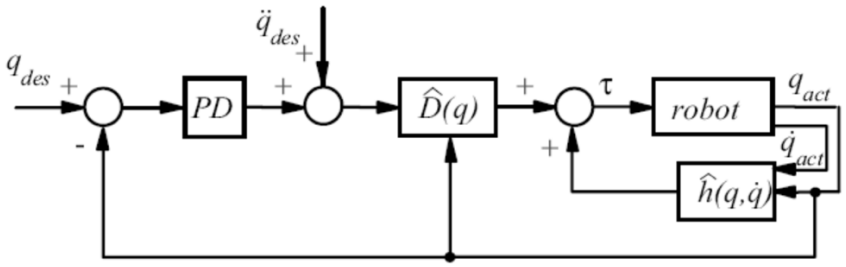

Figure 2: Control loop for the Computed torque controller

The designed computed torque controller uses the above closed feedback loops in each of the servos for the joints. As observed in section 2.C, the feedback signal cancels the effects of Coriolis and centrifugal torques, friction, gravity, and the manipulator inertia torques. After all the cancellations, auxiliary controls drive the controller which are calculated in the outer loop. Here we have used a simple PD controller, but depending on the application, it can be extended to other controllers as well.

4. Simulation Setup

Simulation of the described two-link anthropomorphic controller was done using MATLAB Simulink. The designed Simulink model has four subsystems. The flowchart below sums up the purpose and connection between each of these.

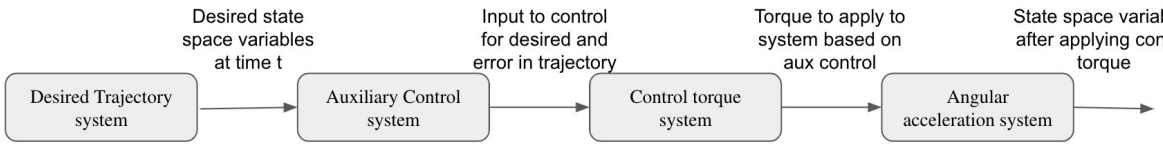

Figure 3: Flow of information between sub-systems

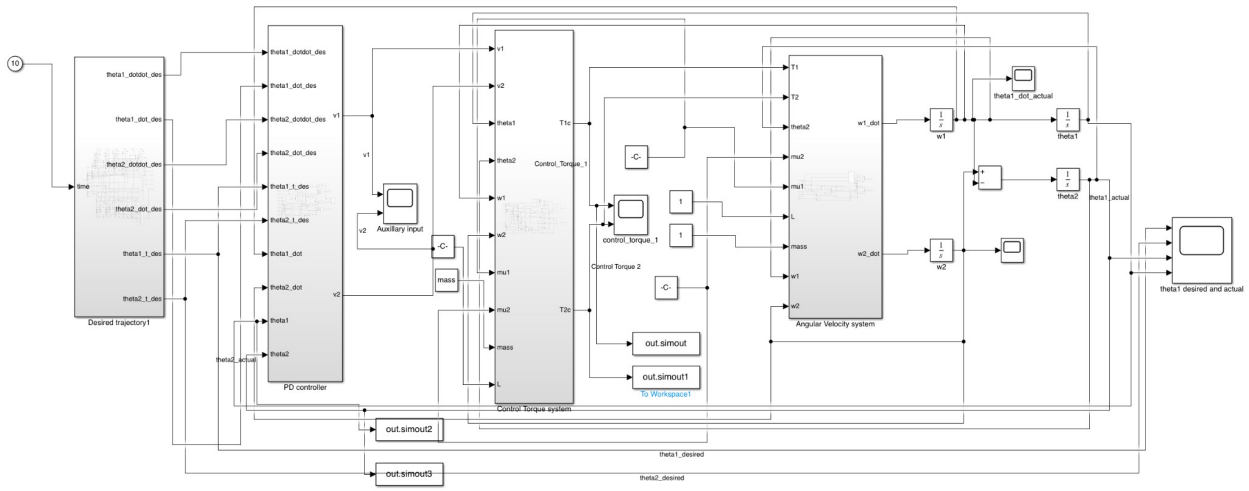

Figure 4: Overall System Block diagram consists of 4 major subsystems. Desired trajectory system, Auxiliary control system, control torque system, and angular acceleration system (From left to right)

A. Desired Trajectory subsystem

Input: Initial conditions on θ_1, θ_2 and simulation time t
Output: $\theta_{1d}(t), \theta_{2d}(t), \dot{\theta}_{1d}(t), \dot{\theta}_{2d}(t), \ddot{\theta}_{1d}(t), \ddot{\theta}_{2d}(t)$

The desired trajectory system takes in the initial and final coordinates of the state variable we want to control and outputs a trajectory to achieve this. In the case of a two-link manipulator, we feed initial and final values of θ_1, θ_2 and find the minimum jerk trajectory (Sharkawy, 2021). Taking the lagrangian as $\ddot{\theta}$, we get the following equation for $\theta_d(t)$:

$$\begin{aligned} &\theta(t) = c_5t^5 + c_4t^4 + c_3t^3 + c_2t^2 + c_1t + c_0 \qquad t \leq 10\ sec \\ &\theta(t) = \theta\left(t_{final}\right) \qquad\qquad\qquad\qquad\qquad\quad t > 10\ sec \end{aligned} \tag{20}$$

Six boundary conditions are needed to solve this trajectory,
Condition 1 and 2: $\theta(0)\ and\ \theta(t_{final})$
Condition 3 and 4: Angular velocity at $t = 0$ and $t = t_{final}$ is 0.
Condition 5 and 6: Angular acceleration at $t = 0$ and $t = t_{final}$ is 0.

Condition 3-6 ensures a smooth trajectory is generated. After solving this equation independently for θ_1, θ_2, we can easily get the desired trajectory. The simulation uses a "clock" block to feed time to the desired trajectory sub-system. At every time step t, this subsystem outputs desired state variables.

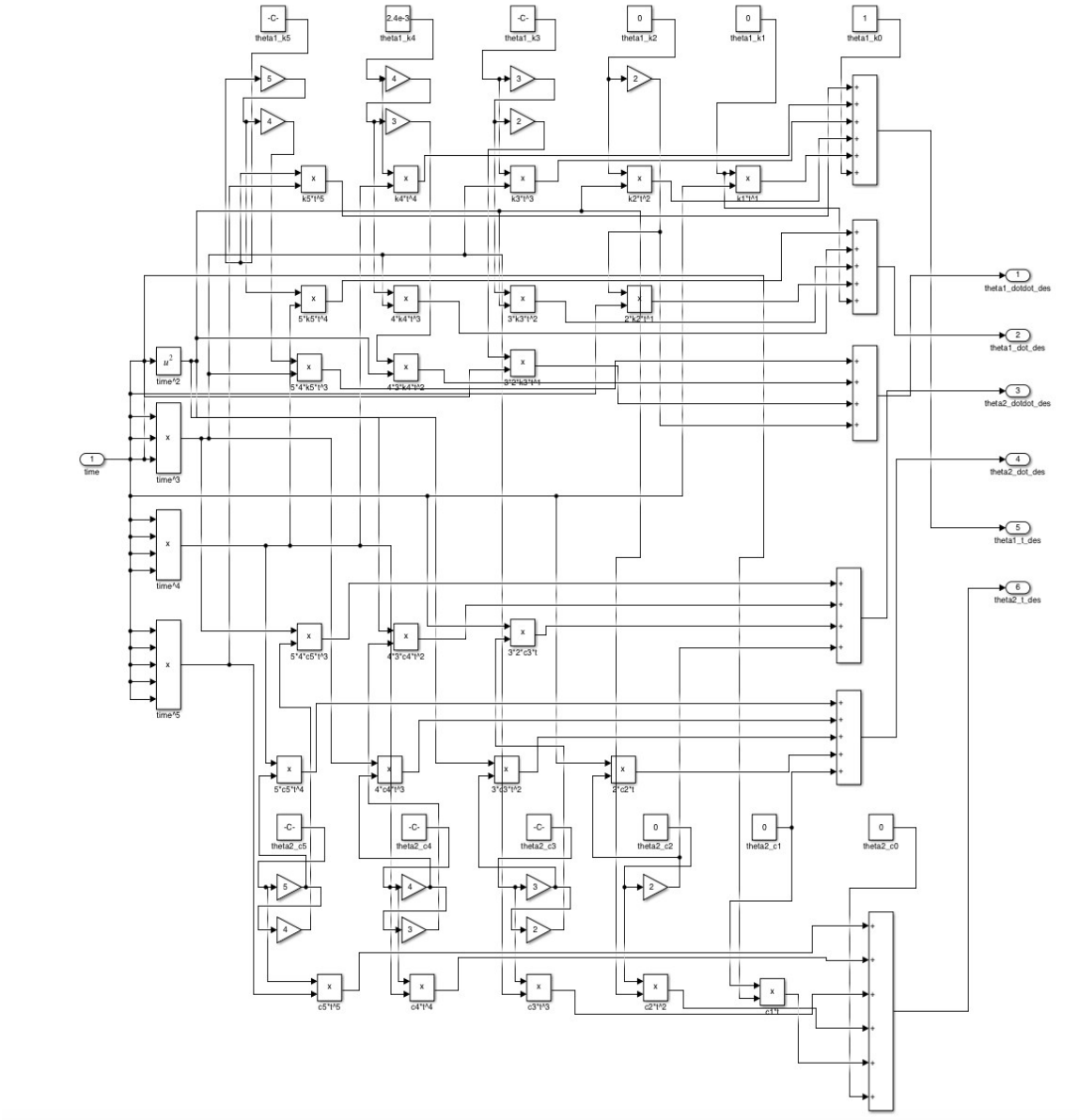

Figure 5: Desired Trajectory sub-system on SIMULINK

B. Auxiliary control subsystem

Input: From Desired trajectory subsystem **:** $\theta_{1d}(t), \theta_{2d}(t), \dot{\theta}_{1d}(t), \dot{\theta}_{2d}(t), \ddot{\theta}_{1d}(t), \ddot{\theta}_{2d}(t)$
Feedback from angular acceleration subsystem: $\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2$
Hyperparameters: ω_1, ω_2

Output: Auxiliary controls $\nu_1 and\ \nu_2$
Auxiliary controls are defined as

$$\begin{bmatrix} \nu_1 \\ \nu_2 \end{bmatrix} = \begin{bmatrix} \ddot{\theta}_{d1} + 2\omega_n \dot{\theta}_{e1} + \omega_n^2 \theta_{e1} \\ \ddot{\theta}_{d2} + 2\omega_n \dot{\theta}_{e2} + \omega_n^2 \theta_{e2} \end{bmatrix} \tag{21}$$

Once we have output from the desired trajectory subsystem, we could use this along with the feedback from the angular acceleration subsystem to find auxiliary controls. In the simulation, the values for $\omega_1 and\ \omega_2$ were tuned to 1 and 10, respectively.

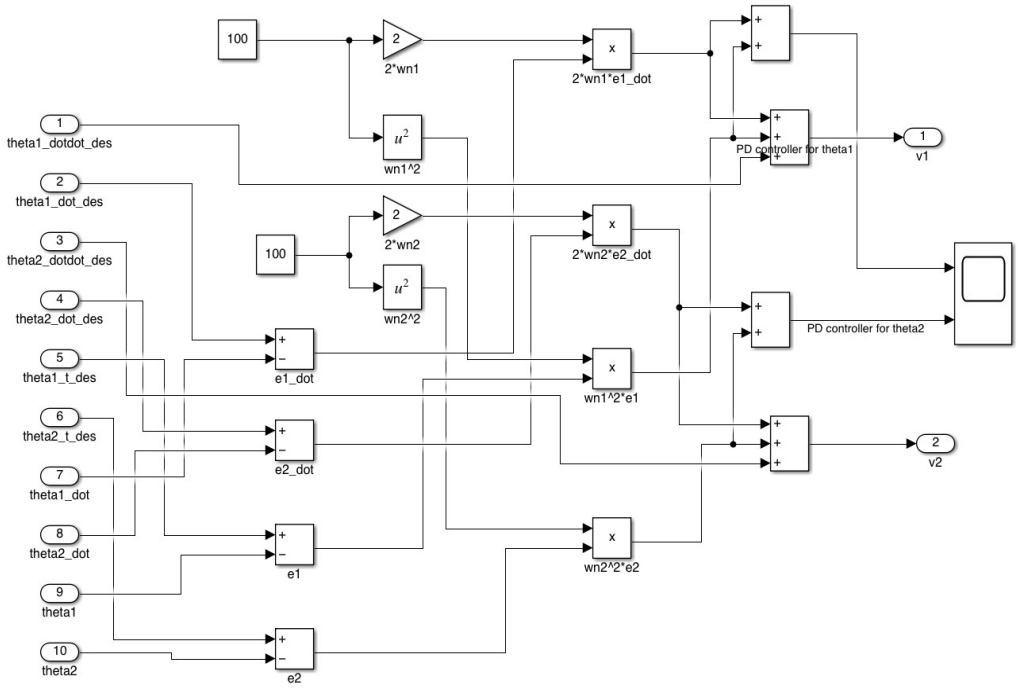

Figure 6: Auxiliary Control subsystem on SIMULINK

C. Control torque subsystem

Input: From Auxiliary control subsystem: $\nu_1 and\ \nu_2$
Feedback from angular acceleration subsystem: $\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2$

Output: Control Torques T_{1c}, T_{2c}

$$\begin{bmatrix} T_{1c} \\ T_{2c} \end{bmatrix} = \begin{bmatrix} I_{11} + I_{21}\cos\theta_2 + I_{22} & I_{21}\cos\theta_2 + I_{22} \\ I_{21}\cos\theta_2 + I_{22} & I_{22} \end{bmatrix} \begin{bmatrix} \nu_1 \\ \nu_2 \end{bmatrix} + I_{21}\sin\theta_2 \begin{bmatrix} \omega_1^2 - \omega_2^2 \\ \omega_1^2 \end{bmatrix} + g \begin{bmatrix} \Gamma_1 \\ \Gamma_2 \end{bmatrix} \quad (22)$$

The control torque subsystem takes the auxiliary controls as input and outputs the control torques using the computed torque controller as discussed in section 2 B. To simplify the diagram, we ignore the $h(\dot{\theta}_d, \theta_d)$ term. This is possible as this term cancels with the identical $h(\dot{\theta}_d, \theta_d)$ term in equations of motion as discussed in section 2 C.

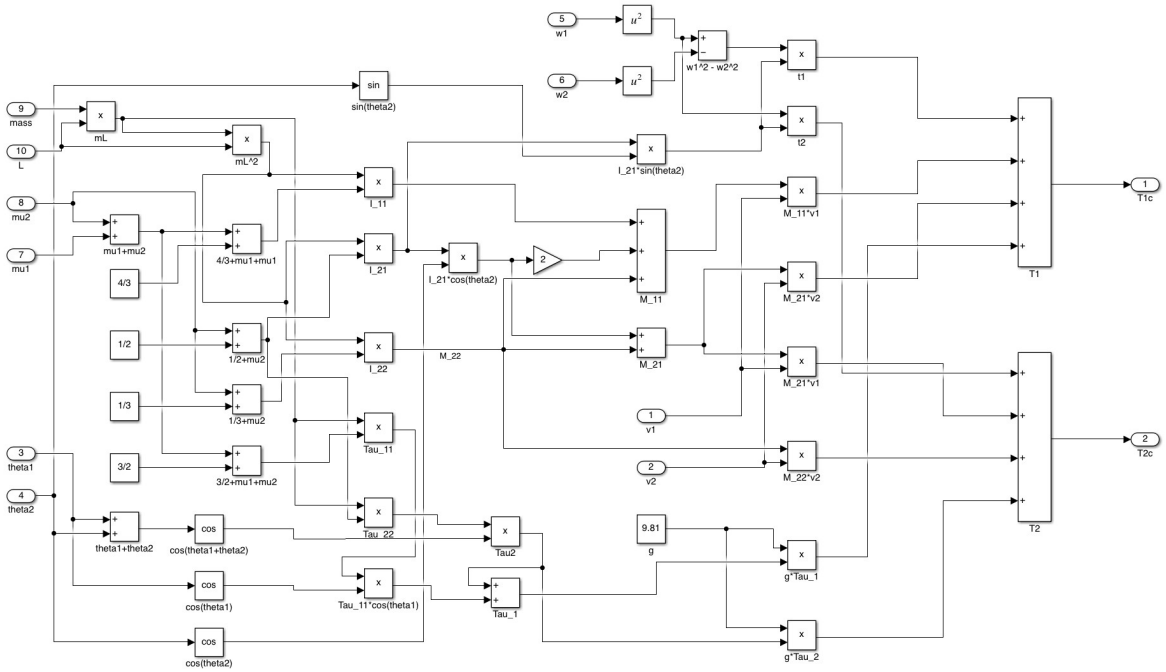

Figure 7: Control torque subsystem on SIMULINK

D. Angular acceleration subsystem

Input: Feedback from angular acceleration subsystem: T_{1c}, T_{2c}

Feedback from angular acceleration subsystem: $\theta_2, \dot{\theta}_1, \dot{\theta}_2$

Output: State-space variables $\theta_{1d}(t), \theta_{2d}(t), \dot{\theta}_{1d}(t), \dot{\theta}_{2d}(t), \ddot{\theta}_{1d}(t), \ddot{\theta}_{2d}(t)$

The angular acceleration subsystem takes the control torques and uses the equation of motion (equation 8) to give out the angular position, velocity and acceleration. This subsystem has a subsystem within it to calculate I_{robot}^{-1}. The state-space variables are then used for feedback to the control loop to calculate trajectory error.

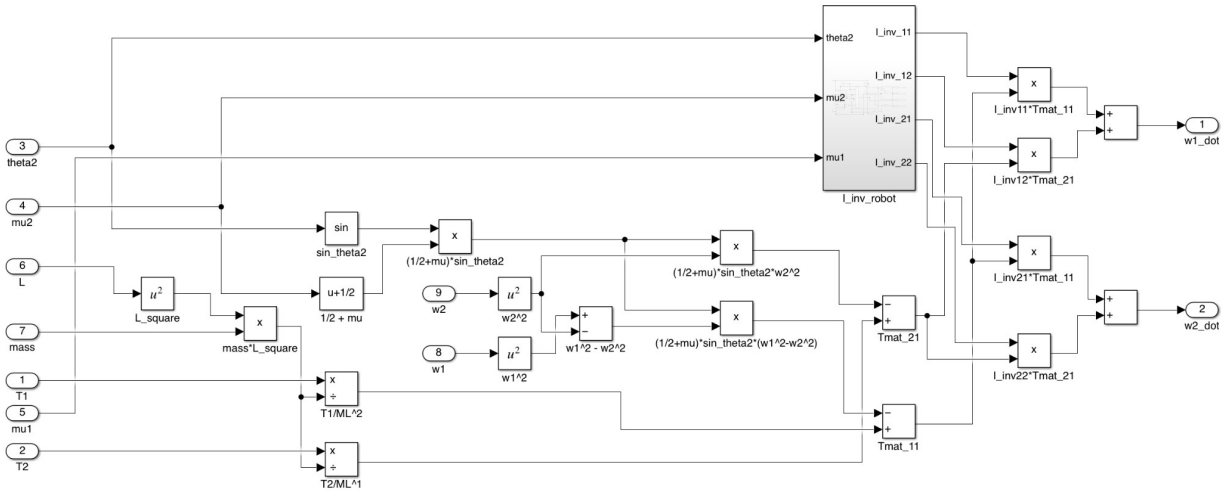

Figure 8: Angular acceleration sub-system on SIMULINK

5. Results and Discussion

As seen in section 2, a computed torque controller converts a complex non-linear control problem into a simple linear controller. With the auxiliary controls defined in equation 21, the trajectory error is guaranteed to reach 0 asymptotically. This section presents results from the MATLAB simulation and discusses them in line with the theory developed.

$L = 1, M = 1,\ \mu_1 = 0.1,\ \mu_2 = 0.1,\ \omega_{n1} = 1\ and\ \omega_{n2} =\ 10$ has been used for all tests unless otherwise stated.

A. Does the end effector reach its desired final position?

To verify if the designed system is working as expected, we feed the following boundary conditions to the trajectory generator as described in section 4.A:

$$\begin{aligned} (\theta_1, \theta_2) &= (1,0) && t = 0\ sec \\ (\theta_1, \theta_2) &= (-0.6,0.8) && t = 10\ \ sec \end{aligned} \tag{23}$$

The following values were obtained for the coefficients in equation 20:

Table 1: Coefficients obtained for minimum jerk trajectory.

	c5	c4	c3	c2	c1	c0
θ_1	-9.60E-05	2.40E-03	-1.60E-02	0	0	1
θ_2	4.80E-05	-1.20E-03	8.00E-03	0	0	0

The figures below show the desired position, velocity and acceleration for θ_1 and θ_2 as obtained from the minimum jerk trajectory generator.

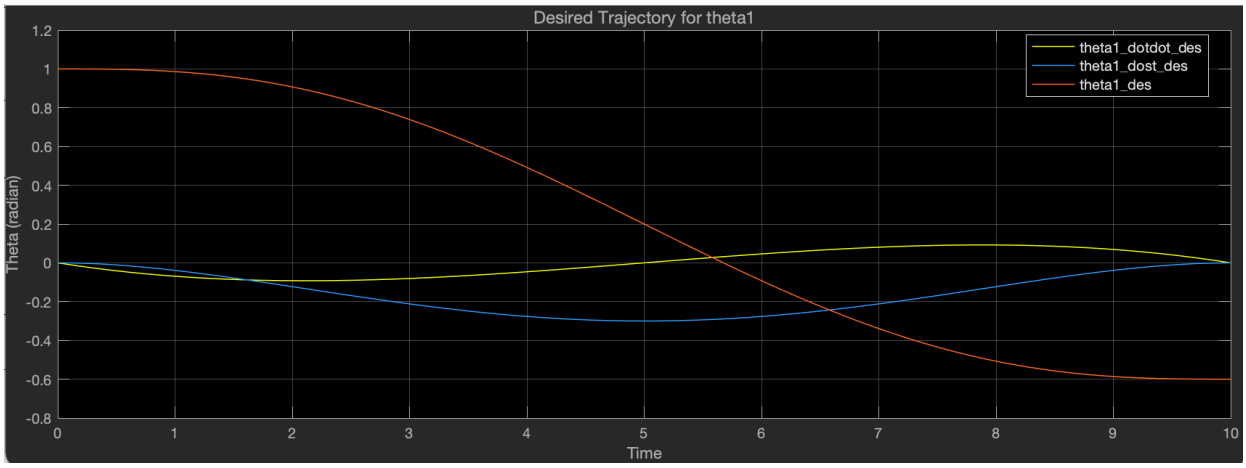

Figure 9: Desired position, velocity and acceleration for θ_1

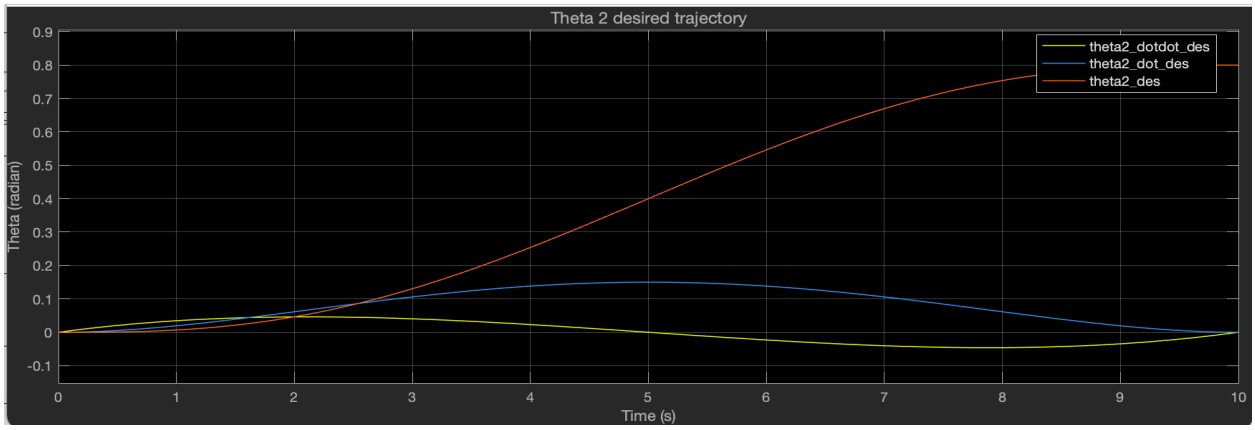

Figure 10: Desired position, velocity and acceleration for θ_2

We then observe the actual trajectory (θ_1, θ_2) followed by the manipulator. Comparing the desired and actual trajectories, we see that these are very close to each other:

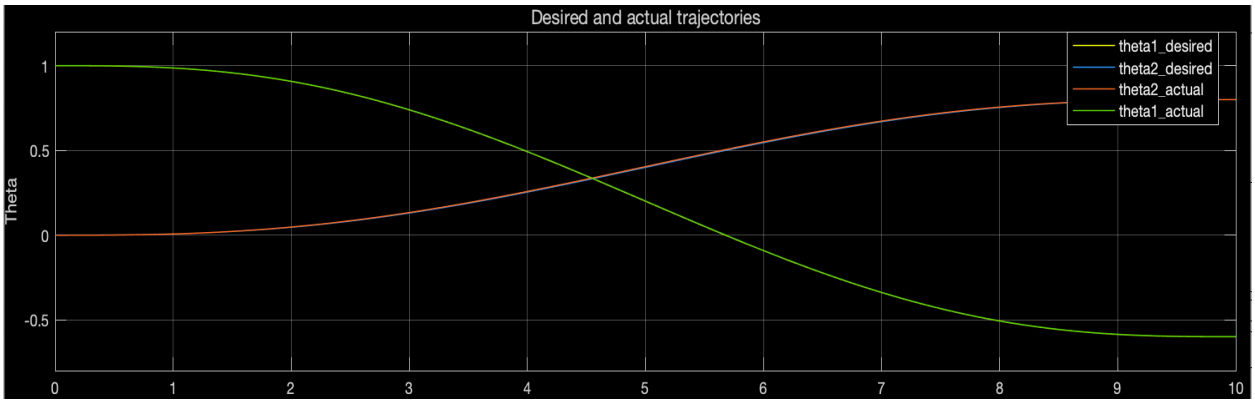

Figure 11: Comparison between desired trajectory values and actual trajectory values (in radians)

Another observation to increase our confidence in the system comes from the control Torque applied. From Fig 7 above and our intuition, we expect the control torque for θ_1 to be negative until $t \approx 4.5s$ and then become positive. For θ_2, we expect an opposite trend as the slope change rate of these curves is opposite in direction.

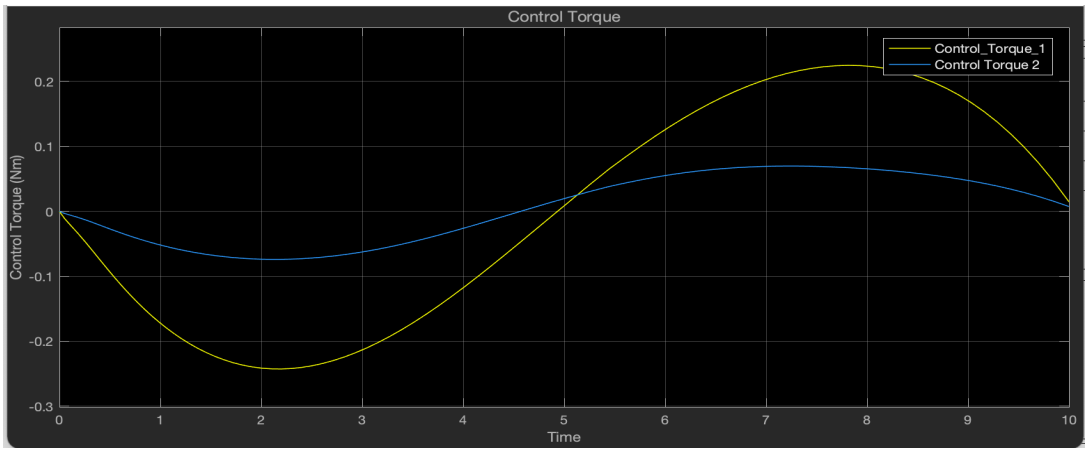

Figure 12: Control torque generated for link 1(yellow) and link 2(blue).

The above figure confirms our intuition. This test shows that the controller is behaving expectedly, and we can reach our desired final configuration with minimal jerks.

B. How does the auxiliary input change with different values of $\mu_1 and\ \mu_2$?

Combining observations from equations 14 and 5.2, we can see that T_{1c} depends on both $\mu_1\ and\ \mu_2$, whereas T_{2c} depends only on μ_2. The equations for control torques take the following form:

$$T_{1c} = \nu_1 M_1(\mu_1, \mu_2) + h_1\left(\dot{\theta}_1, \dot{\theta}_2, \mu_2\right) \quad (24)$$
$$T_{2c} = \nu_2 M_2(\mu_2) + h_2(\dot{\theta}_1, \dot{\theta}_2, \mu_2)$$

Therefore, we expect the auxiliary input curve for ν_2 to be independent of μ_1. This is intuitive as well, once we look at the schematic in figure 1. To verify this from simulation, we test different values of μ_1 and μ_2 and observe the auxiliary inputs. The value of μ_1 and μ_2 tested were:

$$\mu_1 \in \{0, 0.1|$$
$$\mu_2 \in \{0, 0.1, 0.2, 0.5, 1, 2, 10|$$

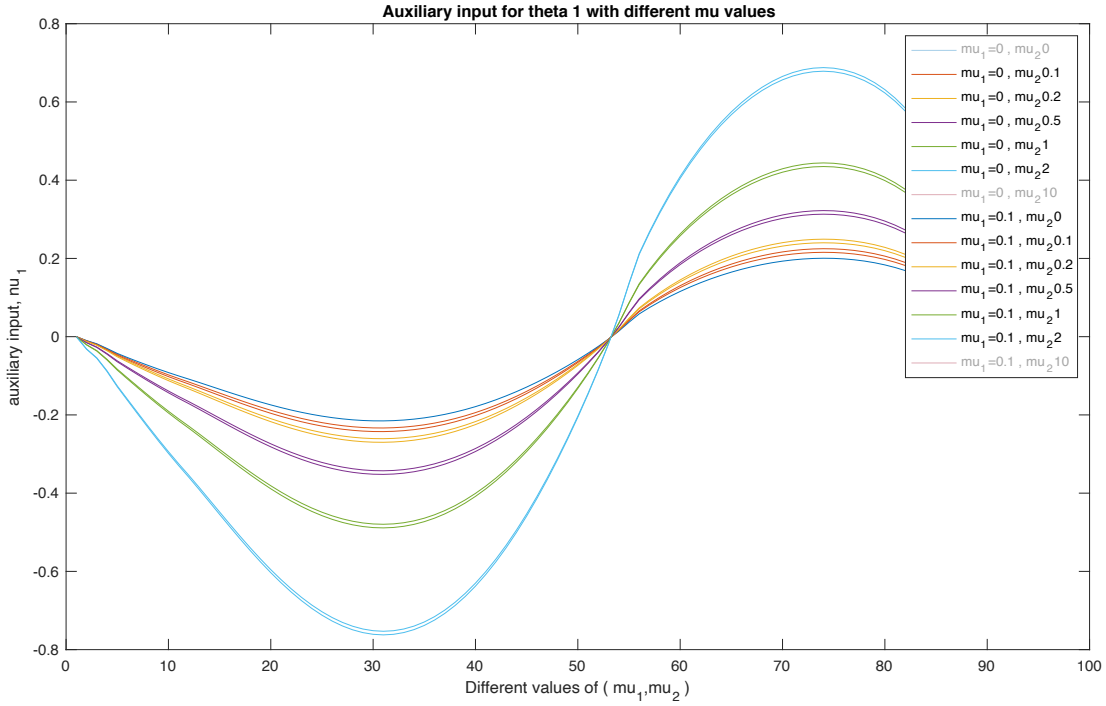

Figure 13: Change in ν_1 with $\mu_1 and\ \mu_2$.

Figure 13 above shows that ν_1 depends on both $\mu_1 and\ \mu_2$. Moreover, the magnitude of ν_1 increases with an increase in $\mu_1 or\ \mu_2$.

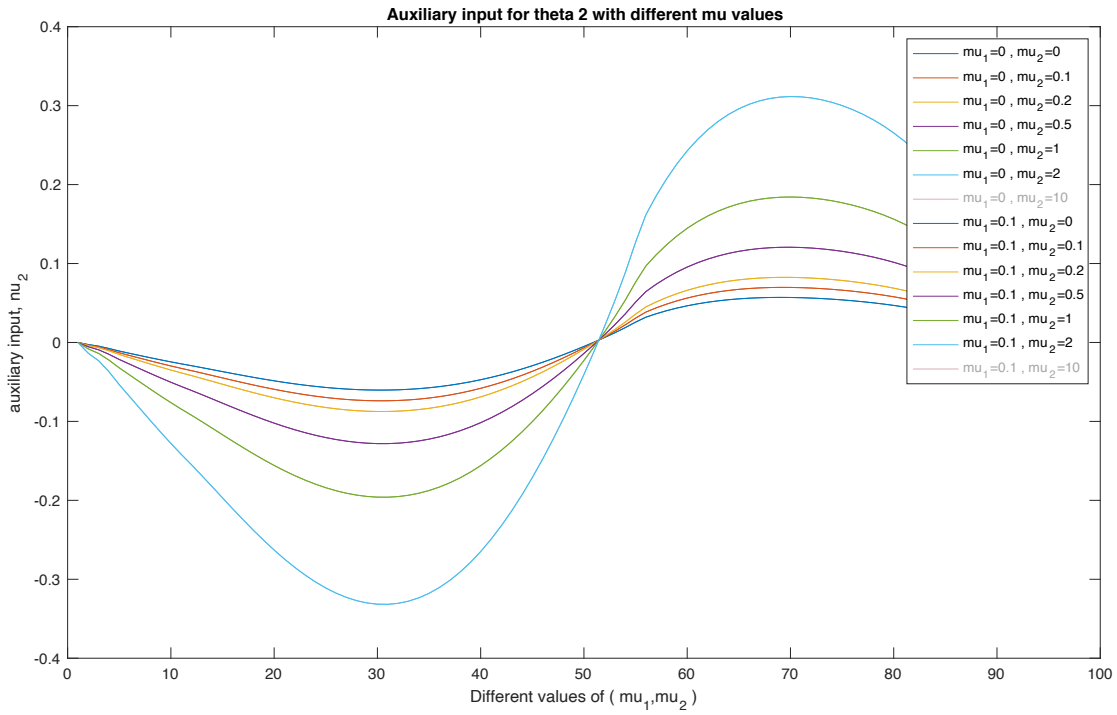

Figure 14: Change in ν_2 with μ_1 and μ_2.

In the above figure, we can see overlapping curves for combinations of $\mu_1 and\ \mu_2$ in which μ_2 is constant. For example, $(\mu_1, \mu_2) = (0, 0.5)$ and $(\mu_1, \mu_2) = (0.1, 0.5)$ have overlapping curves. This shows that ν_2 is independent of μ_1. Also, the magnitude of ν_2 increases with an increase in μ_2.

C. Stability and Robustness

A. How does the system behave in the presence of inaccurate parameters?

If our mathematical model of the system is exact and there are is no error in the values of parameters, the system should be stable on its own and should not require the PD terms. This is so because the desired trajectory ensures zero velocity and acceleration at the final time (t=10 s). Setting $\omega_n = 0$, we see that this is observed in our simulation.

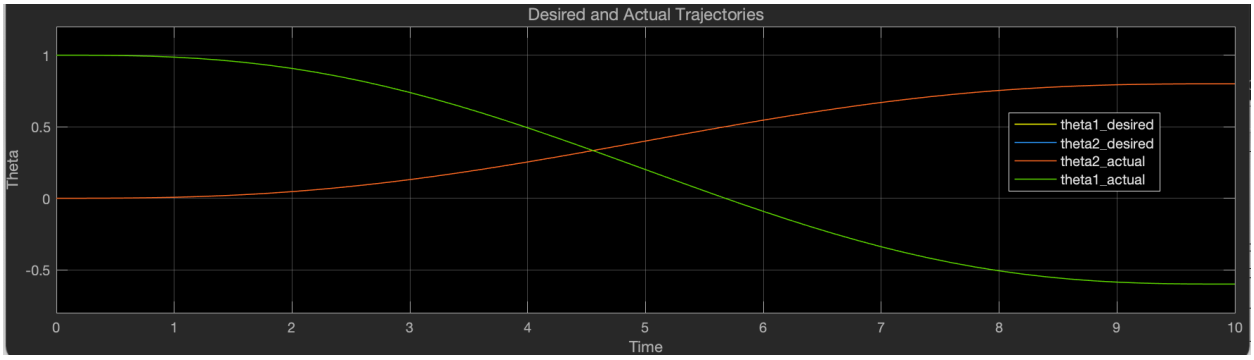

Figure 15: Desired and actual trajectory for $\omega_n = 0$

To test the system's stability in the presence of measurement error, we introduce a discrepancy in the length (L) value for control torque and angular acceleration subsystems. For the control torque subsystem, we give an inaccurate measurement of 0.9 m, while the angular acceleration subsystem has a value of 1.0 m. We compared the desired and actual trajectory for this error of 10% in L. We do the following two tests:

Test Number	M	μ_1	μ_2	ω_1	ω_2
1	1	0.1	0.1	0	0
2	1	0.1	0.1	1	10

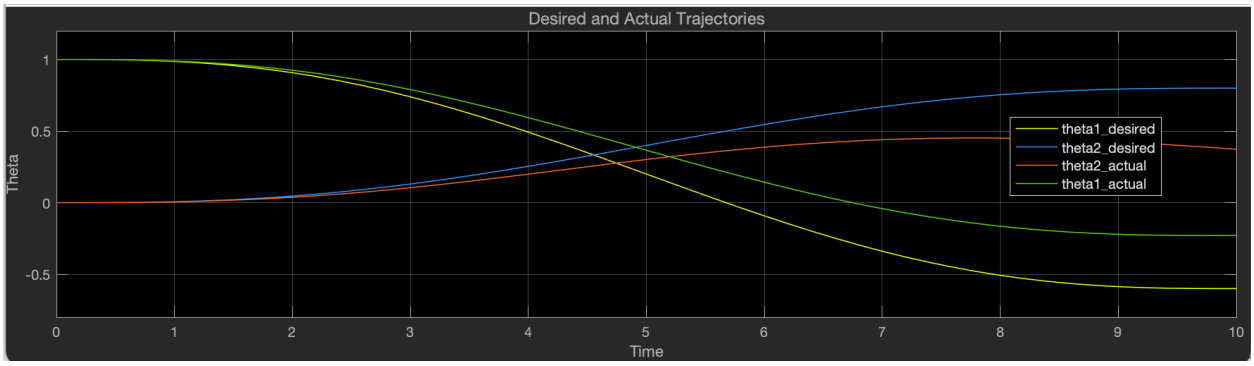

Figure 16: Desired and actual trajectories for test Number 1 (without PD terms)

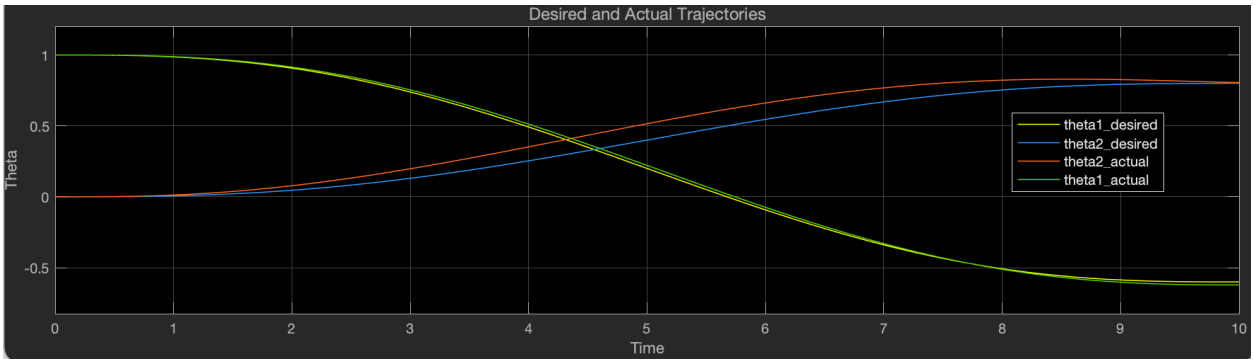

Figure 17: Desired and actual trajectories for test Number 2 (with PD terms)

We can further check the contribution of proportional and derivative terms in the auxiliary input. In the figure below, $PD\ term = 2\omega_n\dot{\theta}_e + \omega_n^2\theta_e$.

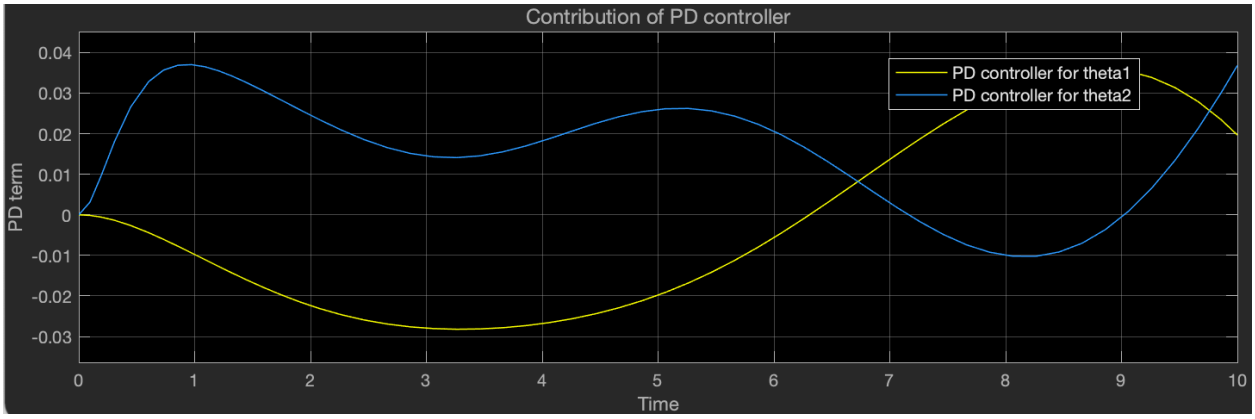

Figure 18: Values of proportional and derivative PD terms for 10% error in L.

This test shows the system's fragility in the absence of the PD terms. In figure 16, the system breaks down after introducing a measurement error of 10% in L. Introducing PD terms generates the corrective torque to keep the system stable (See figure 17 and 18).

Further increasing the error in length measurement to 20%, we observed the auxiliary input then heavily depends on the PD terms, which becomes comparable to $\ddot{\theta}_d(t)$. This can be observed from the plot below which shows the value of these terms for link 1:

Figure 19: Value of PD and $\ddot{\theta}_d(t)$ terms for 20% error in L.

B. Is the system robust to modelling errors?

The mathematical model has to capture the real system precisely for the system to be optimal. When there is a discrepancy between the two, a residual torque remains that drives the trajectory error.

$$\begin{bmatrix} T_1 \\ T_2 \end{bmatrix} = \begin{bmatrix} T_{1c} \\ T_{2c} \end{bmatrix} + \begin{bmatrix} T_1 \\ T_2 \end{bmatrix} - \begin{bmatrix} T_{1c} \\ T_{2c} \end{bmatrix} \equiv \begin{bmatrix} T_{1c} \\ T_{2c} \end{bmatrix} + \begin{bmatrix} \Delta T_1 \\ \Delta T_2 \end{bmatrix}$$

In the above equation, ΔT_1 and ΔT_2 are the residual torques. From equation 15, we can now see that the error dynamics is no longer of the standard-second order form.

$$\begin{bmatrix} \ddot{\theta}_{e1} + 2\omega_n \dot{\theta}_{e1} + \omega_n^2 \theta_{e1} \\ \ddot{\theta}_{e2} + 2\omega_n \dot{\theta}_{e2} + \omega_n^2 \theta_{e1} \end{bmatrix} = \begin{bmatrix} I_{11} + 2I_{21}\cos\theta_2 + I_{22} & I_{21}\cos\theta_2 + I_{22} \\ I_{21}\cos\theta_2 + I_{22} & I_{22} \end{bmatrix}^{-1} \begin{bmatrix} \Delta T_1 \\ \Delta T_2 \end{bmatrix} \quad (15)$$

The above trajectory equation is not guaranteed to be exponentially stable. These modelling errors can arise due to air friction or friction in bearings. We have modelled the end-effector and joints as point masses. In reality, this may not be true and lead to modelling errors. There could be complex reaction forces when the manipulator interacts with an object leading to more modelling errors (Zhang *et al.,* 2021). We can verify this using our simulation by testing different values of ΔT_1 *and* ΔT_2. We will observe θ_{e1} and θ_{e2} for 20 seconds. Remember, our boundary conditions want the system to reach the final state at t = 10 sec.

Tests:

Modelling error	Test 1	Test 2	Test 3
ΔT_1	$-0.1T_{c1}$	$-0.5T_{c1}$	$-0.9T_{c1}$
ΔT_2	$-0.1T_{c1}$	$-0.5T_{c1}$	$-0.9T_{c1}$

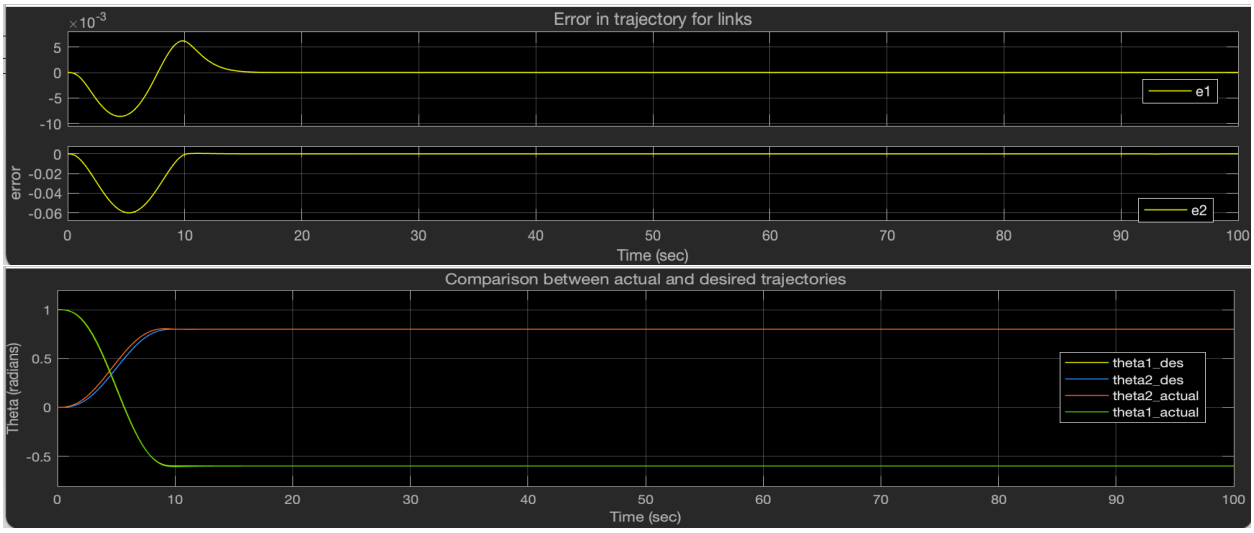

Figure 20: Trajectory errors and comparison between desired and actual trajectory for test 1

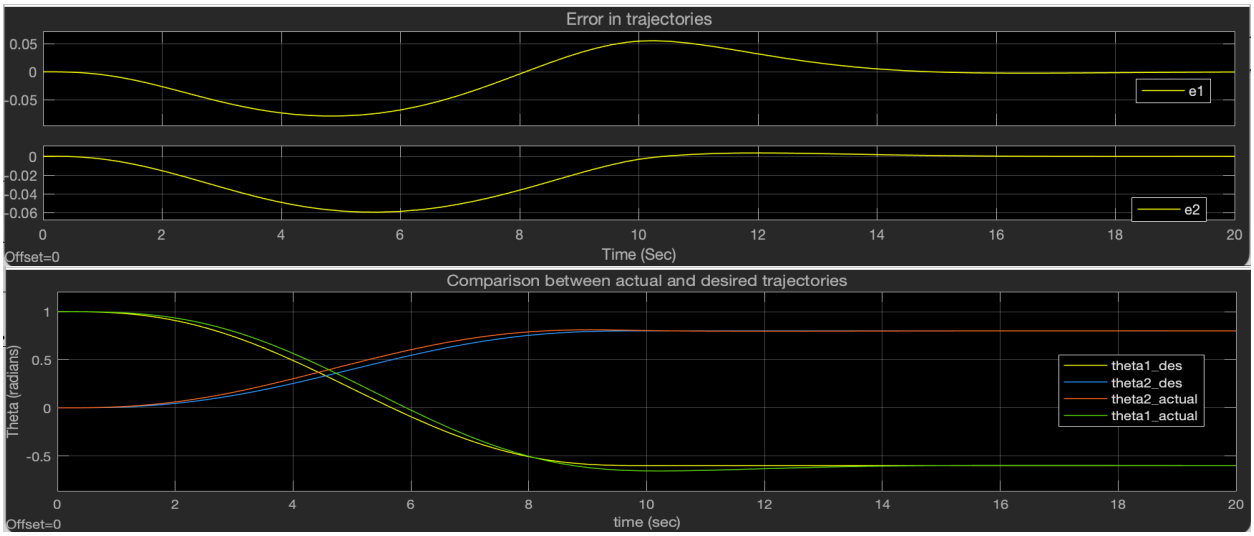

Figure 21: Trajectory errors and comparison between desired and actual trajectory for test 2

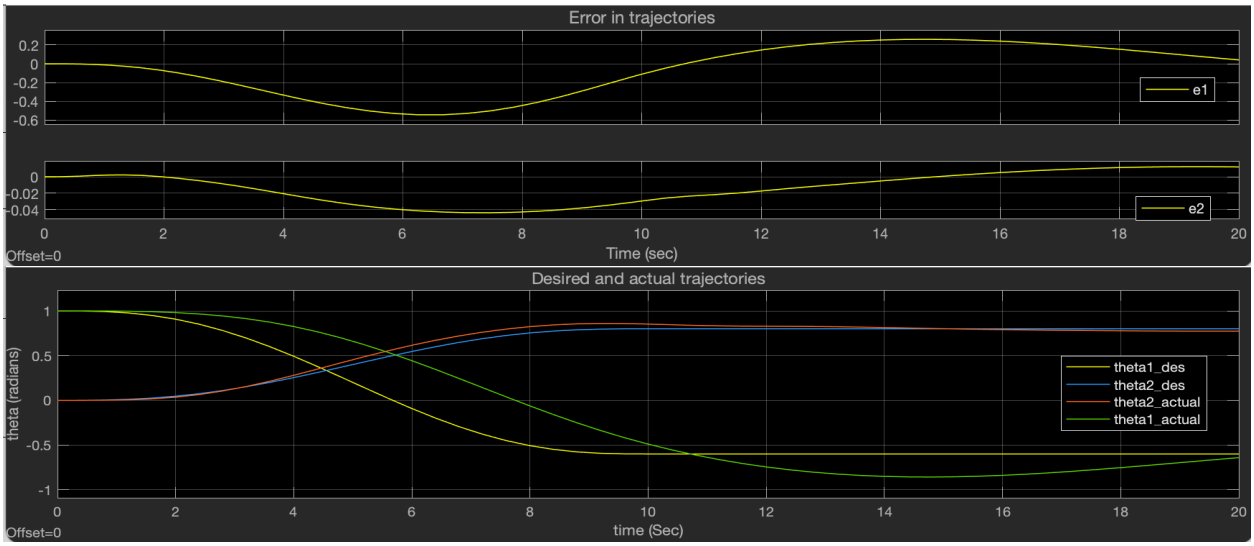

Figure 22: Trajectory errors and comparison between desired and actual trajectory for test 3

In figure 20 and 21, we see that the θ_{e1} and θ_{e2} reach 0 at the end of 20 seconds. Hence, their desired and actual trajectories are also fairly close to each other. In figure 22 we observe that θ_{e2} takes longer to converge to 0 while θ_{e1} keeps oscillating.

The above results show that our system is stable if we can make an accurate mathematical model of the real world and know all the parameters in advance with minimal error. The system is no longer optimal in time-varying or inaccurate system parameters. This is one practical limitation of the designed controller. To get an optimal controller in such settings, we can use adaptive controllers instead of PD (Vepa, 2009). Another solution would be to use Artificial intelligence models like Neural networks to estimate system parameters in real-time. These data-driven approaches could also be used to capture more accurate dynamic equations of motion of the system.

C. Maximum values

a. What is the maximum change in angle per time step?

b. What is the maximum Control Torque?

Figure 11 shows that link 1 rotates through 85.9 degrees counter-clockwise while link 2 rotates 45.8 degrees clockwise. There are limitations on the torque generated at a joint or the maximum rotation that can be achieved per time step in real life. To check these values for the simulation, we record the maximum control torque generated at joint one and joint two and the maximum change in θ_1 and θ_2 per time step.

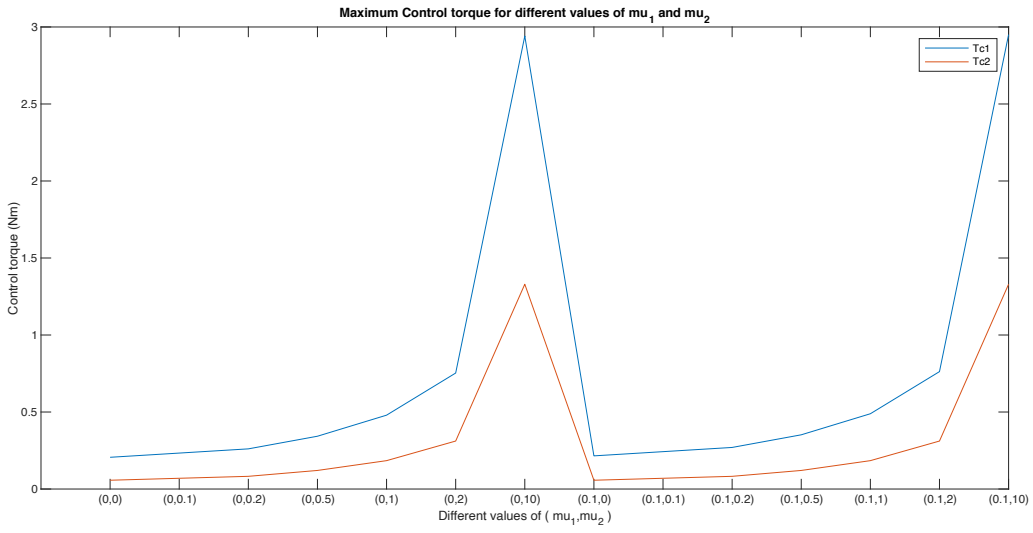

Figure 23: Maximum control torques for different values of ν_1 and ν_2

The above figure shows that the maximum torque required is less than 3 Nm for all combinations of μ_1 and μ_2 tested.

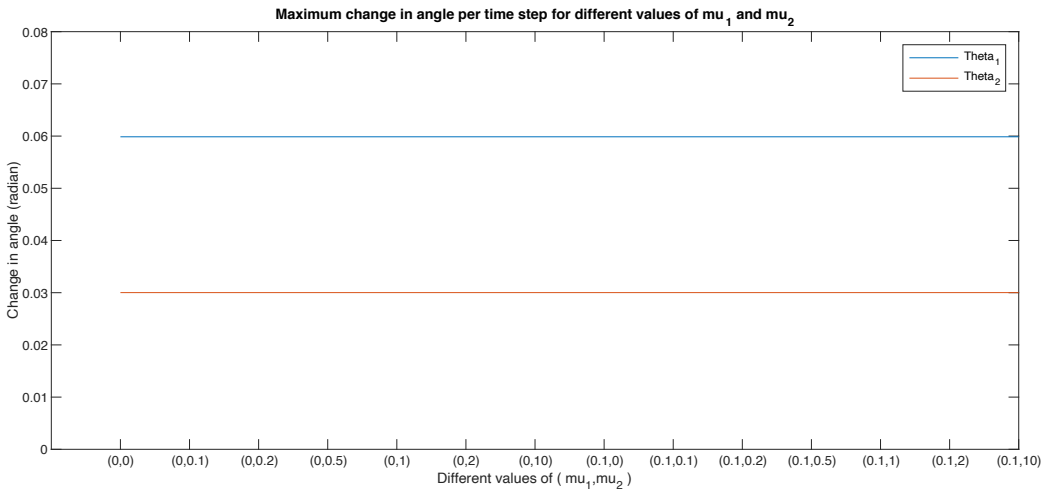

Figure 24: Maximum change in angular position for different values of ν_1 and ν_2

The maximum change in θ_1 *and* θ_2 per time step is 3.43 and 1.71 degrees, respectively. This is the same for all combinations of μ1 and μ2 since our system achieved the desired trajectory for all cases.

D. Impact of Future technologies on the design of robot controllers

Recent advances in artificial intelligence have made it possible to estimate uncertainty in robot dynamics using methods like reinforcement learning. Such models can be used to estimate uncertain dynamics given a prior dynamics model (Lucas *et al.,* 2021). This strategy can be used in combination with existing controllers to improve their accuracy and robustness.

Some controllers have implicit parameters that need to be defined. For example, the impendence controller requires stiffness and damping matrices. Contact rich tasks require these matrices to be set dynamically depending on the situation and object the manipulator is interacting with. Zhang *et al.* (2021) used Inverse reinforcement learning to set these parameters using expert demonstrations.

Modern controllers have started to use multi-modal sensor input to build adaptable controllers. Lee *et al.* (2020) used feedback from vision and haptic to accomplish the peg-in-hole assembly task. Using both vision and touch feedback in a controller brings it closer to how humans function.

Although data-driven approaches work better when modelling is uncertain, and system parameters may not be precise, they cannot usually on their own guarantee safety and asymptotic stability (Lukas *et al.,* 2021). Future research is directed in developing theories to give formal guarantee for these data-driven approaches or use them in combination with analytical ones.

6. Conclusion

Robot control and manipulation have reached high levels of autonomy due to recent advancements in artificial intelligence and its combination with analytical control methods. This exercise looked at computed torque controller, an analytical control method that converts a complicated non-linear control problem into a linear system. The computed torque control was applied to a two-link andromorphic system after deriving its equations of motion using the Euler-Lagrange formulation. The controller was successfully tested in MATLAB Simulink, where the trajectory between initial and final states was obtained using the minimum jerk formulation. The controller was tested for its stability and robustness by introducing modelling errors and inaccurate parameters. In the setup introduced, we obtained low tracking errors even after introducing

50% error in control torque and 20% error in Length measurements beyond which the tracking error was significant and system was unable to reach the desired configuration. Methods to handle larger uncertainty were also discussed. The highest torque generated at joints and maximum change in angle per time step were also analysed for different masses of joint and end-effector. The study found both the torque required and maximum angular displacement to be within reasonable practical limits. Obtaining positive results in this initial study motivates further research on using adaptive or data-driven approaches in combination with the computed torque controller.

7. References

Chung, W., Fu, L.-C. and Hsu, S.-H. (2008). Motion Control. *Springer Handbook of Robotics*, pp.133–159.

Lee, M.A., Zhu, Y., Zachares, P., Tan, M., Srinivasan, K., Savarese, S., Fei-Fei, L., Garg, A. and Bohg, J. (2020). Making Sense of Vision and Touch: Learning Multimodal Representations for Contact-Rich Tasks. *IEEE Transactions on Robotics*, 36(3), pp.582–596.

Lukas Brunke, Melissa Greeff, Adam W. Hall, Zhaocong Yuan, Siqi Zhou, Jacopo Panerati, & Angela P. Schoellig. (2021). Safe Learning in Robotics: From Learning-Based Control to Safe Reinforcement Learning.

Lynch, K.M. and Park, F.C. (2017). Robot control. *Modern robotics : mechanics, planning, and control*. Cambridge: University Press, pp.403-448

Sharkawy, A.-N. (2021) "Minimum Jerk Trajectory Generation for Straight and Curved Movements: Mathematical Analysis", CoRR, abs/2102.07459. Available at: https://arxiv.org/abs/2102.07459.

Vepa, R. (n.d.). Hamiltonian Systems and Feedback Linearization. *Biomimetic Robotics*, pp.198–241

Zhang, X., Sun, L., Kuang, Z. and Tomizuka, M. (2021). Learning Variable Impedance Control via Inverse Reinforcement Learning for Force-Related Tasks. *IEEE Robotics and Automation Letters*, pp.1–1.