Part 1:

Second order inverse differential kinematics algorithm has been used to solve the given problem.
The following is the differential kinematics equation defined in second order,

$$\boxed{\ddot{x_e} = J_A(q)\ddot{q} + \dot{J}_A(q,\dot{q})\dot{q}}$$

We know,

$$\boxed{J_A(q) = \begin{bmatrix} -0.5(S_{\theta 12} + S_{\theta 1}) & -0.5S_{\theta 12} & 0 & 0 \\ 0.5(C_{\theta 12} + C_{\theta 1}) & 0.5C_{\theta 12} & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 1 & 1 & 0 & 1 \end{bmatrix}}$$

$\dot{J}_A$ in the above equation is time derivative of the Jacobian.

$$\boxed{\dot{J}_A(q,\dot{q}) = \begin{bmatrix} -0.5(c_{\theta 12}\dot{\theta}_{12} + c_{\theta 1}\dot{\theta}_1) & -0.5(c_{\theta 12}\dot{\theta}_{12}) & 0 & 0 \\ -0.5(s_{\theta 12}\dot{\theta}_{12} + s_{\theta 1}\dot{\theta}_1) & -0.5(s_{\theta 12}\dot{\theta}_{12}) & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}}$$

The error in the second order can be deduced as,

$$\boxed{\ddot{e} = \ddot{x_d} - J_A(q)\ddot{q} - \dot{J}_A(q,\dot{q})\dot{q}}$$

From here the inverse kinematics can be performed using the equation,

$$\boxed{\ddot{q} = J_A^{-1}(\ddot{x}_d + K_D\dot{e} + K_P e - \dot{J}_A(q,\dot{q})\dot{q})}$$

The major difference of the second order algorithm compared to the first order is that here, the second order error is converged to zero. This allows us to inverse the end-effector position, velocity, as well as its acceleration which is required in the dynamics equations. Also since the manipulator is inherently a second order mechanical system, it is required to perform inversion in second order.

After performing the above equation in Simulink, following results are obtained,

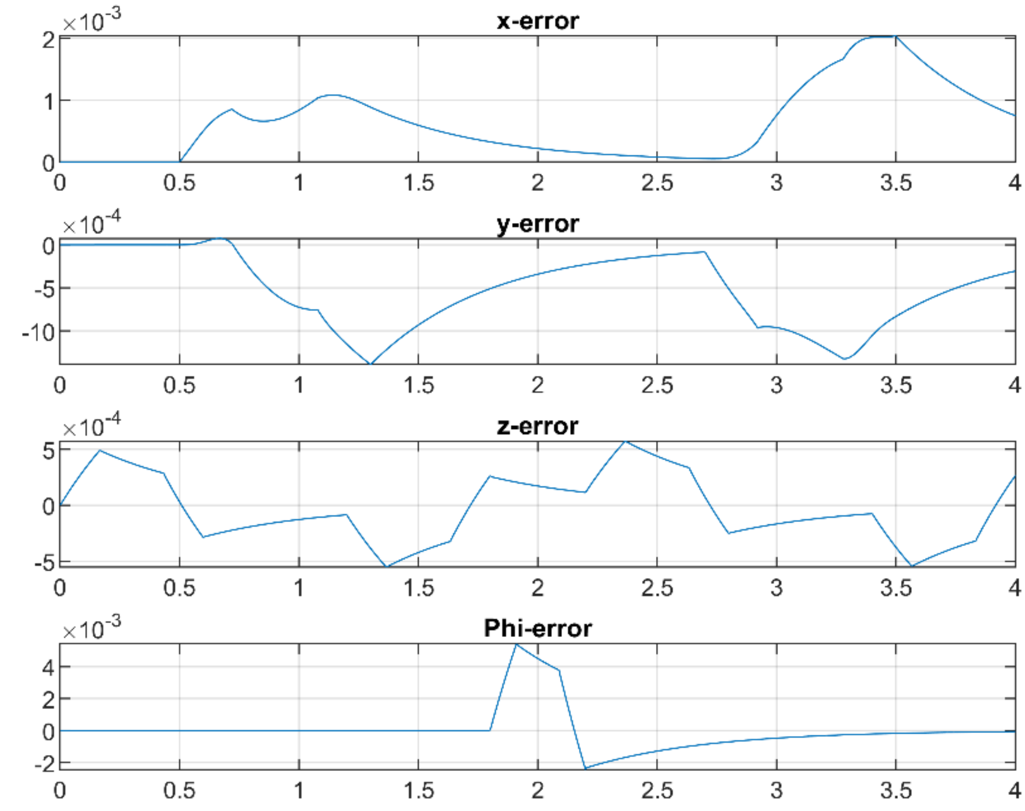

Figure 1: Position Error in Cartesian space

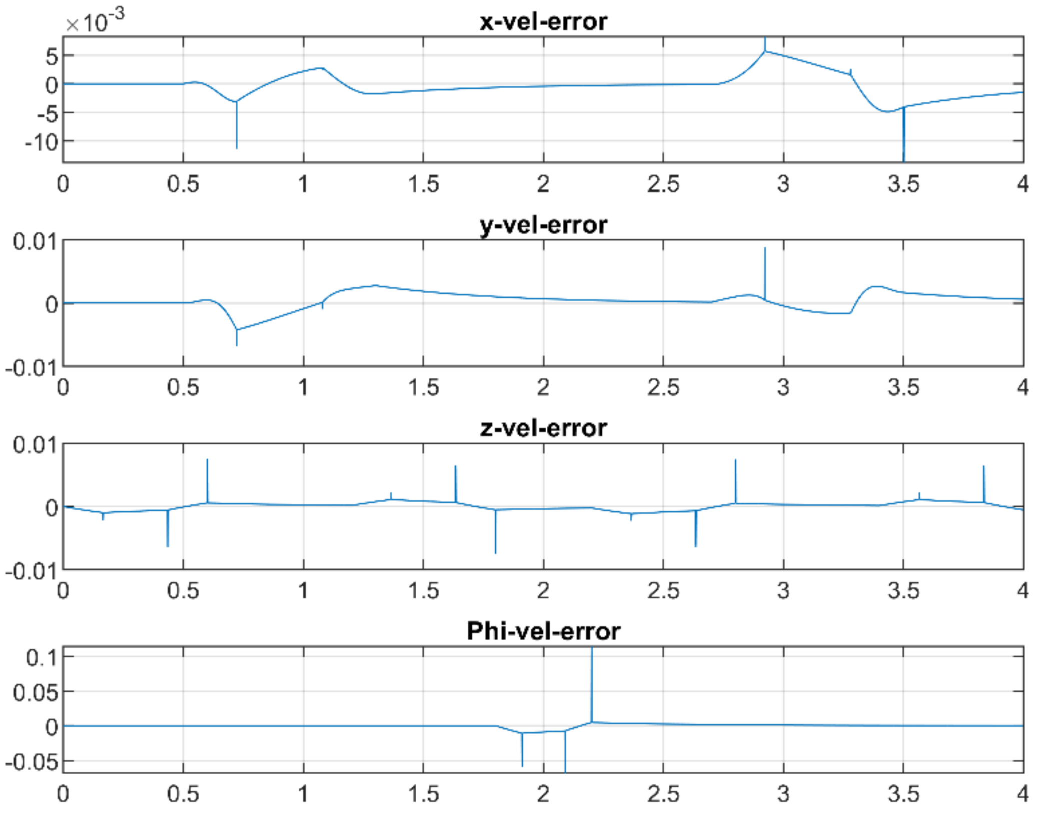

Figure 2: Velocity error in Cartesian space

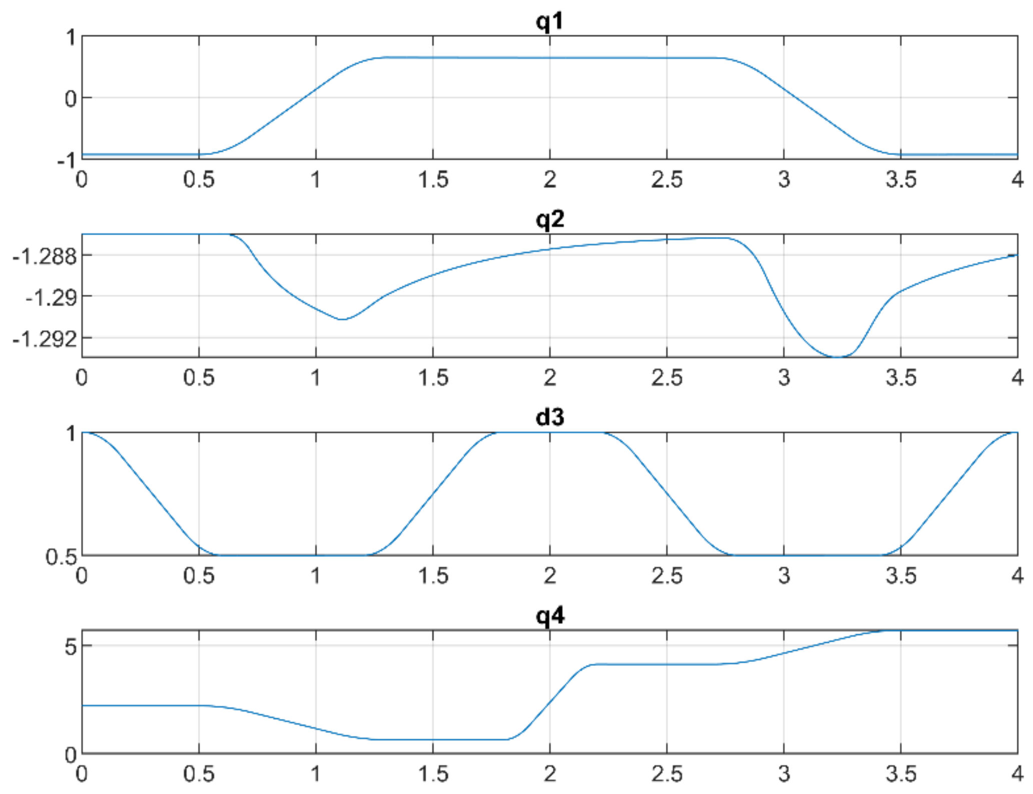

Figure 3: Joint Variables

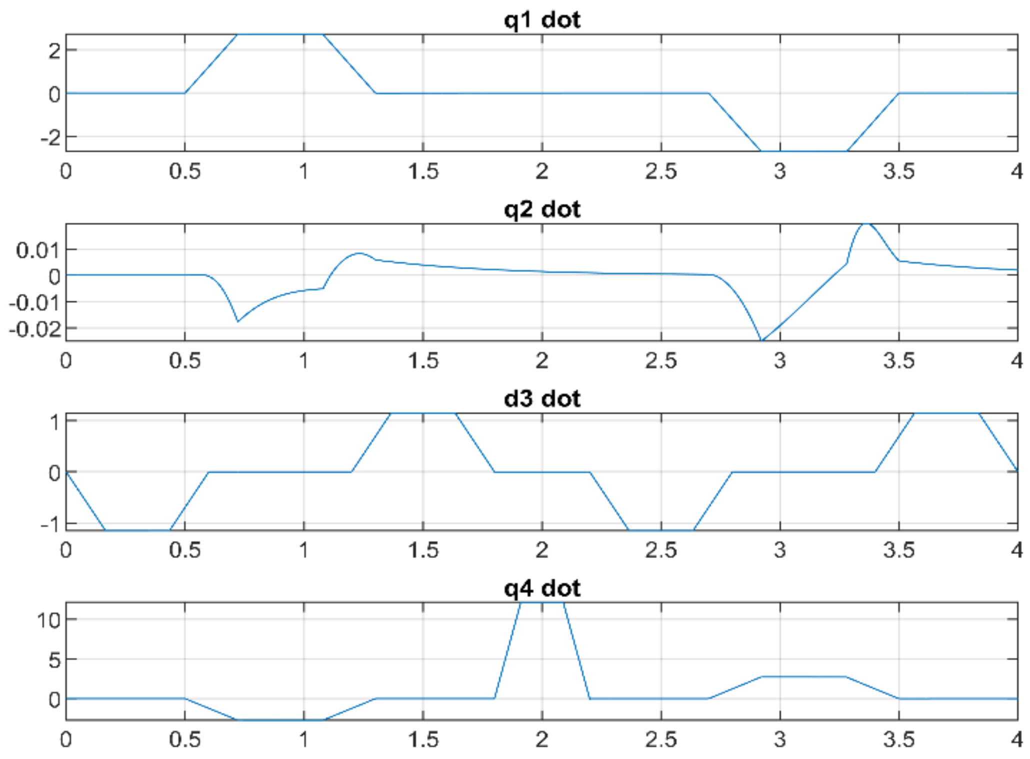

Figure 4: Joint Variables second order

Part 2:

Dynamic model of any manipulator is,

$$\boxed{u = B(q) * \ddot{q} + n(q, \dot{q})}$$

Where,

$$\boxed{n(q, \dot{q}) = (C + F_v) * \dot{q} + g(q)}$$

Here u is the set of motor torques B and n can be deduced as follows,

$$\boxed{\begin{aligned} B(\theta) = \sum_{i=1}^{n} (m_{l_i} * J_p^{(l_i)T} * J_p^{(l_i)} + J_O^{(l_i)T} * R_i * I_{li}^{i} * R_i^T * J_O^{(l_i)} + m_{m_i} * J_p^{(m_i)T} * J_p^{(m_i)} + J_O^{(m_i)T} * R_{m_i} \\ * I_{m_i}^{m_i} * R_{m_i}^T * J_O^{(m_i)}) \end{aligned}}$$

All the required Jacobians of links and motors can be written as,

$$J_p^{l_1} = \begin{bmatrix} -l_1 Sin(\theta_1) & 0 & 0 & 0 \\ l_1 Cos(\theta_1) & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, , J_p^{l_2} = \begin{bmatrix} -a_1 Sin(\theta_1) - l_2 Sin(\theta_{12}) & -l_1 Sin(\theta_{12}) & 0 & 0 \\ a_1 Cos(\theta_1) + l_2 Cos(\theta_{12}) & l_2 Cos(\theta_{12}) & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$J_p^{l_3} = \begin{bmatrix} -a_1 Sin(\theta_1) - a_2 Sin(\theta_{12}) & -a_2 Sin(\theta_{12}) & 0 & 0 \\ a_1 Cos(\theta_1) + a_2 Cos(\theta_{12}) & a_2 Cos(\theta_{12}) & 0 & 0 \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

$$J_p^{l_4} = \begin{bmatrix} -a_1 Sin(\theta_1) - a_2 Sin(\theta_{12}) & -a_2 Sin(\theta_{12}) & 0 & 0 \\ a_1 Cos(\theta_1) + a_2 Cos(\theta_{12}) & a_2 Cos(\theta_{12}) & 0 & 0 \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

$$J_0^{l_1} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}, J_0^{l_2} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix}, J_0^{l_3} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix}, J_0^{l_4} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 \end{bmatrix}$$

$$J_0^{m_1} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ k_{r_1} & 0 & 0 & 0 \end{bmatrix}, J_0^{m_2} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & k_{r_2} & 0 & 0 \end{bmatrix}, J_0^{m_3} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & -k_{r_3} & 0 \end{bmatrix},$$

$$J_0^{m_4} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & k_{r_4} \end{bmatrix}$$

By substituting these Jacobians in the above equations gives the B as follows,

$$B_{11} = I_{l_1} + (m_{l_1} * l_1^2) + (k_{r_1}^2 * I_{m_1}) + I_{l_2} + m_{l_2} * \left(a_1^2 + l_2^2 + 2 * a_1 * l_2 * C(\theta_2)\right) + I_{m_2} + m_{l_3} * \left(a_1^2 + a_2^2 + 2 * a_1 * a_2 * C(\theta_2)\right) + I_{l_4} + I_{m_3} + I_{m_4} + m_{l_4} * (a_1^2 + a_2^2 + 2 * a_1 * a_2 * C(\theta_2))$$

$$B_{21} = B_{12} = I_{l_2} + m_{l_2} * \left(l_2^2 + a_1 * l_2 * C(\theta_2)\right) + k_{r_2} * I_{m_2} + m_{l_3} * \left(a_2^2 + a_1 * a_2 * C(\theta_2)\right) + I_{l_4} + I_{m_3} + I_{m_4} + m_{l_3} * \left(a_1^2 + a_2^2 + 2 * a_1 * a_2 * C(\theta_2)\right)$$

$$B_{13} = -I_{m_3} * k_{r_3}$$

$$B_{14} = I_{l_4} + I_{m_4} * k_{r_4}$$

$$B_{22} = I_{l_2} + m_{l_3} * l_2^2 + \left(k_{r_2}\right)^2 * I_{m_2} + m_{l_3} * a_2^2 + I_{l_4} + I_{m_3} + I_{m_4} + m_{l_4} * a_2^2$$

$$B_{23} = -I_{m_3} * k_{r_3}$$

$$B_{24} = I_{l_4} + I_{m_4} * k_{r_4}$$

$$B_{31} = -I_{m_3} * k_{r_3}$$

$$B_{32} = -k_{r_3} * I_{m_3}$$

$$B_{33} = m_{l_3} + m_{l_4} + I_{m_3} * {k_{r_3}}^2$$

$$B_{34} = 0$$

$$B_{41} = I_{l_4} + I_{m_4} * k_{r_4}$$

$$B_{42} = I_{l_4} + I_{m_4} * k_{r_4}$$

$$B_{43} = 0$$

$$B_{44} = I_{l_4} + I_{m_4} * {k_{r_4}}^2$$

So all the elements of B matrix have been generated.

$$\boxed{n(q, \dot{q}) = \dot{B}(q)\dot{q} - \frac{1}{2}\left(\frac{\delta}{\delta q}\dot{q}^T B(q)\dot{q}\right)^T + \left(\frac{\delta \mathrm{U}(q)}{\delta q}\right)^T + F_v * \dot{q}}$$

From the above equation C can be obtained as,

$$C_{11} = (m_{l_2} * a_1 * l_2 * S(\theta_2) + m_{l_3} * a_1 * a_2 * S(\theta_2) + m_{l_4} * a_1 * a_2 * S(\theta_2)) * -\dot{\theta}_2$$

$$C_{12} = (m_{l_2} * a_1 * l_2 * S(\theta_2) + m_{l_3} * a_1 * a_2 * S(\theta_2) + m_{l_4} * a_1 * a_2 * S(\theta_2)) * -(\dot{\theta}_1 + \dot{\theta}_2)$$

$$C_{21} = (m_{l_2} * a_1 * l_2 * S(\theta_2) + m_{l_3} * a_1 * a_2 * S(\theta_2) + m_{l_4} * a_1 * a_2 * S(\theta_2)) * \dot{\theta}_1$$

The rest of the terms of C matrix are null.

$$F = \begin{bmatrix} F_{m_1} & 0 & 0 & 0 \\ 0 & F_{m_2} & 0 & 0 \\ 0 & 0 & F_{m_3} & 0 \\ 0 & 0 & 0 & F_{m_4} \end{bmatrix} = \begin{bmatrix} 0.0001 & 0 & 0 & 0 \\ 0 & 0.0001 & 0 & 0 \\ 0 & 0 & 0.01 & 0 \\ 0 & 0 & 0 & 0.005 \end{bmatrix}$$

$$g(q) = \begin{bmatrix} 0 \\ 0 \\ 5 * g \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 49 \\ 0 \end{bmatrix}$$

By plugging in all the known constants into the B and C matrices we get,

$$B = \begin{bmatrix} 13.75 * C(\theta_2) + 27.8902 & 8.125 * C(\theta_2) & -0.5 & 1.1 \\ 8.125 * C(\theta_2) + 12.5776 & 11.3276 & -0.5 & 1.1 \\ -0.5 & -0.5 & 40 & 0 \\ 1.1 & 1.1 & 0 & 3 \end{bmatrix}$$

$$C = \begin{bmatrix} -6.875 * \dot{\theta}_2 * S(\theta_2) & -6.875 * S(\theta_2)(\dot{\theta}_1 + \dot{\theta}_2) & 0 & 0 \\ 6.875 * \dot{\theta}_1 * S(\theta_2) & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

From these equations n can be calculated as,

$$\boxed{n(q, \dot{q}) = (C + F_v) * \dot{q}}$$

For Inverse dynamic control we solve the following equation,

$$\boxed{\ddot{q} = B^{-1}(q)(\tau - \tau^1)}$$

Here,

$$\boxed{y = \ddot{\theta}_d + K_D * \dot{\tilde{q}} + K_P * \dot{\tilde{q}}}$$

$$\boxed{\tau^1 = n(q, \dot{q})}$$

Using all the above equations, we get the following results in the Simulink

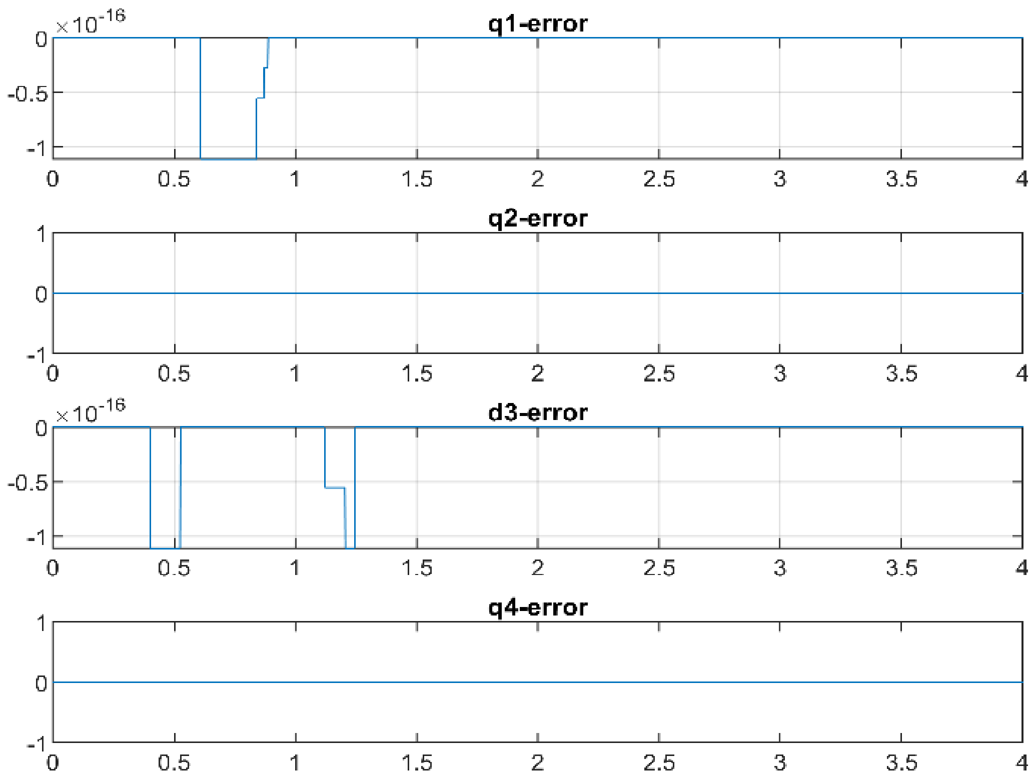

Figure 5: Error in first order joint variables

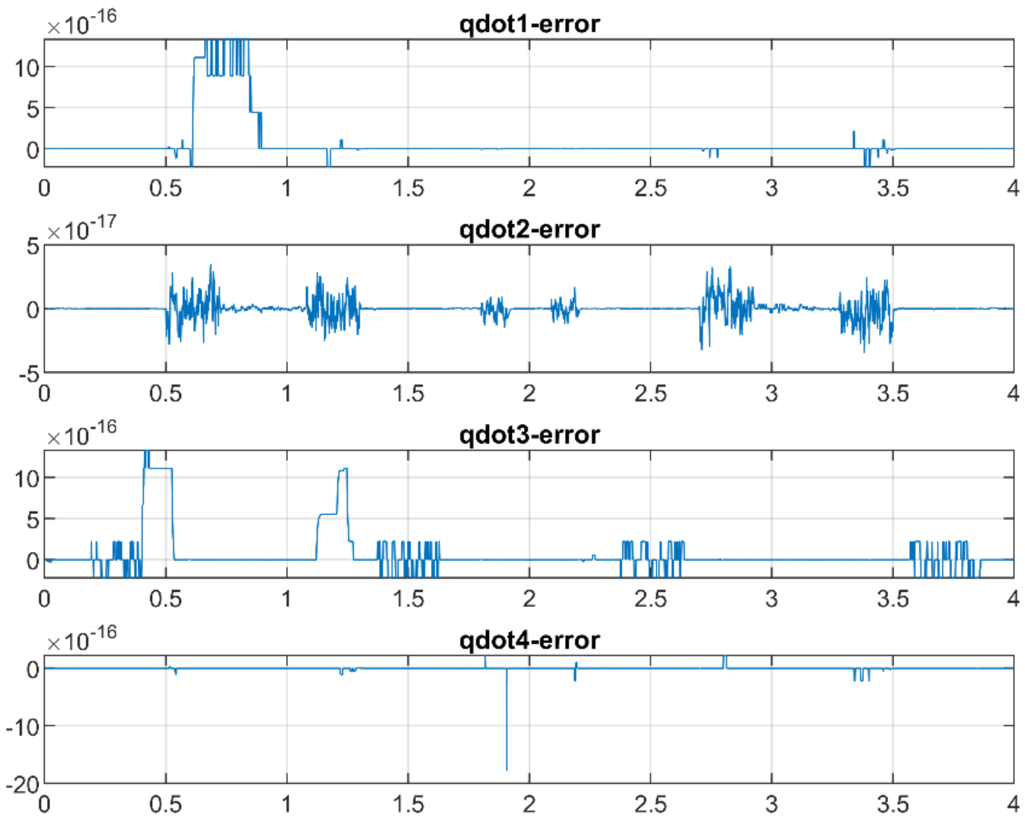

Figure 6: Error in second order joint variables

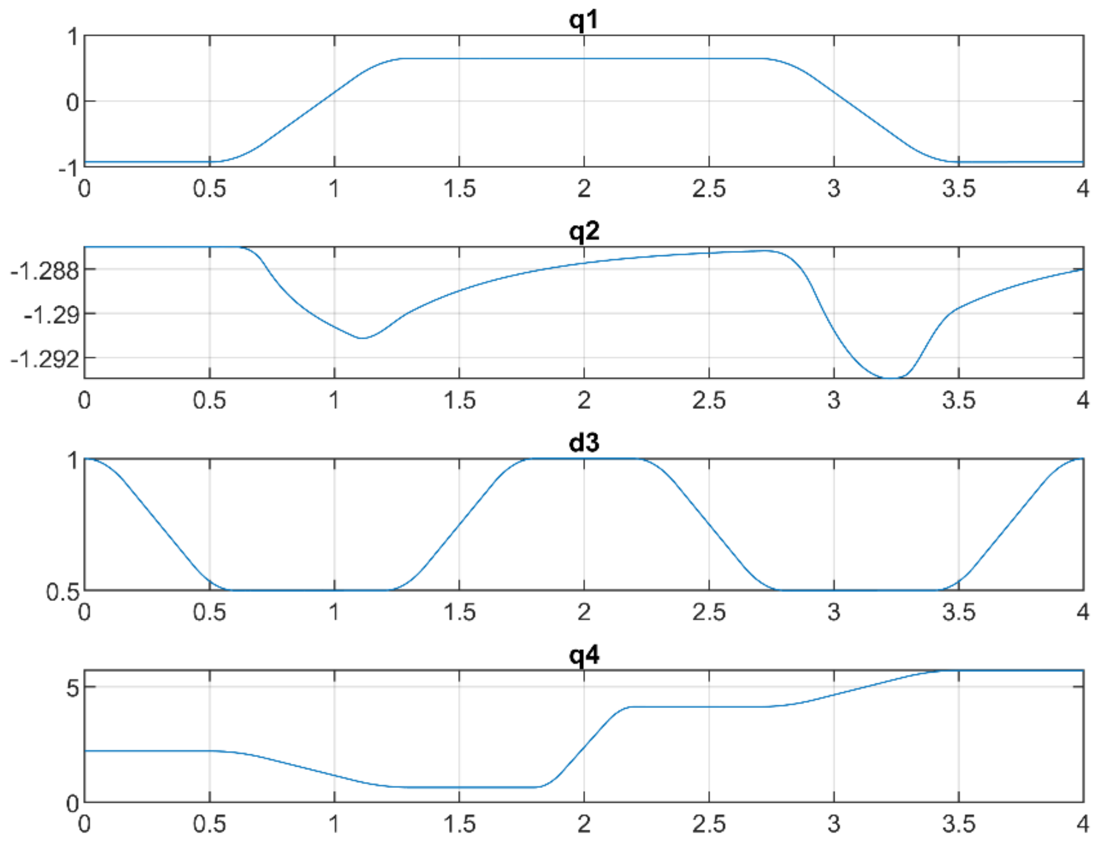

Figure 7: First order Joint variables

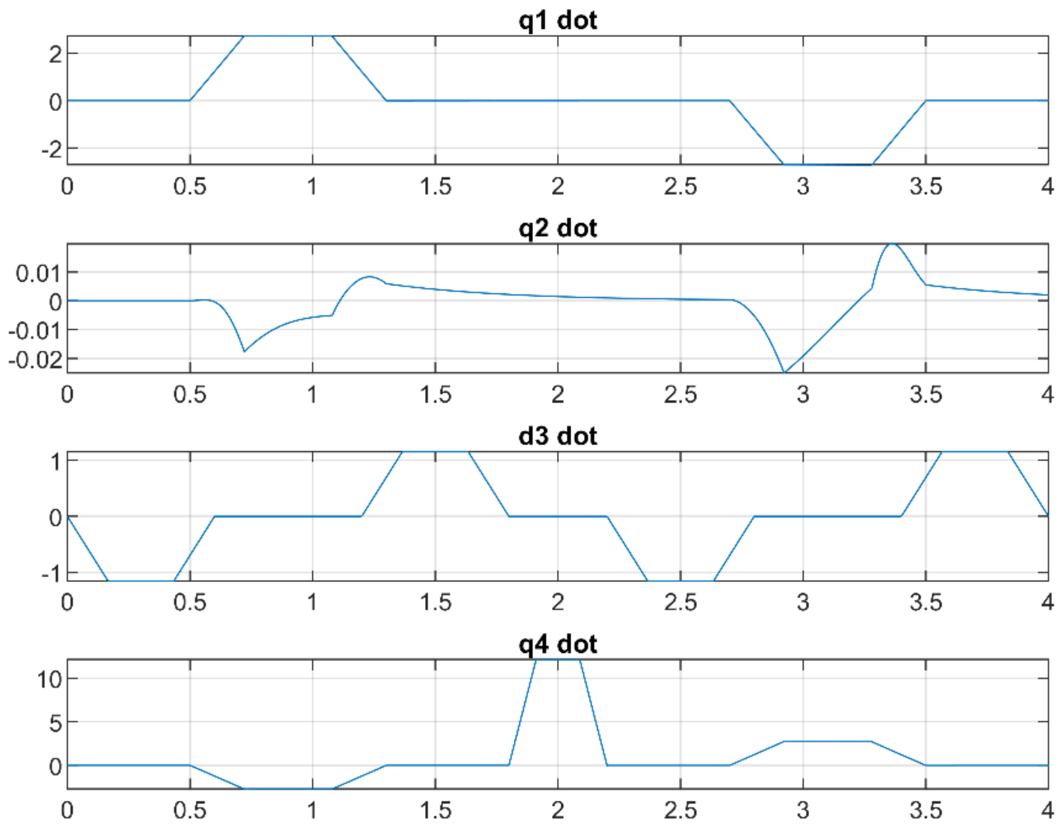

Figure 8: Second Order Joint Variables

Part 3:

The trajectory is defined using starting point, goal and a few via points. Trajectory between two consecutive points is considered as a separate segment and the total trajectory is generated by using trajectory of each segment.

Given N segments in a trajectory we know the path function P can be defined as,

$$\boxed{P_e = P_0 + \sum_{j=1}^{N} \frac{s_j}{\|(P_j - P_{j-1})\|} (P_j - P_{j-1})}$$

Here, p_0 is the initial position, P_e is the final position of the whole trajectory. P_j and P_{j-1} are final and initial positions of j^{th} segment. s_j is the length of trajectory of each segment and it can be defined as,

$$\boxed{S_j(t) = \begin{cases} 0 & 0 \le t \le t_{j-1} - \Delta t_j \\ S_j(t + \Delta t_j) & t_{j-1} - \Delta t_j < t < t_j - \Delta t_j \\ \|P_j - P_{j-1}\| & t_j - \Delta t_j \le t \le t_f - \Delta t_N \end{cases}}$$

t_j *and* t_{j-1} are the final and initial times of the j^{th} segment and Δt_j is the anticipation time. In our case we consider Δt_j to be 0.2s.

In this case we want each of our segment to propagate with a trapezoidal velocity profile. So the following constraints can be laid on $S_j(t)$,

$$\boxed{S_j(t) = \begin{cases} S_i + \frac{1}{2}\ddot{a}_c t^2 & t_i \le t_c \le t_f \\ \mathrm{S_i} + \ddot{\mathrm{a}}_c t_c \left(t - \frac{t_c}{2}\right) & t_c < t \le t_f - t_c \\ S_f - \frac{1}{2}\ddot{a}_c (t_f - t) & t_f - t_c < t \le t_f \end{cases}}$$

The segment lasts from t_i to t_f. S_i and S_f are initial and final lengths of the segment. The joint moves in a parabolic motion with constant acceleration $\ddot{a}_c$ between t_i *and* t_c and between $t_f - t_c$ *and* t_f. It moves with a constant velocity betweent_c *and* $t_f - t_c$.

For the solutions to be real, we consider

$$\ddot{a}_c = \frac{5\,|S_f - S_i|}{t_f^2}$$

From this we get t_c as,

$$t_c = \frac{t_f}{2} - \frac{1}{2}\sqrt{\frac{t_f^2 \ddot{a}_c - 4(S_f - S_i)}{\ddot{a}_c}}$$

The velocity and acceleration of the joint i.e., $\dot{P}_e\ and\ \ddot{P}_e$ can be calculated using the following equations,

$$\boxed{\dot{P}_e = \sum_{j=1}^{N} \frac{\dot{s}_J}{\|(P_j - P_{j-1})\|} (P_j - P_{j-1})}$$

Here $\dot{s}_J$ can be obtained as,

$$\boxed{\dot{S}_J(t) = \begin{cases} \ddot{a}_c * t & t_i \le t_c \le t_f \\ \ddot{a}_c * (t_c - t_i) & t_c < t \le t_f - t_c \\ \ddot{a}_c * (t_c - t_i) - \ddot{a}_c * (t - (t_f - t_c)) & t_f - t_c < t \le t_f \end{cases}}$$

$$\boxed{\ddot{P}_e = \sum_{j=1}^{N} \frac{\ddot{s}_J}{\|(P_j - P_{j-1})\|} (P_j - P_{j-1})}$$

Here $\ddot{s}_J$ can be obtained as,

$$\boxed{\dot{S}_J(t) = \begin{cases} \ddot{a}_c & t_i \le t_c \le t_f \\ 0 & t_c < t \le t_f - t_c \\ -\ddot{a}_c & t_f - t_c < t \le t_f \end{cases}}$$

The following results are obtained by using the above equations,

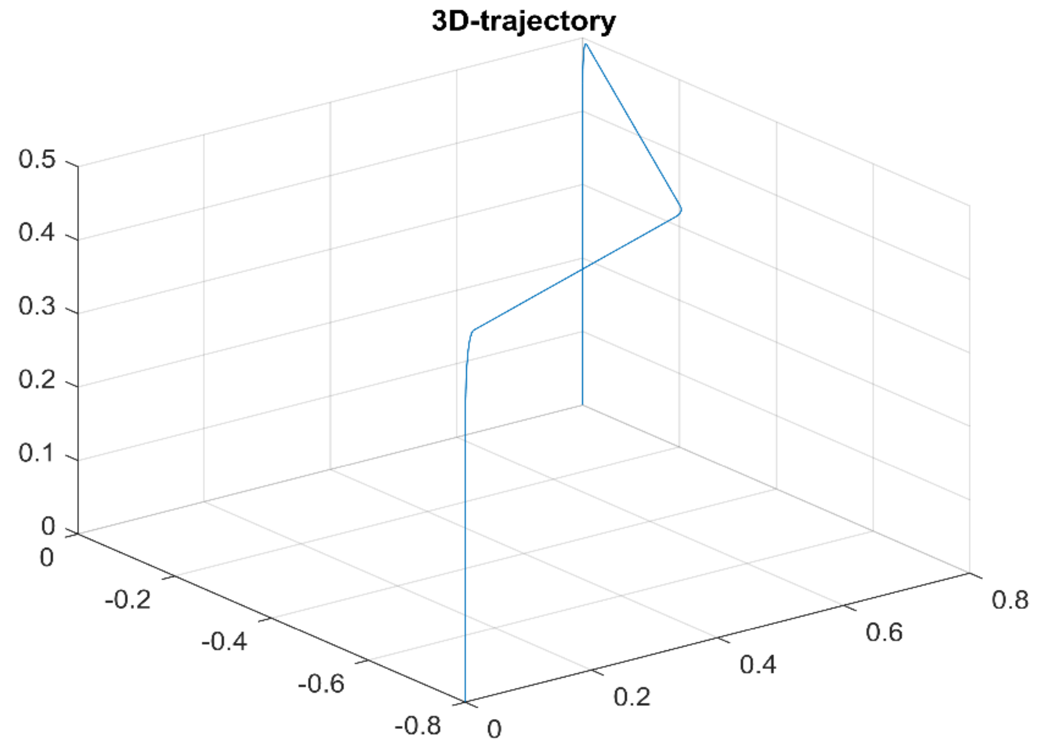

Figure 9: Generated Trajectory

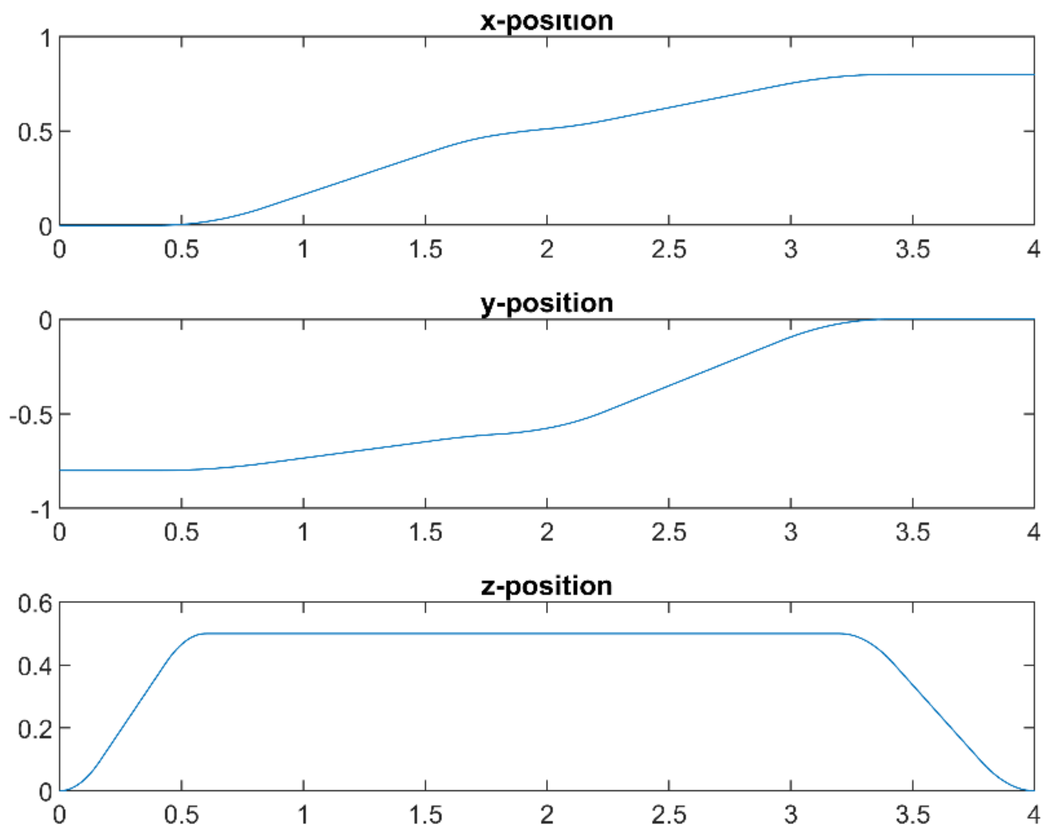

Figure 10: Joint Position with time

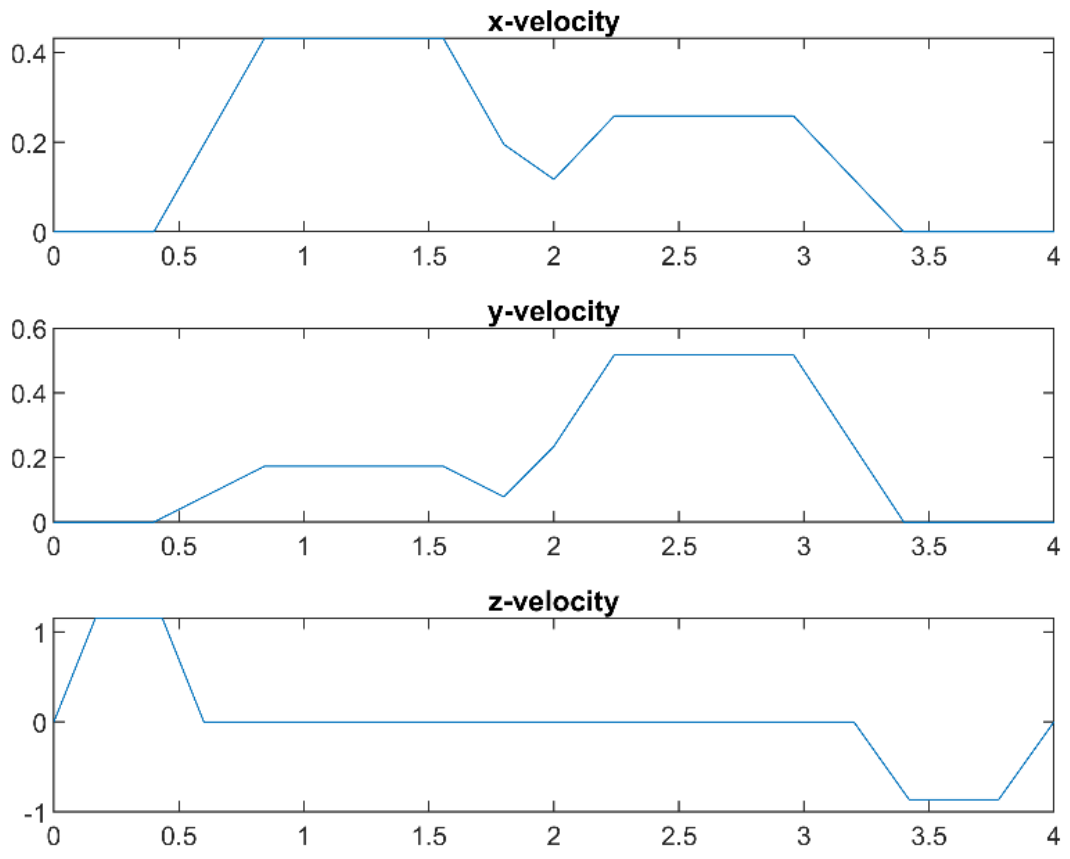

Figure 11: Joint Velocity profile with time

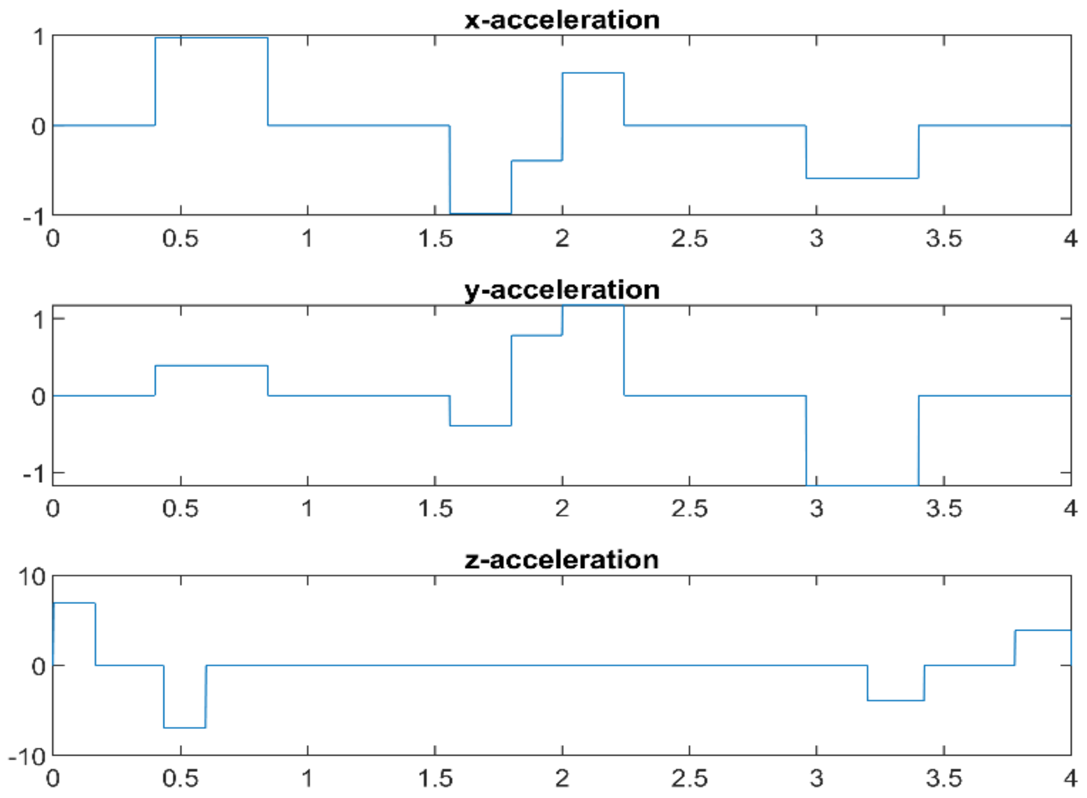

Figure 12: Joint Acceleration with time

Optional:

This is also an inverse dynamic control problem, except that we consider that compensation might not be perfect because of uncertainty of the model and payload or approximations due to real time constraints.

So we introduce compensation functions $\hat{B}$ and $\hat{n}$.

In our case we consider that compensation function is obtained by setting the end-effector mass i.e., m_{l4} to zero.

By substituting this to the obtained B and n functions in “part2” we get,

$$\hat{B} = \begin{bmatrix} 11.25 * C(\theta_2) + 25.3902 & 8.125 * C(\theta_2) + 12.5776 & -0.5 & 0.108 \\ 8.125 * C(\theta_2) + 12.5776 & 10.0776 & -0.5 & 0.108 \\ -0.5 & -0.5 & 35 & 0 \\ 1.1 & 1.1 & 0 & 3 \end{bmatrix}$$

$$\hat{C} = \begin{bmatrix} -5.625 * \dot{\theta}_2 * S(\theta_2) & -5.625 * S(\theta_2)(\dot{\theta}_1 + \dot{\theta}_2) & 0 & 0 \\ 5.625 * \dot{\theta}_2 * S(\theta_2) & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, g(q) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

To deal with this uncertainty we perform robust control. Here similar to “part2” a linear $\ddot{q}$ function is defined but an extra term “w” is added which is called robustness.

The inverse dynamics control can be written as,

$$\ddot{q} = B^{-1}(q)(\tau - \tau^1)$$

Here,

$$\tau = \hat{B}(q) * y + \hat{n}(q, \dot{q})$$

$$y = \ddot{q_d} + K_D * \dot{\tilde{q}} + K_P * \tilde{q} + w$$

$$\tau^1 = n(q, \dot{q})$$

Where,

$$w = \begin{cases} \frac{\rho}{\epsilon} z & per \ \|z\| < \epsilon \\ \frac{\rho}{\|z\|} z & per \ \|z\| \geq \epsilon \end{cases}$$

ρ is a scalar quantity defined based on the amount of uncertainty. ϵ is a small positive number.

$Z \ or \ zeta \ = D^T Q \xi$

D is $2n * n$ matrix $[0 \ I]^T$

$\xi = \begin{bmatrix} \tilde{q} \\ \dot{\tilde{q}} \end{bmatrix}$ and Q is a 2n*2n positive definite matrix.

We have considered Q as I_{2n*2n} and $\rho = 30$ and $\epsilon = 0.06$.

By plugging in all the mentioned values and using inverse dynamic control in the Simulink, we get the following results.

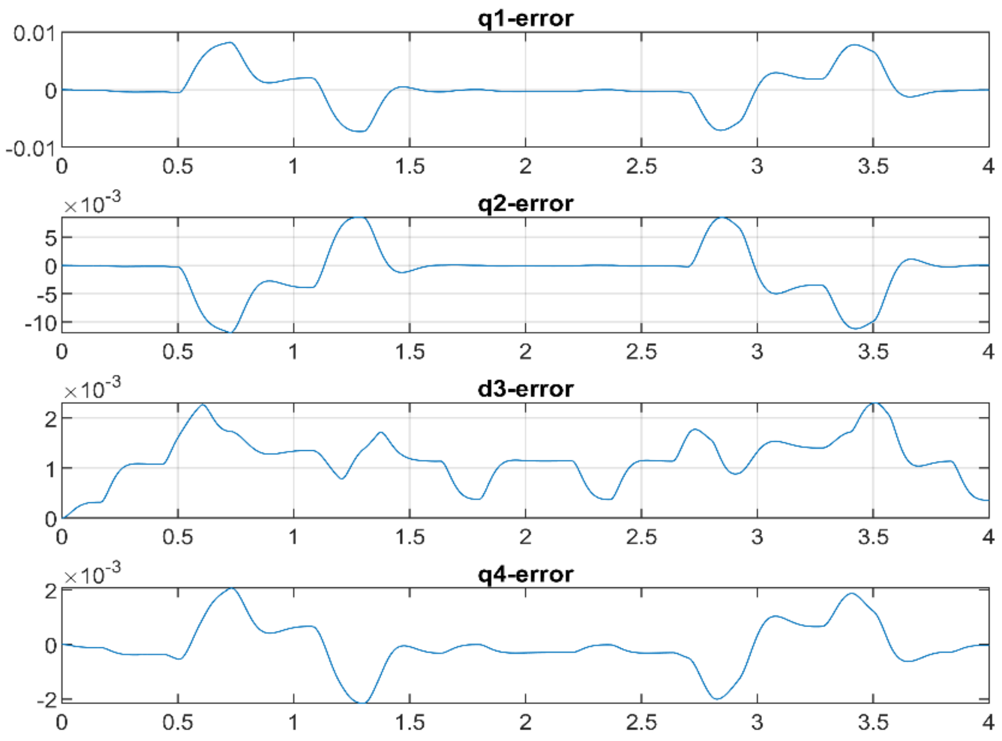

Figure 13: Error in first order joint variables

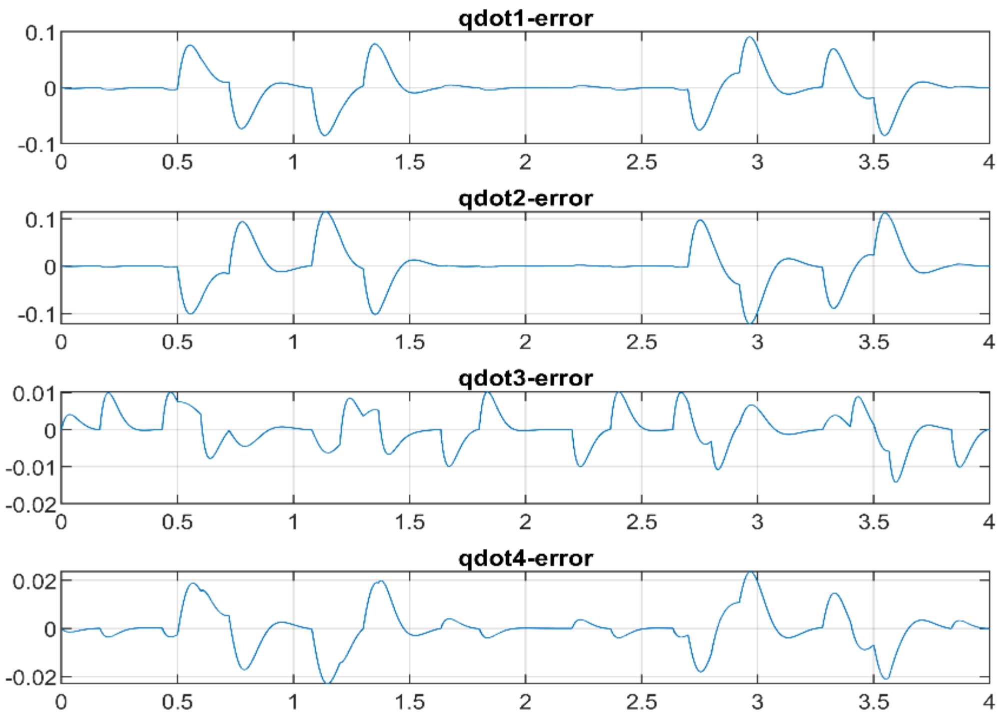

Figure 14: Error in second order joint variables

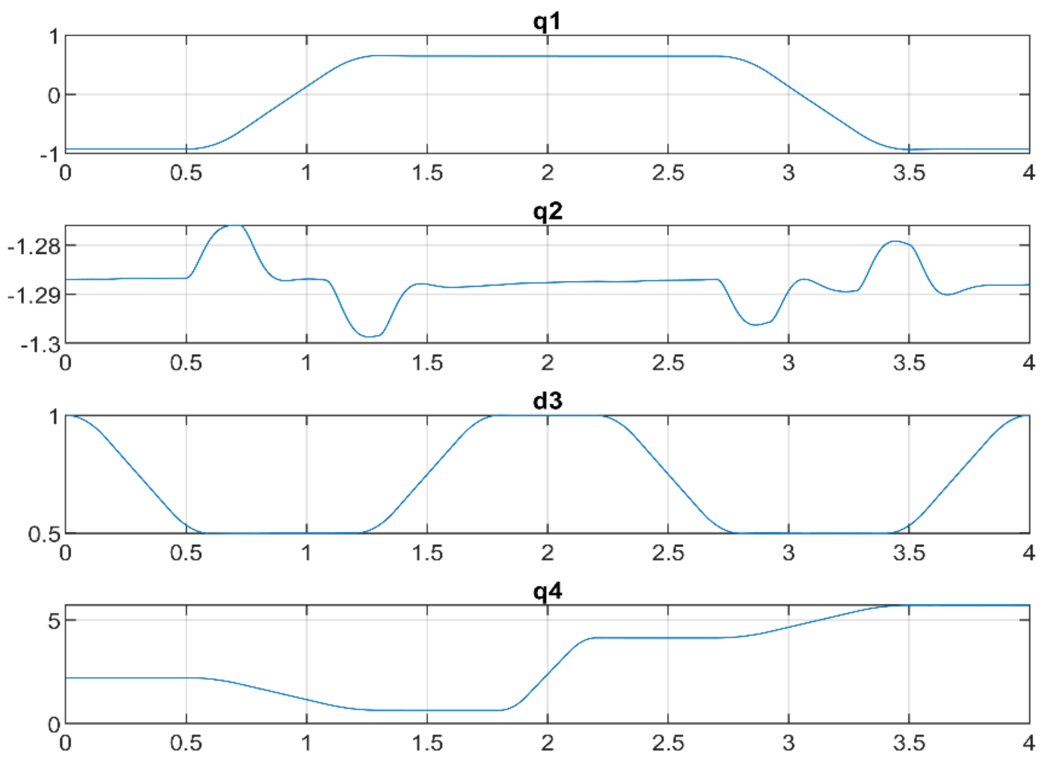

Figure 15: First order joint variables

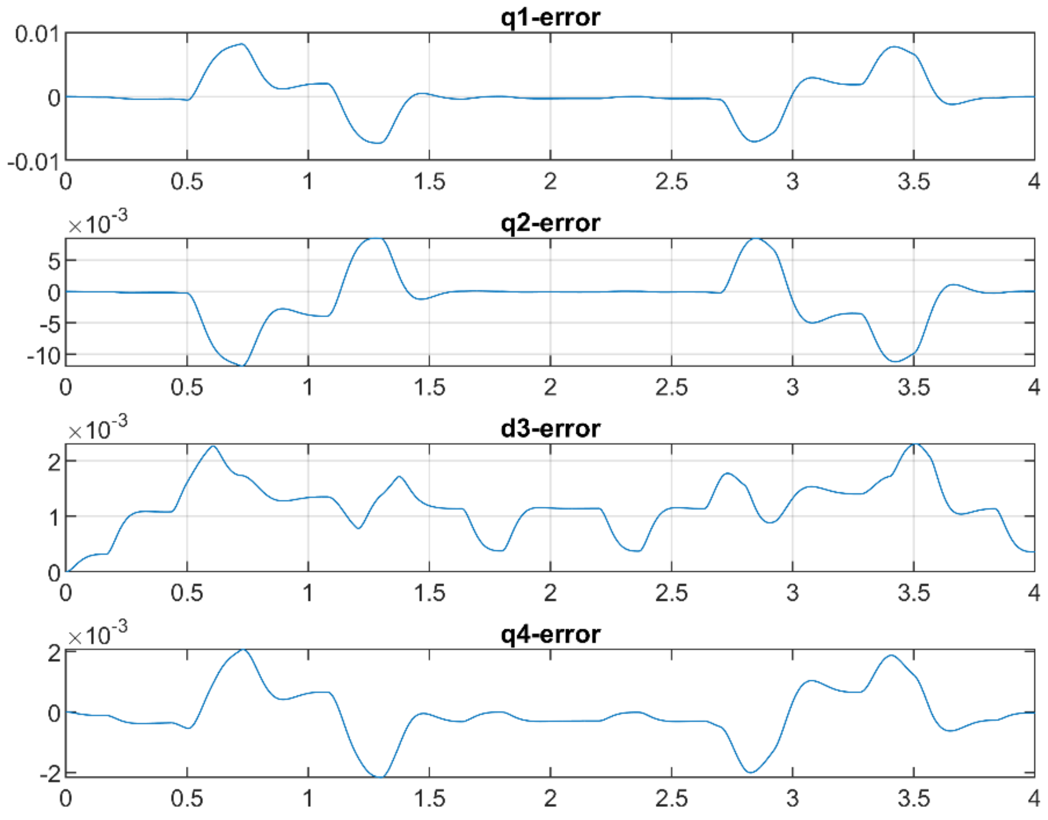

Figure 16: Second order joint variables

The implementation of inverse dynamics control scheme requires computation of the B(q) and n(q,$\dot{q}$). These terms must be computed on-line since control is now based on nonlinear feedback of the current system state, and thus it is not possible to precompute the terms off line as for the previous technique. This method is based on the assumption of perfect cancellation of dynamic terms but that is not the case in reality. Implementation of inverse dynamics control laws requires that parameters of the system dynamic model are accurately known and the complete equations of motion are computed in real time. These conditions are difficult to verify in practice.