SAPIENZA
UNIVERSITÀ DI ROMA

Mechanics of Robot Manipulators
Prof. Dionisio Del Vescovo

Assignment

Group Members

Akshay Shrikant Dhalpe - 1855686
Sandeep Kumar Omkarnath Gupta -1844011

Index

Part 1

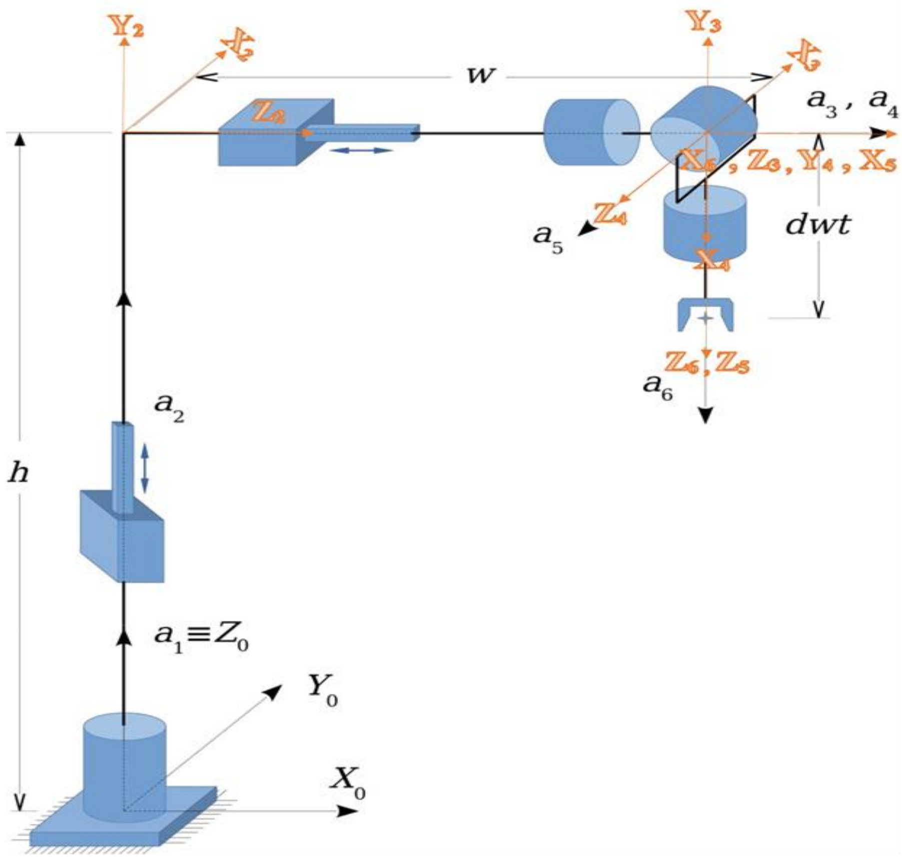
DENAVIT-HARTENBERG CONVENTION is used to locates the coordinate system on each link for robots with serial kinematic chain and screw joints.

Following are the rules used for setting frames from **(0)** to **(n-1)**

1. The Z_i axis must be coincident with the axis of joint $i+1$.
2. The X_i axis must be perpendicular to the axis Z_{i-1}.
3. The X_i axis must intersect the axis Z_{i-1}.

Based on these rules we can further establish **frame(i)** with the following algorithm

- Z_i axis coincident with the axis of joint $i+1$.
- X_i axis along the common normal between Z_{i-1} and Z_i axes and with the positive direction.



Assignment of the frame according to the DENAVIT-HARTENBERG CONVENTION gives a set of parameters referred as **D-H parameters** gives the geometry of the Robot.

- d_i, offset, distance from the origin O_{i-1} to X_i, measured on Z_{i-1}.
- $_i$, rotation, angle from X_{i-1} to X_i, measured about Z_{i-1}.
- a_i, length of link i, distance from Z_{i-1} to the origin O_i, measured on X_i.
- α_i, twist of link i, angle from Z_{i-1} to Z_i, measured about X_i.

i	Type	r_i	t_i	d_{0i}	ϑ_{0i}	a_{0i}	α_{0i}
1	Revolute	1	0	0	Q1	0	0
2	Prismatic	0	1	Q2+d2	90	0	90°
3	Prismatic	0	1	Q3+d3	0	0	0
4	Revolute	1	0	0	270°+Q4	0	90°
5	Revolute	1	0	0	90°+Q5	0	90°
6	Revolute	1	0	0	Q6	0	0

The parameters $\mathbf{d_i}$ and $\boldsymbol{\vartheta_i}$ describe the geometry of the joint i,
The parameters $\mathbf{a_i}$ and $\boldsymbol{\alpha_i}$ describe the geometry of the link i.

Homogeneous transformation between two adjacent coordinate frames is given by D-H matrix $\mathbf{M_{i-1,\,i}}$.

$\mathbf{M_{i-1,\,i}}$ depends on 4 parameters d_i, ϑ_i, a_i, α_i while the relative position of 2 rigid bodies in 3-space requires 6 DoF → this reduction is due to the D-H convention which constrains the choice of frame **(i)** w.r.t. frame **(i-1)**

$$\mathbf{M_{i-1,i}}=\begin{bmatrix} C\vartheta i & -S\vartheta iC\alpha i & S\vartheta iS\alpha i & aiC\vartheta i \\ S\vartheta i & C\vartheta iC\alpha i & -C\vartheta iS\alpha i & aiS\vartheta i \\ 0 & S\alpha i & C\alpha i & di \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$M_{01} = \begin{bmatrix} C1 & -S1 & 0 & 0 \\ S1 & C1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$M_{12} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & w \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$M_{23} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & w \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$M_{34} = \begin{bmatrix} C4 & 0 & S4 & 0 \\ S4 & 0 & -C4 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$M_{45} = \begin{bmatrix} C5 & 0 & S5 & 0 \\ S5 & 0 & -C5 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$M_{56} = \begin{bmatrix} C6 & -S6 & 0 & 0 \\ S6 & C6 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$M_{02} = \begin{bmatrix} S1 & 0 & C1 & 0 \\ C1 & 0 & S1 & 0 \\ 0 & 1 & 0 & h \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$M_{03} = \begin{bmatrix} -S1 & 0 & C1 & C1w \\ C\theta & 0 & S1 & S1w \\ 0 & 1 & 0 & h \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$M_{04} = \begin{bmatrix} -S1C4 & C1 & -S1S4 & C1w \\ C1C4 & S1 & C1S4 & S1w \\ S4 & 0 & -C4 & h \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$M_{05} = \begin{bmatrix} -S1C4C5 + C1S5 & -S1S4 & -S1C4S5 - C1C5 & C1w \\ C1C4C5 + S1S5 & C1S4 & C1C4S5 - S1C5 & S1w \\ S4C5 & -C4 & S4S5 & h \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Part 2

Position Forward kinematics is calculated after determining the homogeneous transformation between adjacent matrix.

$$\mathbf{M_{0n}=M_{01}(q_1) \;....\; M_{i-1,\,i}(q_i)....\; M_{n-1,\,n}(q_n)}$$

$M_{06}=M_{01}M_{12}M_{23}M_{34}M_{45}M_{56}$

$$M_{06}=\begin{bmatrix} (-S1C4C5+C1S5)C6-(S1S4S6) & (-S1C4C5+C1S5)(-S6)-S1S4C6 & -S1C4S5-C1C5 & C1w \\ (C1C4C5+S1S5)C6+C1S4S6 & (C1C4C5+S1S5)(-S6)+C1S4C6 & C1C4S5-S1C5 & S1w \\ S4C5C6-C4S6 & -S4C5S6-C4C6 & S4S5 & h \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$M_{06} = \begin{bmatrix} \cdots & \cdots & \vdots & xp \\ \vdots & R06 & \vdots & yp \\ \vdots & \cdots & \cdots & zp \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Solving the above equations, we get the following values

$$\mathbf{M_{06}(q)=M_{06}(s)}$$

World Coordinates: $\{\, xp \;\; yp \;\; zp \;\; \vartheta 1 \;\; \vartheta 2 \;\; \vartheta 3 \,\}$

Known	Unknown
Q_j	M_{06}

$x_p = C1(Q2+w)$

$y_p = S1(Q2+w)$

$z_p = h$

$\vartheta_1 = Tan^{-1}\left(\frac{x_p}{y_p}\right)$

$$R_e = \begin{bmatrix} C2eC3e & -C2eS3e & S2e \\ C1eS3e + C3eS1eS2e & C1eC3e + S1eS2eS3e & -C2eS1e \\ S1eS3e - C1eC3eS2e & S2eC3e & C2eC1e \end{bmatrix}$$

ϑ_{1e} = arctan (-R_e (2,3), Re (3,3))

ϑ_{3e}= arctan (-R_e (1,2), Re (1,1))

ϑ_{2e}= arctan (R_e (1,3), $\pm\sqrt{1-(R_e(1,3))^2}$)

Position Inverse Kinematics is calculated by using Paul Method.
We have cut the graph from link 3rd so the matrix is given below,

	Known	Unknown
$\mathbf{M_{34}M_{45}M_{56}=M_{01}^{-1}M_{12}^{-1}M_{23}^{-1}M_{06}}$	M_{06}	Q_j

$M_{36}=M_{01}^{-1}M_{12}^{-1}M_{23}^{-1}M_{06}$

$$M_{36}=\begin{bmatrix} C4C5C6+S4S6 & -C4C5S6+C4C6 & C4S5 & 0 \\ C6S4C5-C4S6 & -S4S6C5-C4S6 & -S4S5 & 0 \\ S5C6 & -S5S6 & -C5 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}=\begin{bmatrix} \dots & \dots & \vdots & 0 \\ \vdots & R_{36} & \vdots & 0 \\ \vdots & \dots & \dots & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_{03}^{-1}=\begin{bmatrix} -S1 & C1 & 0 \\ 0 & 0 & 1 \\ C1 & S1 & 0 \end{bmatrix}$$

$R_{36}= R_{03}^{-1}R_e$

We have used the forward and inverse Kinematics to find the below solution:
As,

theta1 = atan2(S1(Q2+w), S1(Q2+w))

Theta1 depends on Q2, as the Q2 changes has two solution w.r.t Q2 theta1 will change.

From the Forward Kinematics we found the Q1(theta1), h and w

From Inverse Kinematics we find the Q4 (theta4), Q5(theta5), Q6(theta6).

We are making matrix of 6 row and 4 Column because we are getting 4 Solution for the given robot and as per the variation of the joint parameters.

	NS1	NS2	NS3	NS4
q1	arctan(R06(2,4), R06(1,4))	arctan(R06(2,4), R06(1,4))	arctan(R06(2,4), R06(1,4))	arctan(R06(2,4), R06(1,4))
q2	R06(3,4)-h	R06(3,4)-h	R06(3,4)-h	R06(3,4)-h
q3	Sqrt((R06(1,4))^2+(R06(2,4)^2)-w	-Sqrt((-R06(1,4))^2+(R06(2,4)^2)-w	Sqrt((R06(1,4))^2+(R06(2,4)^2)-w	-Sqrt((-R06(1,4))^2+(R06(2,4)^2)-w
q4	arctan(R36(1,3), -R36(2,3))	arctan(R36(1,3), -R36(2,3))	arctan(R36(1,3), -R36(2,3))	arctan(R36(1,3), -R36(2,3))
q5	arctan (R (3,3), -sqrt(1-R36(3,3))	arctan (R (3,3), sqrt(1-R36(3,3))	arctan (R (3,3), -sqrt(1-R36(3,3))	arctan (R (3,3), sqrt(1-R36(3,3))
q6	arctan(-R36(3,2), R36(3,1))	arctan(-R36(3,2), -R36(3,1))	arctan(-R36(3,2), R36(3,1))	arctan(-R36(3,2), -R36(3,1))

Now proceeding with the Forward and Inverse Velocity and Acceleration of the Robot we must find out the Screw Matrix (L), Velocity Matrix (W), Acceleration Matrix (H) for each joint with respect to Base.

Define Frame

- The Zi axis must be coincident with the axis of joint i+1.
- The Xi axis must be perpendicular to the axis Zi-1.
- The Xi axis must intersect the axis Zi-1.

Find DH parameters

- di, offset, distance from the origin Oi-1 to Xi, measured on Zi-1,
- θi, rotation, angle from Xi-1 to Xi, measured about Zi-1,
- ai, length of linki, distance from Zi-1 to the origin Oi, measured on Xi,
- αi, twist of linki, angle from Zi-1 to Zi, measured about Xi.

Transformation matrix

- From DH matrix calculate Transformation matrix (Mj-1, i) for link(i=1,2,3,4,5,6) and gripper(t) with respective to previous link and base .
- From M06 matrix and Gripper parameters. Calculate M0t matrix.
- From M0t matrix calculate Rotational matrix R0t and Tralational Matrix (T0t)

Forward Kinematics

- **Input:** We have joint frame of the robot
- **Output**: We want to define the Tool center point of the robot .

Inverse Kinematics

- **Input:** We have joint frame of the robot
- **Output**: We want to define the Tool center point of the robot

Solution

- Using forward and Inverse Kinematics we have 4 solution which will drive out robot.

Part 3

First Let start with Screw Joint
We have two Prismatic and 4 revolute Joint

Screw Matrix: We have considered the Screw axis coincide with the Axis of Rotation so Screw Matrix will be as follow:

L_i (Prismatic Joint) = $\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$

L_i (Revolute Joint) = $\begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$

To find the **Screw Matrix** with respect to base we have used the following method

$L_{0i} = M_{0,\,i-1} * L_i * M_{i-1,0}$ (inverse of Homogenous matrix can be found out with the help of M_rec syntax defines in the global function).

With the help of Above equation, we have found out the $L_{0,1}$; $L_{0,2}$; $L_{0,3}$; $L_{0,4}$; $L_{0,5}$; $L_{0,6}$.

Computing $\mathbf{M}_{21}$ and $\mathbf{M}_{12} = \mathbf{M}_{21}^{-1}$ we can verify $\longrightarrow$ $\mathbf{L}^{(2)} = \mathbf{M}_{21}\mathbf{L}^{(1)}\mathbf{M}_{12}$

For proceed with the **Velocity Matrix**

We must use the following method provide in the Assignment.

Revolute → $\mathbf{L}^{(k)} = \left[\begin{array}{ccc|c} 0 & -z_{za}^{(k)} & z_{ya}^{(k)} & \\ z_{za}^{(k)} & 0 & -z_{xa}^{(k)} & -\underline{\mathbf{z}}_a^{(k)}\mathbf{T}_a^{(k)} \\ -z_{ya}^{(k)} & z_{xa}^{(k)} & 0 & \\ \hline 0 & 0 & 0 & 0 \end{array}\right]$

Prismatic → $\mathbf{L}^{(k)} = \left[\begin{array}{ccc|c} 0 & 0 & 0 & z_{xa}^{(k)} \\ 0 & 0 & 0 & z_{ya}^{(k)} \\ 0 & 0 & 0 & z_{za}^{(k)} \\ \hline 0 & 0 & 0 & 0 \end{array}\right]$

In our case we must define the for revolute zza = 1 and zya=0, zxa=0 in prismatic we have to define zxa=0, zya=0 and zza=1.

Now let's define **Velocity matrix** for the joint and velocity matrix with respect to base. To carry out this process we must define the pattern of the velocity matrix which is in the below form:

$$\left[\begin{array}{ccc|c} 0 & -\omega_z & \omega_y & V_{Ox}^{(i)} \\ \omega_z & 0 & -\omega_x & V_{Oy}^{(i)} \\ -\omega_y & \omega_x & 0 & V_{Oz}^{(i)} \\ \hline 0 & 0 & 0 & 0 \end{array}\right]$$

This matrix consists of angular velocity and linear velocity of that joint W_i. We are required Velocity matrix w.r.t base so we have to find out W_{01}, W_{02}, W_{03},W_{04},W_{05},W_{06}.
To find the respective velocity matrix we have used the assigned formula

$$W_{0i} = W_{0,i-1} + W_{i-1,i}$$

So, using this formula we can proceed and considering W_{01}=L_{01}*velocity of joint(Q(1).) .
W_{12}= L_{12}*velocity of joint(Q(2).) in the same way we can find the velocity of the joint .

To find the velocity of the joint wrt to base we have assign formula so my velocity matrix wrt to base become W_{02}=W_{01}+W_{12} (to define W12 we have used W_{12}=L_{12}(Screw matrix of that joint) *

Velocity of joint(Q(i).)) in the similar way using the for loop we can find W_{03},W_{04},W_{05},W_{06}.

To define **Acceleration Matrix** of the joint and acceleration with respect to base we have used the assign formula.
H_{0i}= $H_{0,i-1}$+$M_{0,i-1}$*($d^2M_{i-1,i}/dt^2$)*$M_{i,0}$+2$W_{0,i-1}$*$W_{i-1,i}$ using this formula we can calculate the following H for the respective frame we have seperatelly calculated the single and Double derivative of the Homogeous matrix with respect to time .
DM01,DM02,DM03,DM04,DM05,DM06 and double derivative .
($dM_{i-1,i}$*Q(1)/dt) = ($dM_{i-1,i}$/dt)*(dQ(1)) ....(First Derivative)

d(($dM_{i-1,i}$/dt)*(dQ(1)))/dt =(d^2($dM_{i-1,i}$)/d^2t)*$(dQ(1)/dt)^2$ +($dM_{i-1,i}$/dt) *(d^2(dQ(1))/d^2t)

.... (Second Derivative) .

For First Joint i=1
H01= ($d^2M_{i-1,i}/dt^2$) = (d^2($dM_{i-1,i}$)/d^2t)*$(dQ(1)/dt)^2$ +($dM_{i-1,i}$/dt) *(d^2(dQ(1))/d^2t)
H02= $H_{0,i-1}$+$M_{0,i-1}$*($d^2M_{i-1,i}/dt^2$)*$M_{i,0}$+2$W_{0,i-1}$*$W_{i-1,i}$ .... (i=2)
Similarly for the rest of the joints with respect to base frame (i=3,4,5,6).

After Defining the Screw Matrix , Velocity Matrix and Acceleration Matrix we will proceed with the **Forward Kinematics** .

In this we have to find the find the velocity of the Gripper with the help of the joint velocity .
To find the position of the joint wrt base frame we have the following equation as $P^{(0)}=W_{0i}*P^i$

Linear Velocity of the TCP .

We have found the Position of the End effector by using the above equation as Position of End effector wrt to base equal to Homogenous matrix last joint wrt to base mulity position of the end effector. Once we get the position of the end effector wrt to base we can compute the velocity of the end effector as velocity of the end effector equal to velocity matrix of last joint wrt to base mulity by position of the end effector.

Angular Velocity of the TCP.
As have define skew symmetry matrix of velocity as follow

$$\begin{bmatrix} 0 & -w_z & w_y & V_x \\ w_z & 0 & -w_x & V_y \\ -w_y & w_x & 0 & V_z \end{bmatrix}$$

We can have angular velocity of joint and Angular velocity of the joint is been calculated with the help of the Velocity matrix of the joint wrt to the base frame.

Similarly we can find the Linear and Angular Accerleration of the joint and wrt to the frame.
Matrix is as below

$$\begin{bmatrix} \dot{w}^{(i)} + {w^{(i)}}^2 & A^{(i)} \\ 0 & 0 \end{bmatrix}$$

We can define the acceleration of the TCP with the help of following equation as
$Sdd=H^i*P^i$
Angular Acceleration can be find using $H_{0i}-W_{0i}^2$ (used in Matlab)

Inverse Kinematics we have end effector Velocity and Acceleation we will find the joint velocity and acceleration of the joint using Jacobian Matrix.

To calculate the **Jcobian Matrix**

From screw matrix we can derive the following Jacobian Matrix as Jaacobian matrix is number of joint x number of joint . In assign robot we have 6 joint so Jacobian Matrix become 6 BY 6 matrix.

If determinate of Jacobian Matrix is less than tiny, in this case to calculate the acceleration and velocity of the joint using inverse Jacobian Matrix and if Jacobian Determinate is greater than tiny we can use $\dot{Q}(Velocity\ of\ Joint)$=Velocity of TCP divided by Jacobian Matrix .

Using Jacobian we can calculate the joint velocity and acceleration.

Screw Matrix

- Roto translation for finite displacement we find the screw matrix
- We have to find the screw axis of each joint and wrt to base.
- L is used in later part to define the velocity .
- We have define the L for prismatic and Revolute joint. We have define the L screw axis conside with the axis of the joint.

Velocity Matrix

- We have assign the formula to calculate the velocity matrix . We have to calculate the velocity of the joint and velocity of the joint wrt to base
- W(Velocity) of joint is calculated with the help of formula $W_{12} = L_{12}$ * velocity of that joint.

Accelration Matrix

- We have assign the formula to calculate the acceleration matrix . We have to calculate the acceleration of the joint and accleration of the joint wrt to base .
- To derive the acceleration of the joint we have to calculate single and Double derivative of the Homogenous matrix.

Velocity Forward Kinematics

- In velocity forward Kinematics we have joint velocity and we have to calculate the TCP velocity.

Accelration Forward Kinematics

- In acceleration forward Kinematics we have joint accelration and we have to calculate the TCP acceleration.

Jacobian Matrix

- Jacobian is used in the inverse Kinematics which is a matrix of size "no of joint X no of joint"

Part 4

The execution of a task by a robot involves suitable movements: the handling of this problem is denoted with trajectory generation or planning.
For computation of **executive time** we analyze two cases

1. **1st case** Maximum velocity is reached as displacement is large enough.
2. **2nd case** Maximum velocity is not yet been reached in this situation only the acceleration and deceleration limit the trajectory.
3. **3rd case** is the borderline case.

$$S = \frac{1}{2}at^2$$
$$v = at$$
$$from\ the\ two\ equations\ above\ , S = \frac{(v)^2}{a}$$
$$Limiting\ displacement = \frac{(velocitymax)^2}{accelerationmax}$$

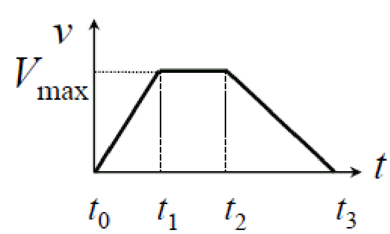
If $displacment \geq Limiting\ displacement$ then it belongs to **first case**.

$$T = \frac{S}{vmax} + \frac{1}{2}vmax(\frac{1}{a} + \frac{1}{d})$$

$$Here\ a = acceleration\ , d = deceleration$$

Assuming constant acceleration and deceleration

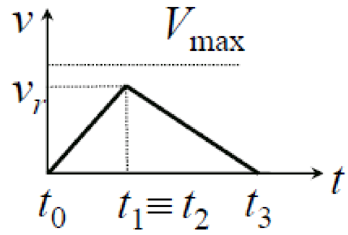
$$T = \frac{S}{vmax} + \frac{vmax}{amax}$$



If $displacment < Limiting\ displacement$ then it belongs to **second case**.

$$T = \sqrt{\frac{a}{d} + \frac{d}{a}} * \sqrt{\frac{2S}{a + d}}$$

$$Here\ a = acceleration\ , d = deceleration$$

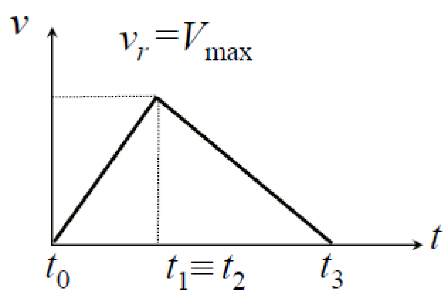


Assuming constant acceleration and deceleration

$$T = 2\sqrt{\frac{S}{amax}}$$

Compute the value of time for each joint.
Divide the time taken by the joints into equal intervals.
Compute the value of $displacement$, $velocity\ and\ acceleration$ for each time interval of all the joints.

For **Borderline (Third Case)** we have the formula for min execution time as follows



$$T = V_{max}\left(\frac{1}{A_a} + \frac{1}{A_d}\right)$$

Again, $A_a = A_d = A_{max}$
So, we have the modified equation as

$$T = V_{max}\left(\frac{2}{A_{max}}\right)$$

So, we will have all the min execution time for each joint I as T_i

That gives us final execution time → $Tmax = max_{i=1}^{i=6}\{T_i\}$.

Now, the actual velocity and acceleration reached by the joints for their respective execution times T_i is V_r and A_r.

Therefore, it is clear for case 1 and borderline $V_r = V_{max}$ and $A_r = A_{max}$.

And for the remaining case 2 we can calculate with general equations of motion

$$A_r = (A_{max}) = \left(\frac{2DQ}{T_i^{\,2}}\right)$$

and

$$V_r = \left(\frac{A_r * T_i}{2}\right)$$

Now, it is necessary for the joints to start and stop simultaneously to avoid motor overload i.e. adopt the same execution time for each joint.

So, we optimize the trajectory for minimization of product between acceleration and velocity, and then scale w.r.t time.

- **Input**
  - Inputs
  - $DQ, V_{max}, A_{max}, Q_{in}$
- **Calculate DQlim**
  - Calculate DQ lim to determine the in which case the examples falls in.
- **Determine Tmax**
  - Determine the value for T_i for the respective case.
  - Calculate Executive Time(T_{max}) = max(T_i)
- **Calculate Vel and Acc**
  - Compute the Velocity and Acceleration of the the joint for the repective time intervals.
- **Break and Coeff**
  - Select the number of breaks and the coefficient of the equation according to the case.
- **Output**
  - Using the file Cylindrical_Robot_Trajectory('Polynomials')
  - Output V, A, T_a Q, Qd Qdd

• **5th order Polynomial** for reducing the vibrations**:**

We have for this part the following equation of motion:

$$q(t) = Qin + \frac{DQ}{T^3} * \left(10t^3 - \frac{15}{T}t^4 + \frac{6}{T^2}t^5\right) \text{ (0)}$$

First of all we must find the Global Execution Time T_{max} related to this equation.
Then we need to determine $q(\dot{t})$, $q(\ddot{t})$, $q(\dddot{t})$.

$$q(\dot{t}) = \frac{DQ}{T^3} * \left(30t^2 - \frac{60}{T}t^3 + \frac{30}{T^2}t^4\right) \quad \text{(1)}$$

$$q(\ddot{t}) = \frac{DQ}{T^3} * \left(60t - \frac{180}{T}t^2 + \frac{120}{T^2}t^3\right) \text{ (2)}$$

$$\dddot{q}(t) = \frac{DQ}{T^3} * (60 - \frac{360}{T} t + \frac{360}{T^2} t^2) \quad \textit{(3)}$$

We will do the **same procedure** for every joint, the goal is to find T_i the minimum execution time for the joint i. We have A_{max} and V_{max} as input values for each joint, so let's use them.

a) $\dddot{q}(t) = 0$

This equation gives the time solutions t_a as $\ddot{q}(t_a) = Amax$. Then we can get an equation for T as follows:

T_{ia} = $\pm \sqrt{\frac{DQ}{Amax} * (60W - 180W^2 + 120W^3)}$ with W = $\frac{360 \pm \sqrt{43200}}{760}$

Here we keep only the positive solution -> time.
We should also not that the value of W varies between

$$0 \leq W \leq \frac{1}{2}$$

b) $\ddot{q}(t) = 0$

This equation gives the time solutions t_v as $\dot{q}(t_v) \ = Vmax$. Then we can get an equation for T as follows:

T_{iv} = $\frac{30 * DQ}{16 * Vmax}$

c) Finally, we can determine for the robot the global execution time as follows:

T_a = max (T_i) with T_i = max (T_{ia},T_{iv})

Then, to obtain Q, Qd, Qdd we can use the function "Polynomials" which will build the trajectories.
We are working with only one interval because we have just one kind of polynomial between 0 and T_a.
The coefficients are given by the equation *(0)*.

• <u>Assymetric Shp-5</u> <u>for reducing the vibrations</u>**:**

We have for this part the following equation of motion:

$$q(t) = Qin + \frac{DQ}{T^3} * (28t^3 - \frac{48}{\sqrt{T}} t^{7/2} + \frac{21}{T} t^4) \quad \textit{(0)}$$

First of all, we must find the Global Execution Time T_a is related to this equation.
Then we need to determine $\dot{q(t)}$, $\ddot{q(t)}$, $\dddot{q(t)}$.

$$\dot{q(t)} = \frac{DQ}{T^3} * (84t^2 - \frac{168}{\sqrt{T}} t^{5/2} + \frac{84}{T} t^3) \quad (1)$$

$$\ddot{q(t)} = \frac{DQ}{T^3} * (168t - \frac{420}{\sqrt{T}} t^{3/2} + \frac{252}{T} t^2) \quad (2)$$

$$\dddot{q(t)} = \frac{DQ}{T^3} * (168 - \frac{630}{\sqrt{T}} t^{1/2} + \frac{504}{T} t) \quad (3)$$

As previously we will do the **same procedure** for every joint, the goal is to find T_i the minimum execution time for the joint i. We have A_{max} and V_{max} as input values for each joint, so let's use them.

a) $\dddot{q(t)} = 0$

This equation gives the time solutions t_a as $\ddot{q(t_a)} = Amax$. Then we can get an equation for T as follows:

$T_{ia} = \sqrt{\frac{DQ}{Amax} * (168W_a^2 - 420W_a^3 + 252W_a^4)}$ with $W_a = \frac{630 \pm \sqrt{58212}}{1008}$

We should also not that the value of W varies between

$$0 \leq W \leq \frac{1}{2}$$

b) $\ddot{q(t)} = 0$

This equation gives the time solutions t_v as $\dot{q(t_v)} = Vmax$. Then we can get an equation for T as follows:

$T_{iv} = \frac{DQ}{Vmax} * (84W_i^4 - 168W_i^5 + 84W_i^6)$

c) Finally, we can determine for the robot the global execution time as follows:

$T_a = \max(T_i)$ with $T_i = \max(T_{ia}, T_{iv})$

For this case, to obtain Q, Qd, Qdd we can’t use the function “Polynomials” which will build the trajectories because we have the order 7/2 which is not a whole number.
So, we are just going to use the equation written analytically: (0), (1) and (2).

IMPORTANT NOTE WHILE CODING FOR MATLAB (for all trajectories)-

While calculating execution time for each joint for all trajectories it is essential to take the absolute values of the give DQ as failing to do so will result in -ve time roots and complex conjugate solutions.

But while calculating V and A it is important to consider the sign of DQ!

Part 5

• **Inertial matrix of a Rigid Body**

The Inertial Matrix $I_k^{(k)}$ is given by

$$I_k^{(k)} = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} & m_{xG} \\ I_{yx} & I_{yy} & I_{yz} & m_{yG} \\ I_{zx} & I_{zy} & I_{zz} & m_{zG} \\ m_{xG} & m_{yG} & m_{zG} & m \end{bmatrix}$$

Where $[m_{xG} \quad m_{yG} \quad m_{zG}]$ are the centroids
And m is the mass of the link

And we have the relationship with the Inertial tensor as

$$I_{xx} = \frac{-J_{xx} + J_{yy} + J_{zz}}{2} \qquad I_{yy} = \frac{J_{xx} - J_{yy} + J_{zz}}{2} \qquad I_{xx} = \frac{-J_{xx+}J_{yy} + J_{zz}}{2}$$

$$I_{xy} = I_{yx} = -J_{xy} = -J_{yx} \quad , \qquad I_{xz} = I_{zx} = -J_{xz} = -J_{zx} \quad , \qquad I_{yz} = I_{zy} = -J_{yz} = -J_{zy}$$

• <u>Other related important matrix operations</u>

1. Skew

$$skew\{A\} = A - A^T$$

2. Trace

$$trace\{A\} = \sum_{i=1}^{n} a_{ii}$$

3. Pseudo dot product

$$A \bullet B = a_{14}b_{14} + a_{24}b_{24} + a_{34}b_{34} + a_{32}b_{32} + a_{13}b_{13} + a_{21}b_{21}$$

• **Forward Dynamics**

To find the forward dynamics, we use the set equation in matrix form

$$M = \ddot{Q}\, F$$

Where

$$M = \left[m_{ij}\right] = trace\,\{L_i \hat{I}_h L_j^T\} \qquad \text{where } h = \max(i, j)$$

And, The elements of the column vector of the forces F are all the terms in the equations that do not multiply the joint accelerations $\ddot{q}$:

$$F = [f_i] = \varphi_i + \left\{ \left(\sum_{j=i}^{n} \widetilde{\Phi}_j \right) \bullet L_i \right\} + trace\{L_i \hat{I}_i H_g^T\} + \sum_{j=i}^{n} (trace\,\{L_i I_j H_j^{*T}\})$$

Now to solve the above equations we find the required terms as follows.

$$\hat{I}_i = \sum_{j=i}^{n} I_j$$

$$H_j^* = \sum_{h=1}^{j} L_{h-1,h}^2 \dot{q}_h^2 + 2 \sum_{r=2}^{j} \sum_{s=1}^{r-1} L_{s-1,s} L_{r-1,r} \dot{q}_s \dot{q}_r$$

As seen above, we can find H_j^* the same way we found H in the kinematics part but this time for every joint j

So we finally have after solving the above equations

$$M(Q)\ddot{Q} = F(Q, \ddot{Q}, \Phi_{ext}, \Phi_{act}, t)$$

Block diagram to describe the process

Input
- Joint Coordinates (Q,Qd,njl,I_loc,Motors,h,w,Hg)
  njl = number of links Hg = Gravitational matrix w.r.t frame 0

Calculate I and $\hat{I}$
- $I_i = M_{0i} I_{loc} M_{oi}^T$ and $\hat{I}$ with the formula metioned above

Compute M
- $M = [m_{ij}] = trace\ \{L_i \hat{I}_h L_j^T\}$ where $h = \max(i,j)$

Calculate H_j^*
- Compute the matrix using the above formula.

Calculate F
- $F = [f_i] = \varphi_i + \{(\sum_{j=i}^{n} \widetilde{\Phi}_j) \bullet L_i\} + trace\{L_i \hat{I}_i H_g^T\} + \sum_{j=i}^{n} (trace\ \{L_i I_j H_j^{*T}\})$

Output
- F and M matrix

• Inverse Dynamics

We compute the actuator actions for any trajectory, the kinetic energy of the robot and the inertia opposing at each joint.

First, we have inertial matrix I_i(i) given for every joint i and relative to the local frame (i). But we want to work with the base frame (0). Then, we must express the inertial matrices w.r.t the base frame (0) as follow:

$$I_i^{(0)} = M_{0i} * I_i^{(i)} * M_{0i}^t$$

Here is the dynamic equilibrium equations of the rigid body i :

$$-\Phi_i^* + \Phi_{i+1}^* + \Phi_i = 0$$

- Φ_i^* : action matrix of i on i-1 -> driving force and constraint forces
- Φ_{i+1}^* : action matrix of i+1 on i -> driving force and constraint forces
- Φ_i : all other forces acting on i

$$\Phi_i \;= skew\{H_g I_i\}_A - skew\{H_{0i} I_i\}_B + \widetilde{\Phi}_{i\ C}$$

* A : weight force
* B : inertial forces
* C : other forces
* $skew\{H_a I_b\} \;= H_a I_b - I_b^t H_a^t$

Now we can compute the action matrix as follows:

$$\Phi_i^* = -skew\{H_{0i} I_i\} + skew\{H_g I_i\} + \Phi_{i+1}^* + \widetilde{\Phi}_i$$

In fact, the procedure is starting with i = 6 (i decreasing) and with $\Phi_{i+1}^* = 0$.

$$\Phi_i^* = -skew\{H_{0i} I_i\} + skew\{H_g I_i\} + \Phi_{i+1}^* + \widetilde{\Phi}_i$$

Finally, we can determine the driving force following this block diagram:

$$I_i^{(0)} = M_{0i} * \; I_i^{(i)} * M_{0i}^t$$

$$\Phi_i^* = -skew\{H_{0i} I_i\} + skew\{H_g I_i\} + \Phi_{i+1}^* + \widetilde{\Phi}_i$$

$\varphi_i = -\Phi_i^* \circ L_{i-1,i}$ using the pseudo-scalar product

Furthermore, it is possible to calculate the kinetic energy of every link as follows, using base frame (0):

$$E_{ci} = \frac{1}{2} trace\{W_{0i}^{(0)} * I_i^{(0)} * W_{0i}^{(0)t}\}$$

Calculation of equivalent inertia for each link

We know for a prismatic joint

Equivalent inertia = sum of the Inertial mass

And for revolute joint

Equivalent inertial = sum of the Inertial tensor J_{zz}

But we know $J_{zz} = I_{xx} + I_{yy}$

So, we use these conditions to determine the equivalent inertia
So for that we calculate an array of matries as follows

$$\begin{bmatrix} M_{01} & 0 & 0 & 0 & 0 & 0 \\ M_{02} & M_{12} & 0 & 0 & 0 & 0 \\ M_{03} & M_{13} & M_{23} & 0 & 0 & 0 \\ M_{04} & M_{14} & M_{24} & M_{34} & 0 & 0 \\ M_{05} & M_{15} & M_{25} & M_{35} & M_{45} & 0 \\ M_{06} & M_{16} & M_{26} & M_{36} & M_{46} & M_{56} \end{bmatrix}$$

And with that we calculate $I_{ik} = M_{ik} I_{loc\,k} M_{ik}^T$

$$\begin{bmatrix} I_{01} & 0 & 0 & 0 & 0 & 0 \\ I_{02} & I_{12} & 0 & 0 & 0 & 0 \\ I_{03} & I_{13} & I_{23} & 0 & 0 & 0 \\ I_{04} & I_{14} & I_{24} & I_{34} & 0 & 0 \\ I_{05} & I_{15} & I_{25} & I_{35} & I_{45} & 0 \\ I_{06} & I_{16} & I_{26} & I_{36} & I_{46} & I_{56} \end{bmatrix}$$

And we have an vector with equivalent inertia values for every link i

$$I_{eq\ prismatic}{}^{(i)} = \sum_{j=i+1}^{6} I_{xx}{}^{(i,j)} + I_{yy}{}^{(i,j)}$$

$$I_{eq\ revolute}{}^{(i)} = \sum_{j=i+1}^{6} M_{inertial}{}^{(i,j)}$$

Where for each link i and j =(i+1 → 6)

$I_{xx}{}^{(i,j)}$ is the (1,1) term of I_{ij}

$I_{yy}{}^{(i,j)}$ is the (2,2) term of I_{ij}

$M_{inertial}{}^{(i,j)}$ is the (4,4) term of I_{ij}

Inertial Matrix

- Using the transformation matrix equation we find the inetial matrix wrt to (0) frame
- $I_i^{(0)} = M_{0i} * I_i^{(i)} * M_{0i}^t$

Accerration due to gravity

- To find the acceration due to gravity wrt to base we use the same eqaution homogenous transformation and Acceleration due to gravity of that link
- $I_i^{(0)} = M_{0i} * H_i^{(i)} * M_{0i}^t$

Action matrix

- Computing the action matrix we huse the following law as sum of all the forces acting on the link is equal to zero
- $-\Phi_i^* + \Phi_{i+1}^* + \Phi_i = 0$

Action matrix Inertial Force

- By newton Second law of motion inertial matrix is given by = -skew(H_{0i}*I_i)

Action matrix Acceleration due to gravity

- Acceleration due to gravity force we calculate using the above method as = -skew(Hg*I_i)

Action Matrix

- Sum all the action matrix and in assign robot no other force are action so we have considered as zero (i.e only acceration due to gravity force is acting)
- Action matrix of the forward link is also taken into consideration . (Action $_{i+1}$)

Motor Matrix

- We have motor matrix as mutual action between the link i and i-1. It is given by Pseudo product of = -Action matrix * $L_{i-1,i}$.

Kinectic Energy

- Kinectic Energy is given by 1/2 trace {W_{0i} * I_I*W_{i0}^T }

Inertial Matrix

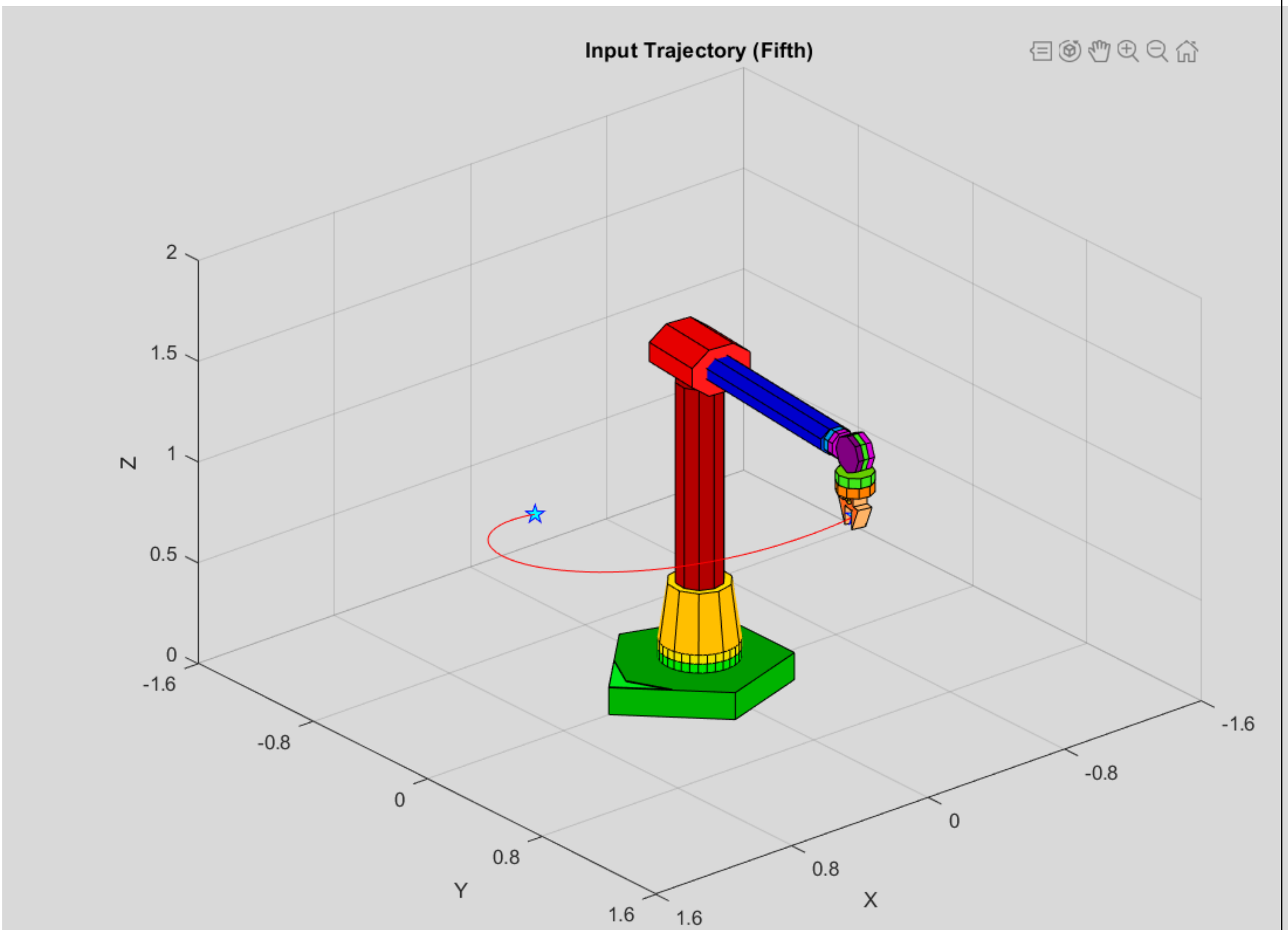
- To find the inertial matrix opposing the motor we have the following relation as mention above

Example of Dynamics

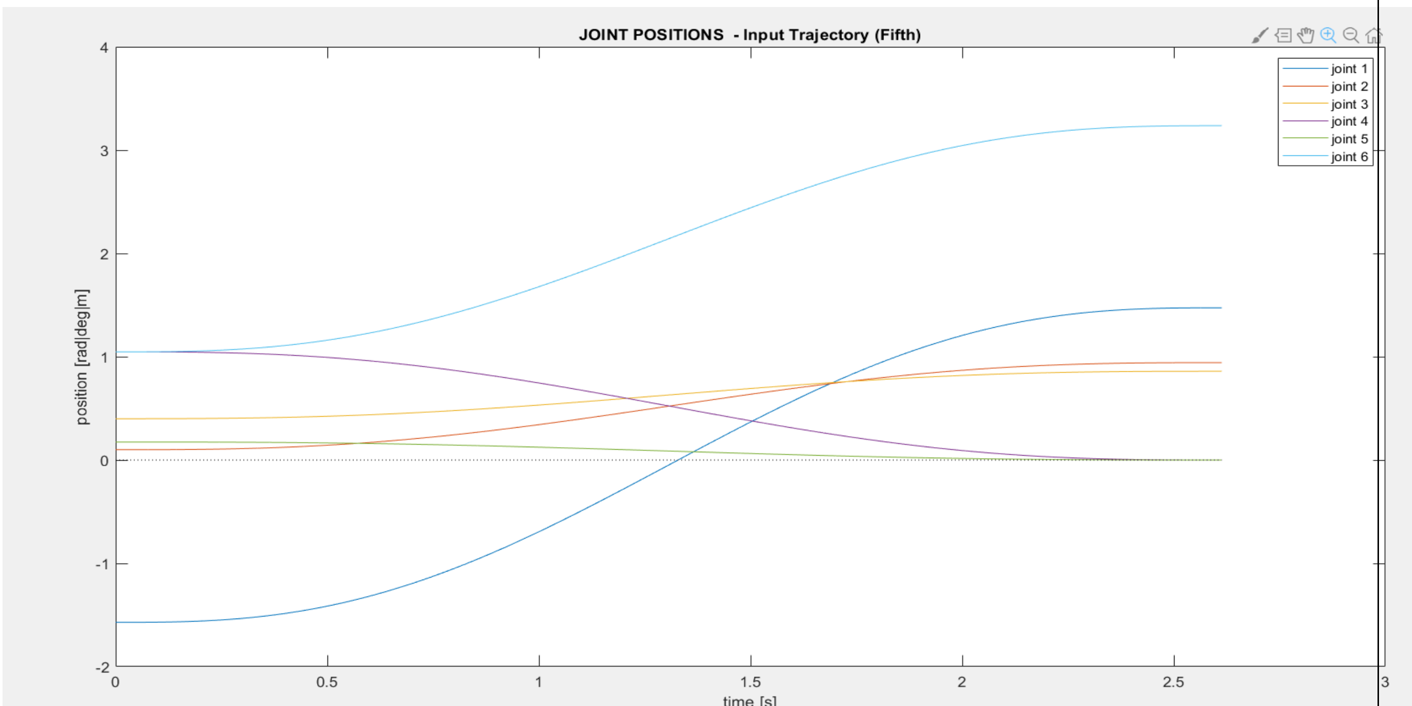
Cylindrical_Robot_Dynamics('Test_Dynamics', 1, 1,'XYZ','OpenGL',2000,[140 25], 'ode45', 0.01,[-90; 0.1*180/pi; 0.4*180/pi; 60; 50; 20]/180*pi, [0.12; 1.23; 1.30; [180;   0; 11]/180*pi],'Fifth', 5000)

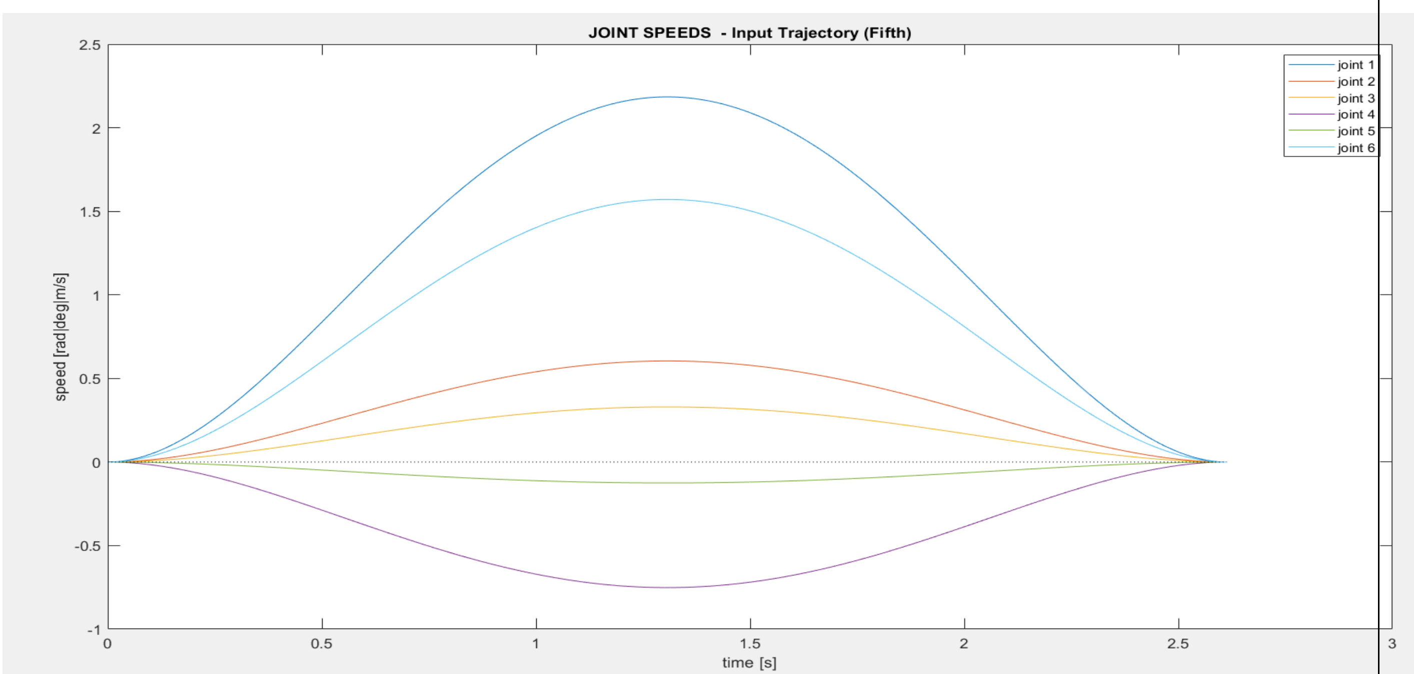
INPUT
payload -> ratio of robot payload -> 1 for full load (see function 'Link_InertialMatrix' in 'Cylindrical_Robot_Dynamics' library for absolute maximum value).
wrist -> wrist magnification factor (1 for the original size).
eul -> Euler angle kinds: 'XYZ', 'ZXZ' or 'ZYZ'
Renderer -> used for graphics. Try: 'ZBuffer', 'OpenGL' or 'Painter' (doc 'Figure Properties').
fig -> number of the figure for the first graph: only the values 1000, 2000, ... are valid.
ViewP -> [azimuth,elevation], vector 1x2 for setting the view point; doc 'view';
The next two arguments are for the Forward Dynamics and if they are input empty, only the Inverse Analysis is performored
ODEsolver -> Method for numerical integration: 'ode45', 'ode23', 'ode113' or 'Euler' (not reliable);
stepX -> Maximum time step in numerical integration as a ratio of the time span (1/minimum_number_of_steps)
The following arguments are for the function Cylindrical_Robot_Trajectory and are optional
Qin -> Column array for the Initial Configuration of the robot expressed in the joint space Q (degrees)
FinalPose -> Column array for the Final Goal Pose of the tool with Euler (eul) angles in degrees:
phi and psi must be limited between 0 and 360 degrees and chi between 0 and 180 degrees.
trajectory -> equation of motion: see Cylindrical_Robot_Trajectory help for trajectories available and strings of characters are accepted:
nf -> Number of frames for the animation.

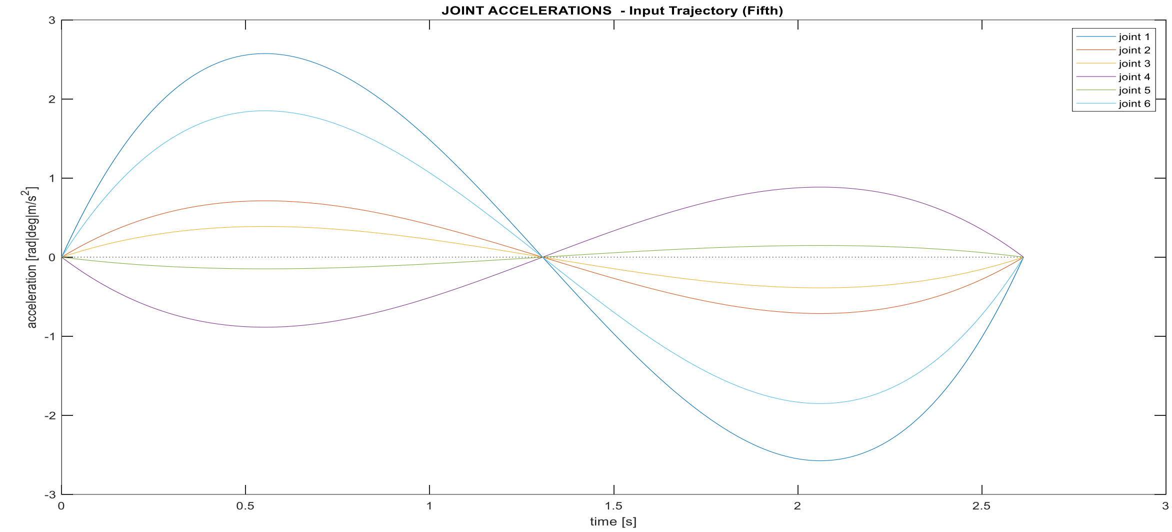
Trajectory.mat
Q,Qd,Qdd -> (number-of-joints x number-of-frames) arrays describing the tajectory in joint space.
execution_time -> time of execution of the trajectory.

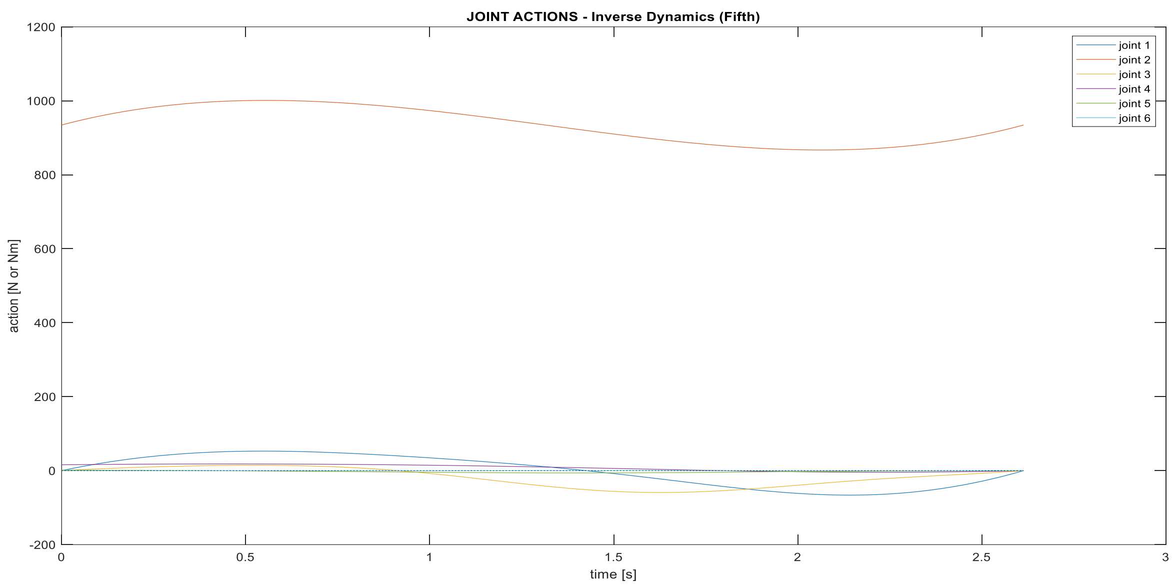


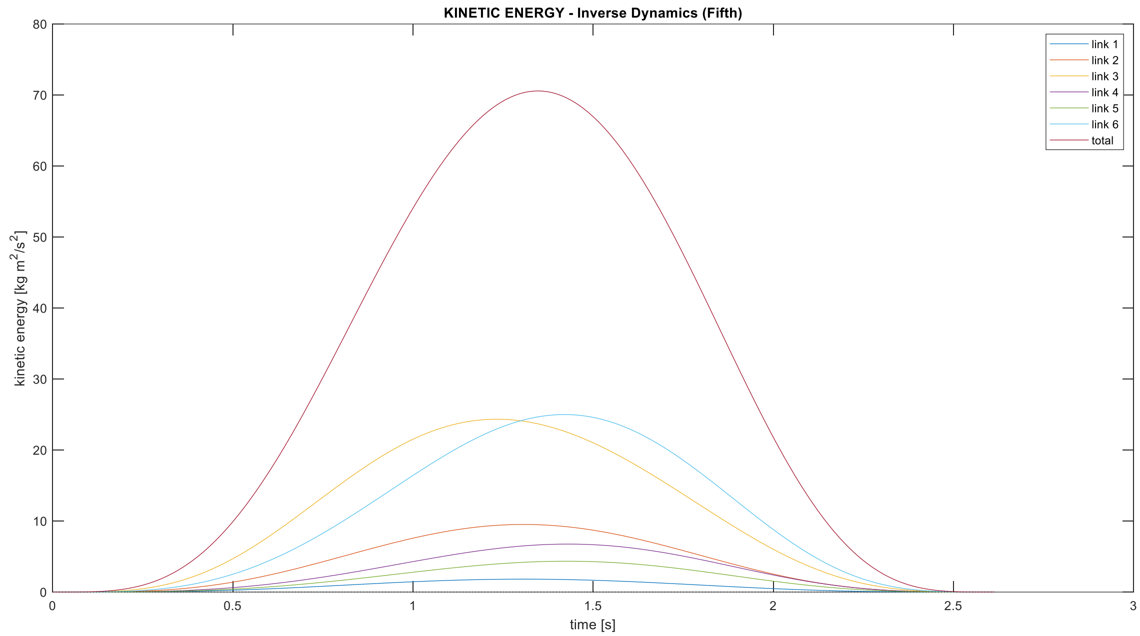
Input Trajectory (Fifth)
Z
2
1.5
1
0.5
0
-1.6
-0.8
0
0.8
1.6
Y
1.6
0.8
0
-0.8
-1.6
X

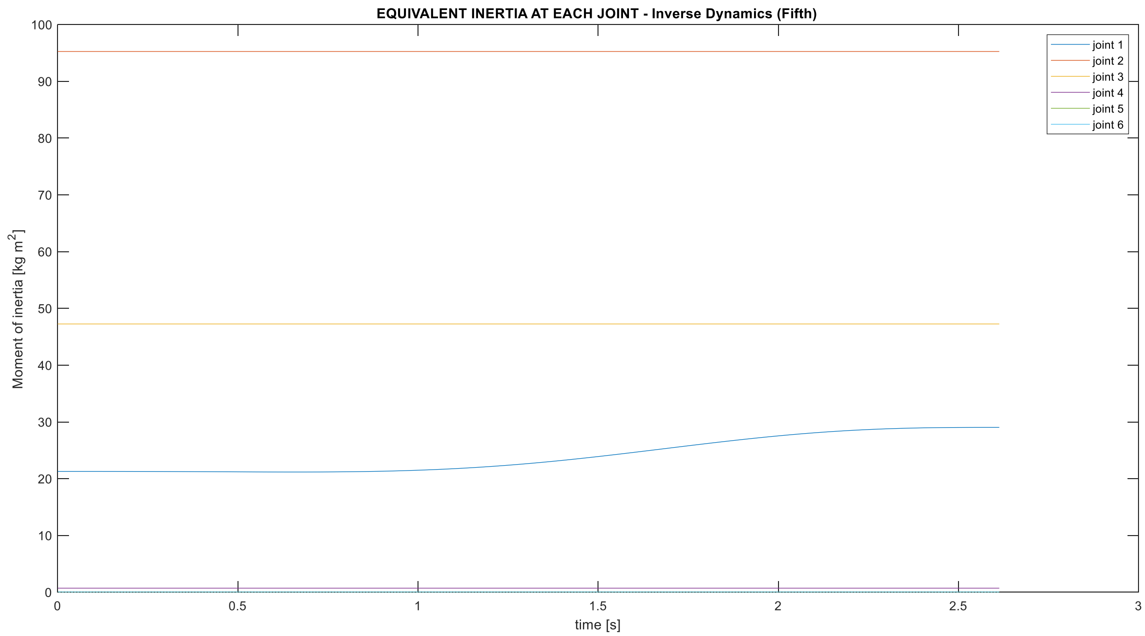

JOINT POSITIONS - Input Trajectory (Fifth)
position [rad|deg|m]
time [s]
joint 1
joint 2
joint 3
joint 4
joint 5
joint 6
4
3
2
1
0
-1
-2
0
0.5
1
1.5
2
2.5
3

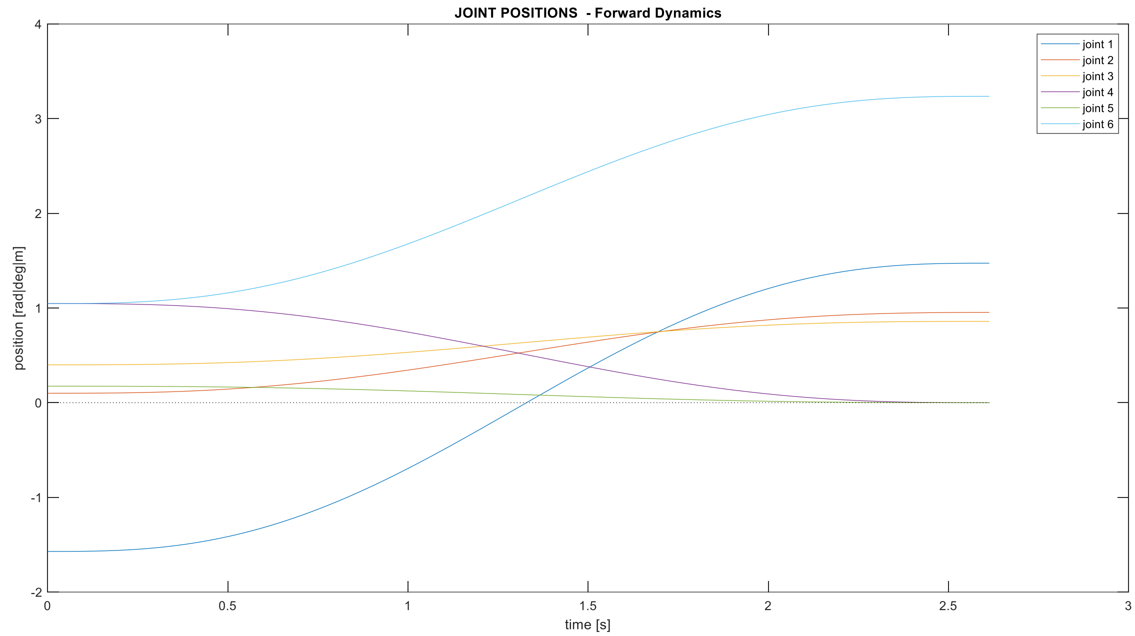

JOINT SPEEDS - Input Trajectory (Fifth)
speed [rad|deg|m/s]
time [s]
joint 1
joint 2
joint 3
joint 4
joint 5
joint 6
2.5
2
1.5
1
0.5
0
-0.5
-1
0
0.5
1
1.5
2
2.5
3

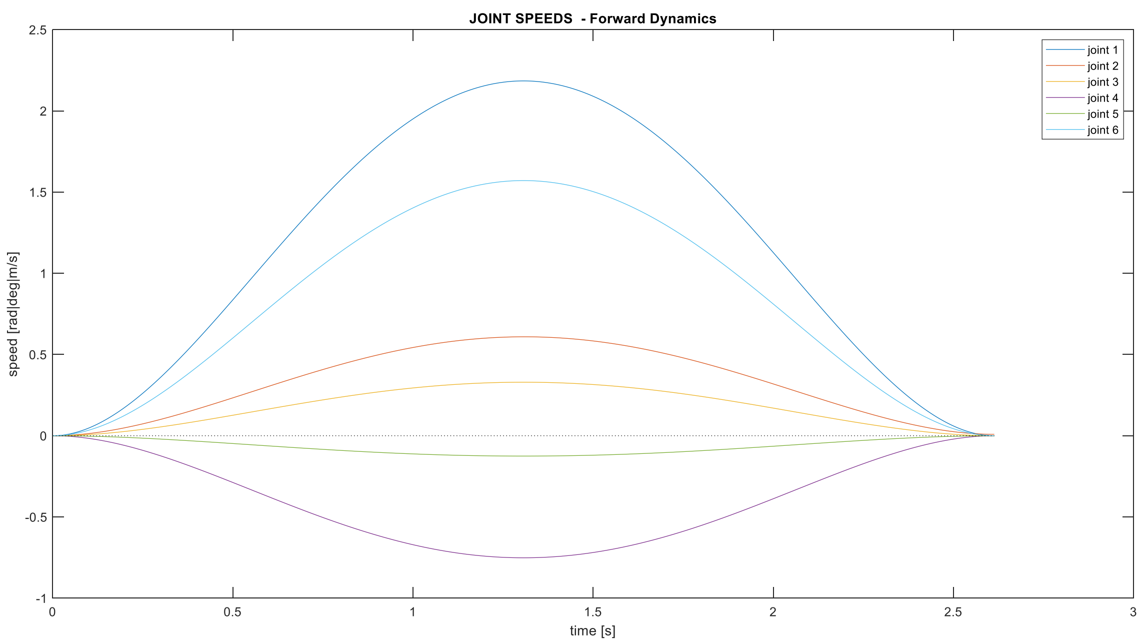

JOINT ACCELERATIONS - Input Trajectory (Fifth)
acceleration [rad|deg|m/s^2]
time [s]
joint 1
joint 2
joint 3
joint 4
joint 5
joint 6
3
2
1
0
-1
-2
-3
0
0.5
1
1.5
2
2.5
3

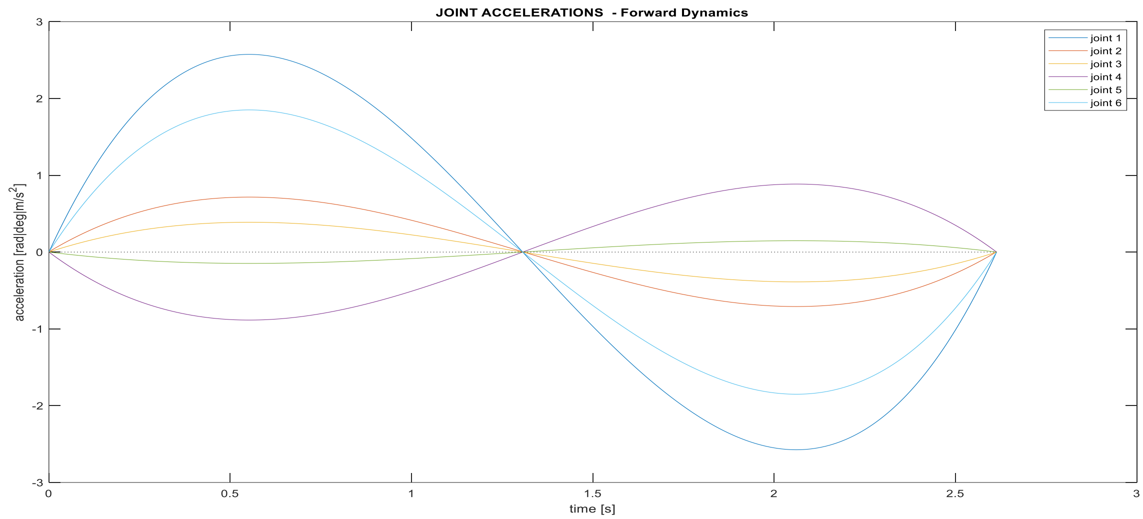

JOINT ACTIONS - Inverse Dynamics (Fifth)
action [N or Nm]
time [s]
joint 1
joint 2
joint 3
joint 4
joint 5
joint 6
1200
1000
800
600
400
200
0
-200
0
0.5
1
1.5
2
2.5
3


KINETIC ENERGY - Inverse Dynamics (Fifth)
kinetic energy [kg m^2/s^2]
time [s]
link 1
link 2
link 3
link 4
link 5
link 6
total


EQUIVALENT INERTIA AT EACH JOINT - Inverse Dynamics (Fifth)
Moment of inertia [kg m^2]
time [s]
joint 1
joint 2
joint 3
joint 4
joint 5
joint 6


JOINT POSITIONS - Forward Dynamics
position [rad|deg|m]
time [s]
joint 1
joint 2
joint 3
joint 4
joint 5
joint 6


JOINT SPEEDS - Forward Dynamics
speed [rad|deg|m/s]
time [s]
joint 1
joint 2
joint 3
joint 4
joint 5
joint 6


JOINT ACCELERATIONS - Forward Dynamics
acceleration [rad|deg|m/s^2]
time [s]
joint 1
joint 2
joint 3
joint 4
joint 5
joint 6
3
2
1
0
-1
-2
-3
0
0.5
1
1.5
2
2.5
3