Politecnico di Torino

A.Y. 2024/2025

Master degree in Automotive Engineering

Multibody Dynamics of a 5-DOF Robotic Arm

Simulation of a 5-axis robotic arm in MATLAB, with kinematic, dynamic, and trajectory analysis.

Students:

Marcello Marrella (S346927)

Giuseppe Martire (S344386)

Contents

Report 1

Adams

A process of modeling and simulation of a 5-axis robot with an articulated arm (3 d.o.f.) and a wrist (2 d.o.f.) is followed with the aim to analyze the kinematic and dynamic behavior of the end effector. The simulation was carried out using Adams software, which allows the modeling and dynamic analysis of multibody systems.

Figure 1.1 shows a 3D view with indication of the joints degrees of freedom q_1 to q_5, which are all rotative while figure 1.2 shows two views from which the dimensions of the links and the position of the joints can be deduced. Both figures depict the robot in the configuration with null joints degrees of freedom. Base 0 is fixed.

Joints angles, speeds and accelerations are saved in file trajectory1.mat, containing also the time vector. Angles are expressed in degrees.

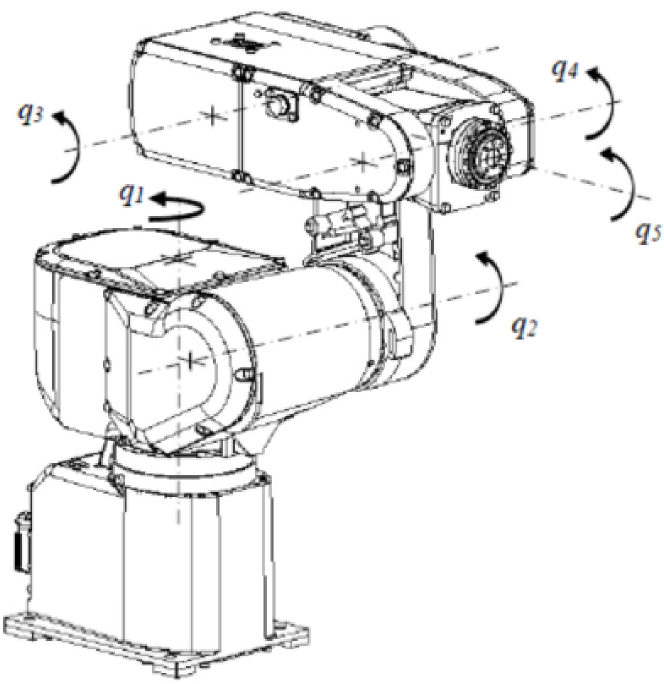


Figure 1.1: 5 dof robot

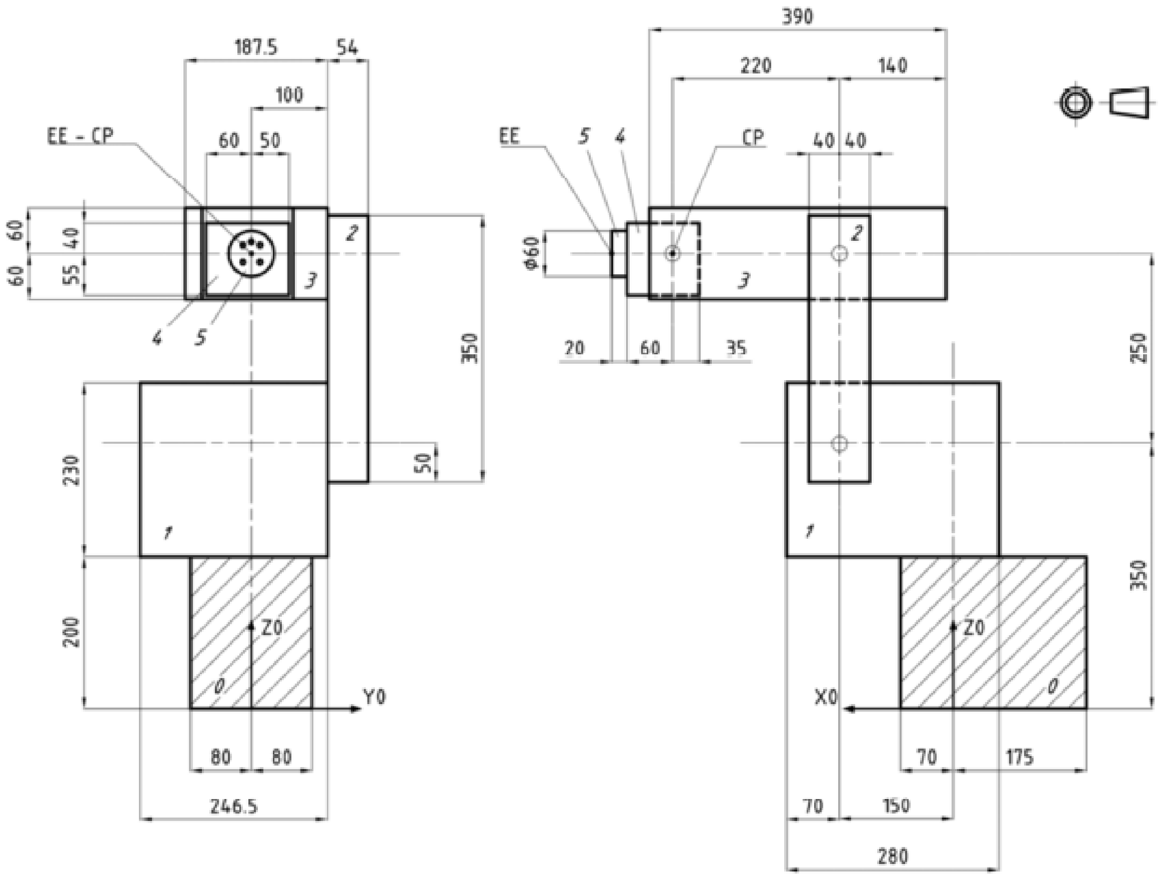


Figure 1.2: $y - z$ and $x - z$ views

The robot was constructed in Adams using the specific geometric and kinematic parameters for each joint. Subsequently, time-varying vectors, including displacement, velocity, and acceleration data for each joint, were provided as input. These inputs were used to generate the system's dynamic and kinematic behavior over time.

As a visual output of the simulation, the trace marker of the end effector trajectory was plotted, representing the path followed by the manipulator's end point.

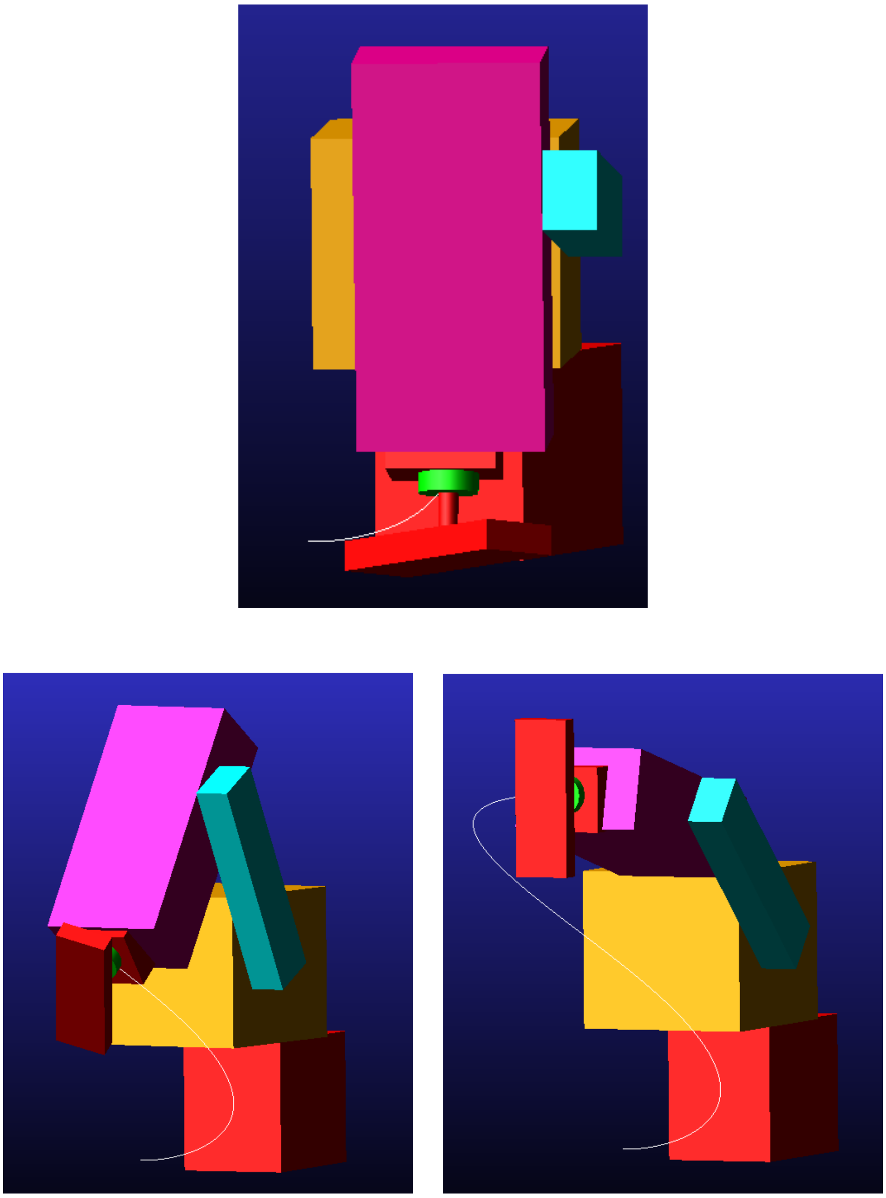


Figure 1.3: Trace marker of link 5

Figure 1.3 provides a clear and immediate visual representation of the trajectories traced by the end effector over three different instants of time.

Report 2

Transformation matricies

A triangle of vertex A, B and C is considered. The coordinates of the vertex of the triangle expressed in the fixed reference $0xyz$ are:

$$p_A = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} \quad p_B = \begin{bmatrix} 0 \\ 7 \\ 1 \end{bmatrix} \quad p_C = \begin{bmatrix} 0 \\ 4 \\ 7 \end{bmatrix} \quad p_{0_1} = \begin{bmatrix} 0 \\ 4 \\ 1 \end{bmatrix}$$

The mobile reference frame $0_1z_1x_1y_1$ is fixed in the middle of the base of the triangle.

2.1 Unit vectors of the reference frames

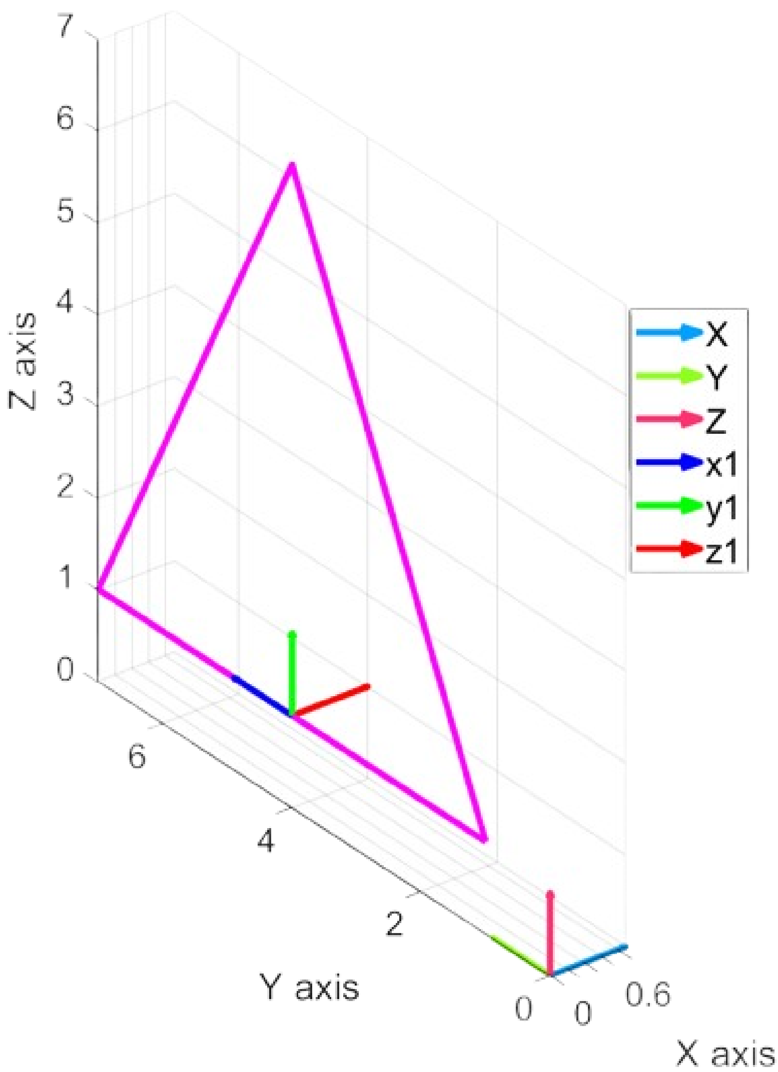
The unit vectors of the basement $0xyz$, the ones related to the local reference frame $0_1z_1x_1y_1$ and the triangle ABC are plotted using the following script.

MATLAB script

```matlab
% Triangle vertices are defined as column vectors, from the origin
    ↪ of the basement to the vertex of the triangle

p_a = [0 1 1]';
p_b = [0 7 1]';
p_c = [0 4 7]';

p_01 = [0 4 1]'; % distance vector between basement and reference
    ↪ frame 1

```

```
% X,Y,Z are vectors which contain the coordinate the vertex with resect to each axis of the basement
X = [p_a(1) p_b(1) p_c(1) p_a(1)];
Y = [p_a(2) p_b(2) p_c(2) p_a(2)];
Z = [p_a(3) p_b(3) p_c(3) p_a(3)];

%Representation of the triangle in the basement (as sum of polyline)
figure
plot3(X,Y,Z,'m','linewidth',3,'MarkerSize',8)
hold on
xlabel('X axis')
ylabel('Y axis')
zlabel('Z axis')
lgd=legend('Line 1','Line 2', 'Line 3');
lgd.FontSize=16;
grid on, zoom on
set(gca,'FontSize',16)
axis equal

%Representation of the basement unitary vectors i,j,k
scale=1;
i=[1 0 0 0]*scale;
j=[0 1 0 0]*scale;
k=[0 0 1 0]*scale;
p_i=quiver3(0,0,0,i(1),i(2),i(3), Color = "#00A0FF", Linewidth = 3),hold on
pj=quiver3(0,0,0,j(1),j(2),j(3), Color = "#90FF20", Linewidth = 3),hold on
pk=quiver3(0,0,0,k(1),k(2),k(3),Color = "#FF3070", Linewidth = 3),hold on
axis equal
set(gca,'FontSize',16)

% Representation of the local system unitary vectors u1,v1,w1
scale=1;
```

```
u1=[0 1 0 0]*scale;
v1=[0 0 1 0]*scale;
w1=[1 0 0 0]*scale;
pu1=quiver3(0,4,1,u1(1),u1(2),u1(3),'b','linewidth',3),hold on
pv1=quiver3(0,4,1,v1(1),v1(2),v1(3),'g','linewidth',3),hold on
pw1=quiver3(0,4,1,w1(1),w1(2),w1(3),'r','linewidth',3),hold on

axis equal
lgd=legend([ p_i(1) pj(1) pk(1) pu1(1) pv1(1) pw1(1)],
 ↪  'X','Y','Z','x1','y1','z1');
set(gca,'FontSize',16)
```



2.2 Positioning matrix ${}^0A_1$

The positioning matrix ${}^0A_1$ of reference frame $0_1z_1x_1y_1$ with respect to the basement is evaluated as it follows.

MATLAB script

```
p_01o = [0 4 1 1]';

A01o = [u1' v1' w1' p_01o];  % Homogeneous positioning matrix of
    ↪  local reference frame
A01= A01o(1:3, 1:3); % Positioning matrix of local reference frame

T0o = [p_a p_b p_c p_a;
        1   1   1   1]; % Position matrix of triangular geometry
            ↪  expressed in system 0
T1o = inv(A01o) * T0o; % Position matrix of triangular geometry
    ↪  expressed in system  1
```

$$
{}^0A_1 = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}
$$

$$
{}^0T = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 7 & 4 & 1 \\ 1 & 1 & 7 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \qquad {}^1T = \begin{bmatrix} -3 & 3 & 0 & -3 \\ 0 & 0 & 6 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}
$$

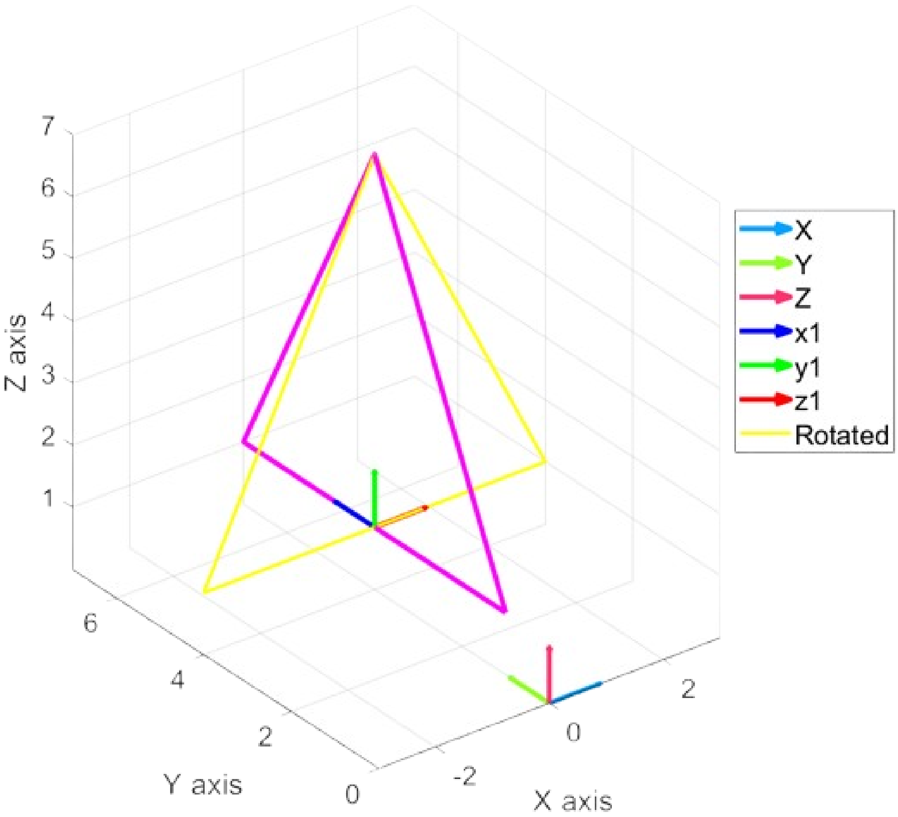
2.3 $90°$ rotation of the initial triangle about axis y_1

MATLAB script

```
% Homogeneous rotation matrix is defined
Rot_y_90 = [cos(pi/2) 0 sin(pi/2)   0;
                0       1      0       0;
             -sin(pi/2) 0 cos(pi/2)   0;
                0       0      0       1];

```

```
% Post-multiplication is needed to perform a rotation about an axis of the local reference system 1.
B01o = A01o * Rot_y_90;
T0_rotated_B = B01o * T1o;
% Representation of the rotated triangle (yellow)
plot3(T0_rotated_B(1,:),T0_rotated_B(2,:), T0_rotated_B(3,:), 'y','linewidth',2,'MarkerSize',6)
```

$$
{}^0\hat{B}_1 = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 4 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad {}^0T_{(B)} = \begin{bmatrix} 3 & -3 & 0 & 3 \\ 4 & 4 & 4 & 4 \\ 1 & 1 & 7 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}
$$



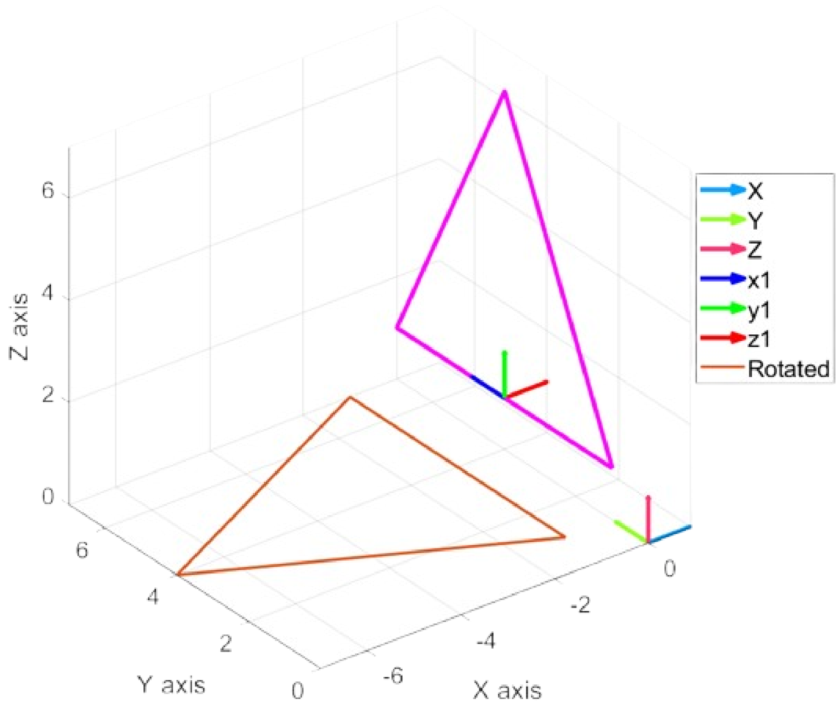
2.4 $-90°$ rotation of the initial triangle about axis y

While running the following script: script 2.1 and 2.2 must be considered and script 2.3 is neglected.

MATLAB script

```
% Homogeneous rotation matrix is defined
Rot_y_270 = [cos(3*pi/2) 0 sin(3*pi/2)  0;
                         0      1       0      0;
                -sin(3*pi/2) 0 cos(3*pi/2)  0;
                         0      0       0      1];

% Pre-multiplication  is needed to perform a rotation about an axis of the basement
C01o = Rot_y_270 * A01o;
T0_rotated_C = C01o * T1o;

% Representation of the rotated triangle (orange)
plot3(T0_rotated_C(1,:),T0_rotated_C(2,:), T0_rotated_C(3,:), Color = "#D95319", Linewidth = 2 ,MarkerSize = 6)
```

$$
{}^{0}\hat{C}_1 = \begin{bmatrix} 0 & -1 & 0 & -1 \\ 1 & 0 & 0 & 4 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad {}^{0}T_{(C)} = \begin{bmatrix} -1 & -1 & -7 & -1 \\ 1 & 7 & 4 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}
$$



2.5 90° rotation of the rotated triangle about axis x_2

While running the following script: script 2.1, 2.2, 2.4 must be considered and script 2.3 is neglected.

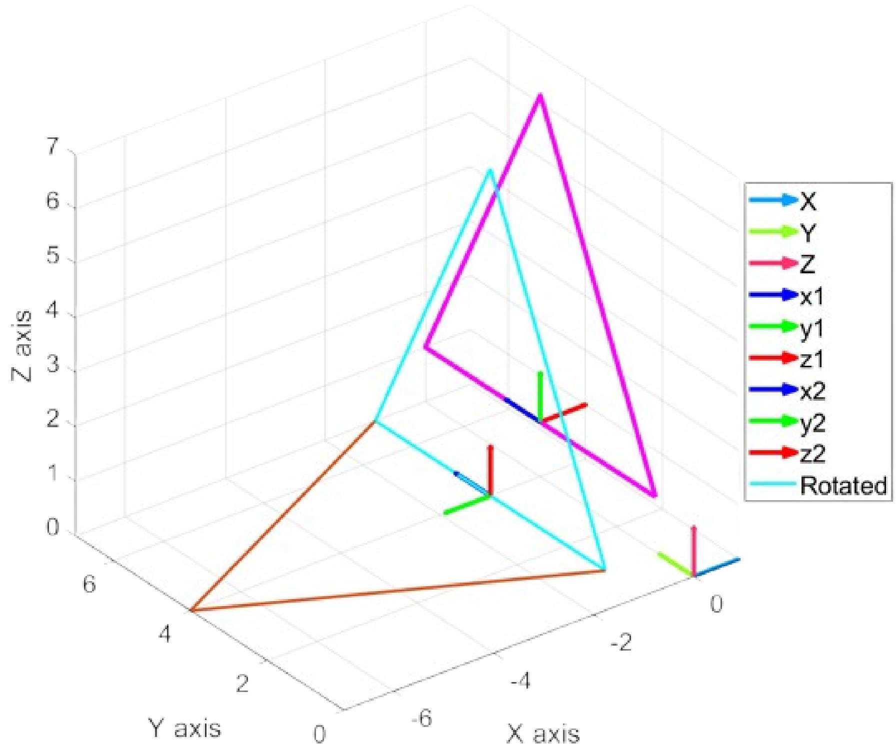
${}^0\hat{C}_1$ of the previous request is now the positioning matrix of reference system 2 before the wanted rotation: that's the reason why, for the request 5, ${}^0\hat{C}_1$ is going to be called ${}^0\hat{C}_2$.

MATLAB script

```matlab
C02o = C01o;
% Definition of the local system 2 with unitary vectors u2,v2,w2
scale=1;
u2=[0 1 0 0]*scale;
v2=[-1 0 0 0]*scale;
w2=[0 0 1 0]*scale;
pu2=quiver3(-1,4,0,u2(1),u2(2),u2(3), 'b','linewidth',3),hold on
pv2=quiver3(-1,4,0,v2(1),v2(2),v2(3),'g', 'linewidth',3),hold on
pw2=quiver3(-1,4,0,w2(1),w2(2),w2(3),'r','linewidth',3),hold on
axis equal

% Homogeneous rotation matrix
Rot_x_90 = [1       0          0           0;
            0    cos(pi/2)  -sin(pi/2)     0;
            0    sin(pi/2)  cos(pi/2)      0;
            0       0          0           1];

% Post-multiplication  is needed to perform a rotation around a
↪   local axis
D02o = C02o * Rot_x_90;
T2o = inv(C02o) * T0_rotated_C;
T0_rotated_D = D02o * T2o

% Representation of the rotated triangle
plot3(T0_rotated_D(1,:),T0_rotated_D(2,:),
↪  T0_rotated_D(3,:),'cyan','linewidth',2,'MarkerSize',6)
lgd.FontSize=16;
```

```
lgd=legend([p_i(1) pj(1) pk(1) pu1(1) pv1(1) pw1(1) pu2(1) pv2(1)
 ↪  pw2(1)], 'X','Y','Z','x1','y1','z1','x2','y2','z2');
set(gca,'FontSize',16)
```

$$
{}^{0}\hat{D}_1 = \begin{bmatrix} 0 & 0 & 1 & -1 \\ 1 & 0 & 0 & 4 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad {}^{0}T_{(D)} = \begin{bmatrix} -1 & -1 & -1 & -1 \\ 1 & 7 & 4 & 1 \\ 0 & 0 & 6 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}
$$



Report 3

Scara robot

3.1 Introduction

The following Scara Robot is considered:

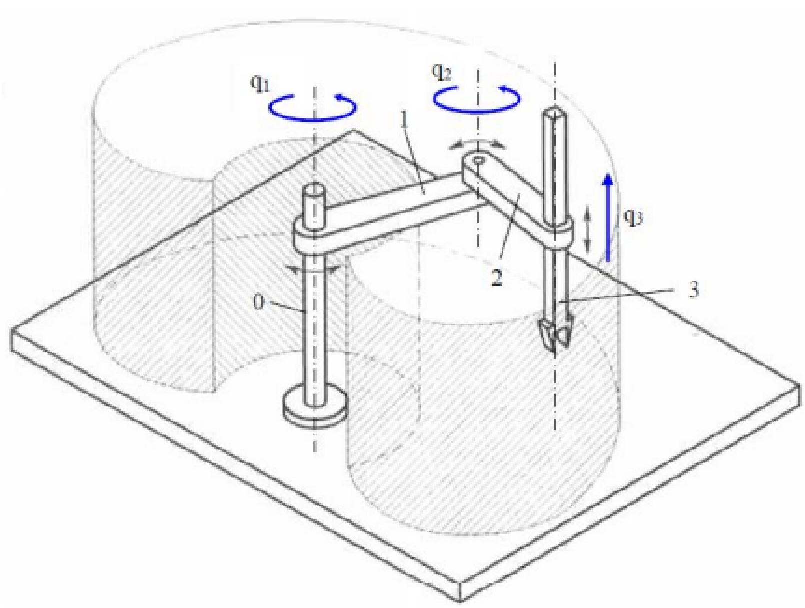


Figure 3.1: Scara robot

The trajectories of each joint are given in matrix M (file joints.mat) in which the vectors of displacements are memorized as column vectors in the following order q_1 (rad), q_2 (rad) and q_3 (mm).

$$q_1 = \begin{bmatrix} \dots \\ \dots \\ \dots \end{bmatrix} \qquad q_2 = \begin{bmatrix} \dots \\ \dots \\ \dots \end{bmatrix} \qquad q_3 = \begin{bmatrix} \dots \\ \dots \\ \dots \end{bmatrix}$$

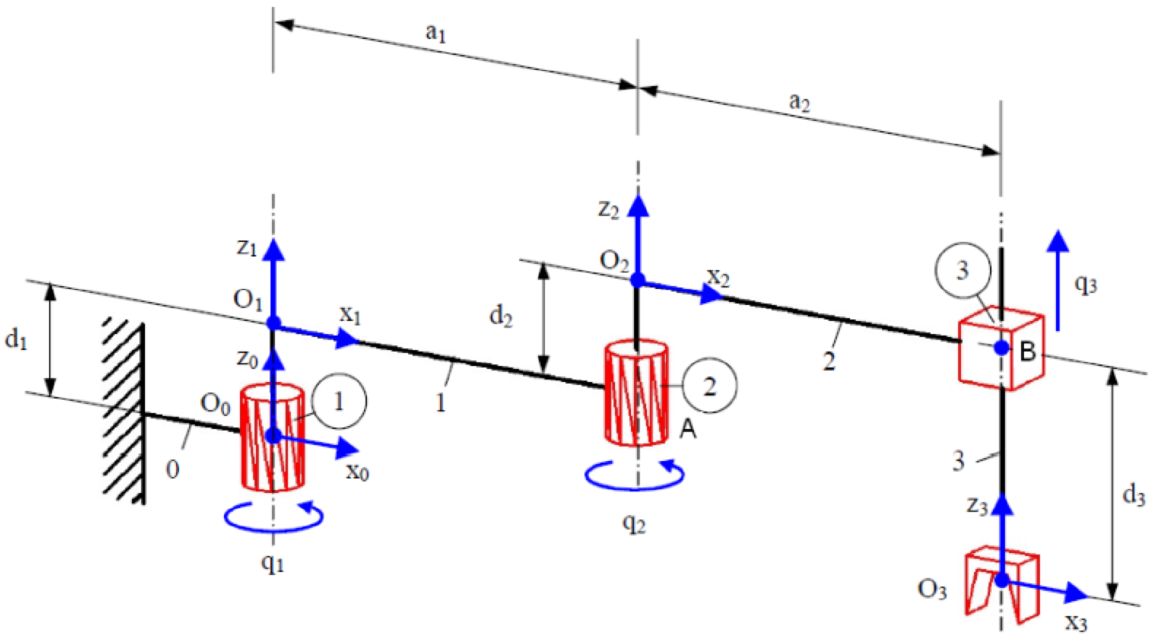


Figure 3.2: 2D representation of Scara robot

Parameter	**Value**
$a1$	$50\ mm$
$a2$	$10\ mm$
$d1$	$25\ mm$
$d2$	$15\ mm$
$d3$	$10\ mm$

Table 3.1: Dimensions

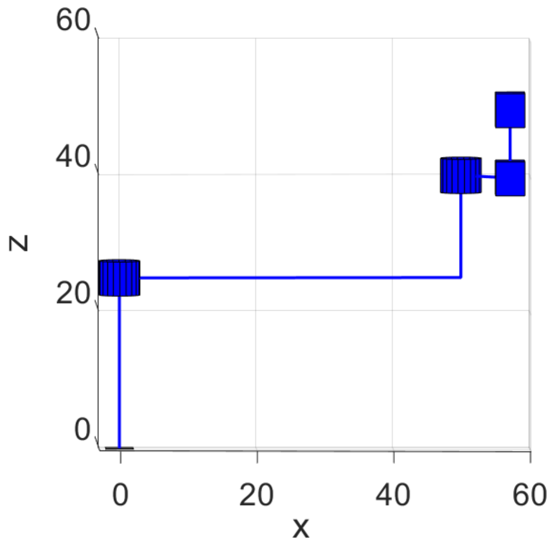
3.2 Plot of the wire frame $O_0O_1AO_2BO_3$ of the robot in the initial position

MATLAB script

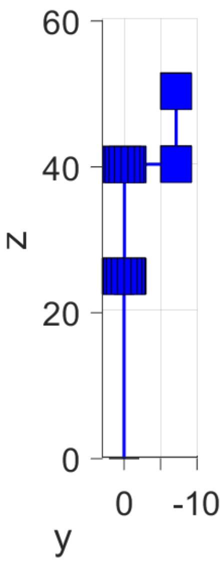
```
set(0,'DefaultFigureWindowStyle','docked')
set(0,'DefaultLineLineWidth',1.5)
set(0,'DefaultAxesFontSize',16)
% load the matrix of the joint variables
load joints

% data multibody system with D.H. convention
```

```
alfa1=0; a1=0; d1=25; %Denavit-Hartenberg's constant parameters of link 1 [rad][m][m] % k1=[0 0 1]'; %unit vector axis z1 expressed in frame 1 % delta1=0; %joint 1 revolute
alfa2=0; a2=50 ; d2= 15; %Denavit-Hartenberg's constant parameters of link 2 [rad][m][m] %k2=[0 0 1]'; %unit vector axis z2 expressed in frame 2 %delta2=0; %joint 2 revolute
alfa3=0; a3=10 ; d3= 10; %Denavit-Hartenberg's constant parameters of link 3 [rad][m][m] %k3=[0 0 1]'; %unit vector axis z3 expressed in frame 3 %delta3=1; %joint 3 prismatic

EE3o=[0 0 0 1]'; %position vector of point EE with respect to frame 3 expressed in frame 3 [m]

% wire-frame representation of the multibody system
% initial configuration
i=1;
q1=M(i,1); q2=M(i,2); q3=M(i,3);
P0o =[0 0 0 1]';   %position vector of the global frame origin 0 in the global frame 0
P21o=[a2 0 0 1]';   %position vector of the point P2 in the global frame 1
P42o=[a3 0 0 1]'; %position vector of the origin O3=P4 in the global frame 2

teta1=q1;
A10o=denhar_en01(alfa1,a1,d1,teta1);    %matrix 4x4
P1o=A10o(:,4);      %position vector of the origin O1=P1 in the global frame 0
P2o=A10o*P21o;      %position vector of the point P2 in the global frame 0

teta2=q2;
A21o=denhar_en01(alfa2,a2,d2,teta2);    %matrix 4x4
A20o=A10o*A21o;
A20=A20o(1:3,1:3);%matrix 3x3
```

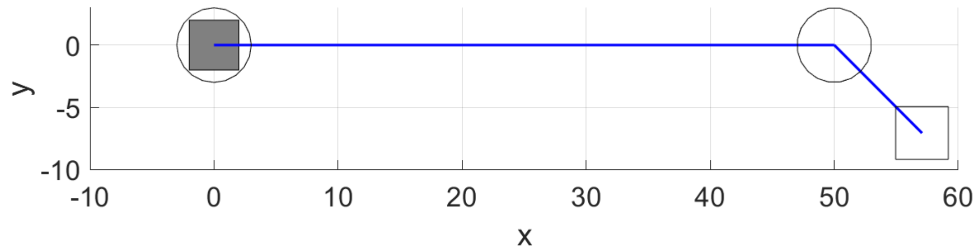
```
P3o=A20o(:,4);     %position vector of the origin O2=P3 in the
 ↪  global frame 0
P4o=A20o*P42o;     %position vector of the origin P4 in the global
 ↪  frame 0

teta3= q3;
A32o=denhar_en01(alfa3,a3,d3,teta3); %matrix 4x4
A30o= A20o*A32o; % (homogeneous) matrix 4x4: positioning of ref
 ↪  frame 3 with respect to %frame 0
A30= A30o(1:3,1:3); % matrix 3x3
pEE0o= A30o(:,4); %position of point P5=EE in frame O0 end

punti_om=[P0o P1o P2o P3o P4o pEE0o];
punti=punti_om(1:3,:);

% plot initial configuration
figure('Name','Initial configuration')
hold on
plot3(punti(1,:),punti(2,:),punti(3,:),'b')
grid
xlabel('x');ylabel('y');zlabel('z')
joint_rev_01(3,5,20,A10o,'blu')
joint_rev_01(3,5,20,A20o,'blu')
A3g2o = denhar_en01(alfa3,a3,0,teta3);
A3g0o = A10o*A21o*A3g2o;
joint_rev_01(3,5,4,A3g0o,'blu')
joint_rev_01(3,5,4,A30o,'blu')

% plot base
fill3([-2 2 2 -2],[-2 -2 2 2],[0 0 0 0],[.5 .5 .5])
daspect([1 1 1]) % fix the ratio between axis dimensions
view([-39 16])   % specify camera position as [azimuth, elevation]
hold off
```



(a) $x - z$ view



(b) $y - z$ view



(c) $x - y$ view

Figure 3.3: 2D initial position of the robot

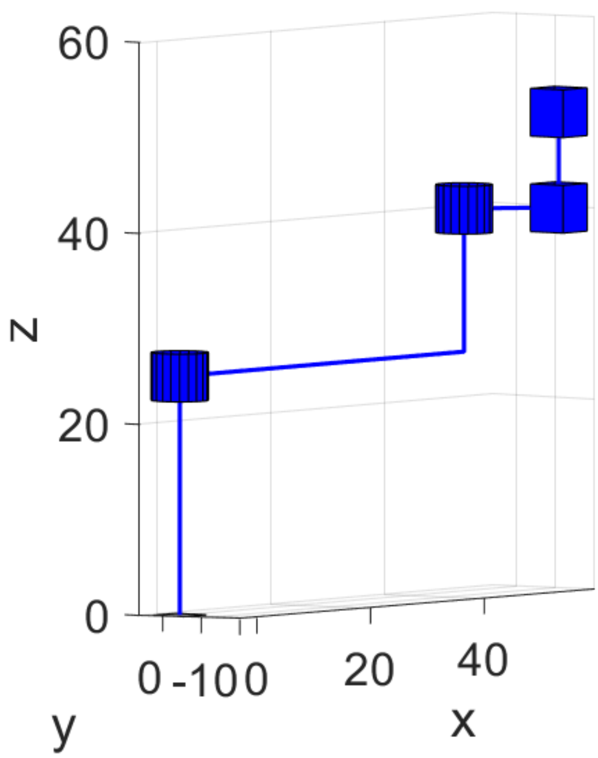


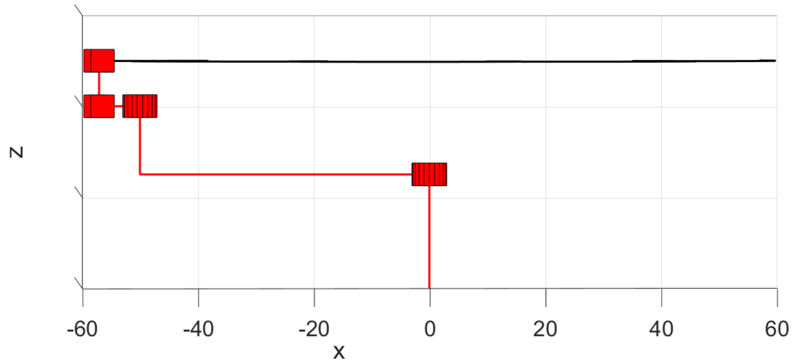
Figure 3.4: Isometric view of the final position

3.3 Trajectory plot of the end effector O_3 and final position of the robot

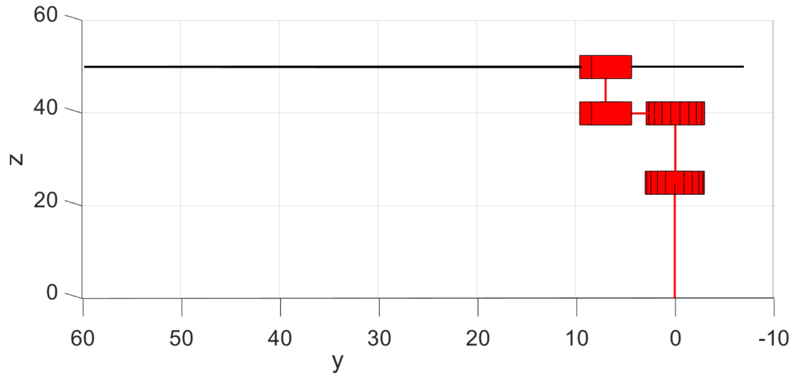
MATLAB script

```
%% computation of trajectory of point EE for i=1:1000 q1=M(i,1); q2=M(i,2); q3=M(i,3); % [rad], [rad],[m]
ANIMFIG = figure('Name','Animation');

for i=1:max(size(M)) % matrix 1000x3 of variables joint q1, q2, q3

    q1=M(i,1); q2=M(i,2); q3=M(i,3); % [rad], [rad],[m] % define the joint variables

    P0o=[0 0 0 1]';   %position vector of the global frame origin 0 in the global frame 0
    P21o=[a2 0 0 1]';   %position vector of the point P2 in the global frame 1
    P42o=[a3 0 0 1]'; %position vector of the origin O3=P4 in the global frame 2

    %transformation matrixes and vectors;
    teta1=q1;
    A10o=denhar_en01(alfa1,a1,d1,teta1);    %matrix 4x4
    P1o=A10o(:,4);     %position vector of the origin O1=P1 in the global frame 0
    P2o=A10o*P21o;     %position vector of the point P2 in the global frame 0

    teta2=q2;
    A21o=denhar_en01(alfa2,a2,d2,teta2);    %matrix 4x4
    A20o=A10o*A21o;
    A20=A20o(1:3,1:3);%matrix 3x3
    P3o=A20o(:,4);    %position vector of the origin O2=P3 in the global frame 0
    P4o=A20o*P42o;    %position vector of the origin P4 in the global frame 0
```

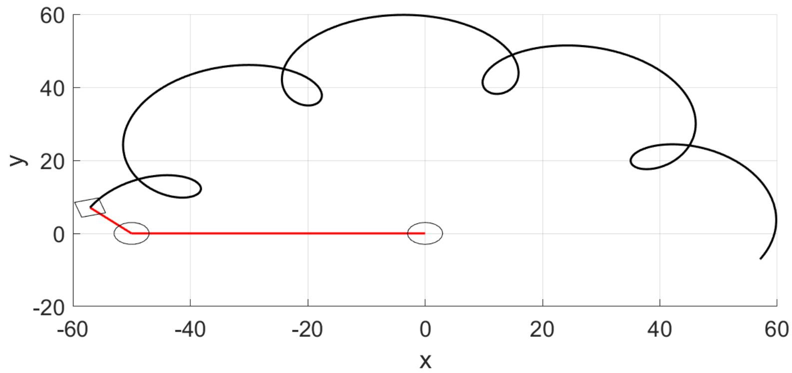
```

    teta3= q3;
    A32o=denhar_en01(alfa3,a3,d3,teta3);    %matrix 4x4
    A30o= A20o*A32o; % (homogeneous) matrix 4x4: positioning of ref
     ↪  frame 03 with respect to %frame 00
    A30= A30o(1:3,1:3); % matrix 3x3
    pEE0o= A30o(:,4); %position of point P5=EE in frame 00 end

    punti_om=[P0o P1o P2o P3o P4o pEE0o];
    punti=punti_om(1:3,:);

    %variables storage
    pEE0vect(:,i)=pEE0o(1:3);

    % plot trajectory and final configuration
    plot3(punti(1,:),punti(2,:),punti(3,:),'r')
    hold on
    plot3(pEE0vect(1,:), pEE0vect(2,:), pEE0vect(3,:),'k');
    grid
    xlabel('x');ylabel('y');zlabel('z')
    joint_rev_01(3,5,20,A10o,'red')
    joint_rev_01(3,5,20,A20o,'red')
    A3g2o = denhar_en01(alfa3,a3,0,teta3);
    A3g0o = A10o*A21o*A3g2o;
    joint_rev_01(3,5,4,A3g0o,'red')
    joint_rev_01(3,5,4,A30o,'red')
    view([-39 16])
    hold off
    pause (0.0000000001)
    drawnow
end
rmpath('MatlabFunctions')
```



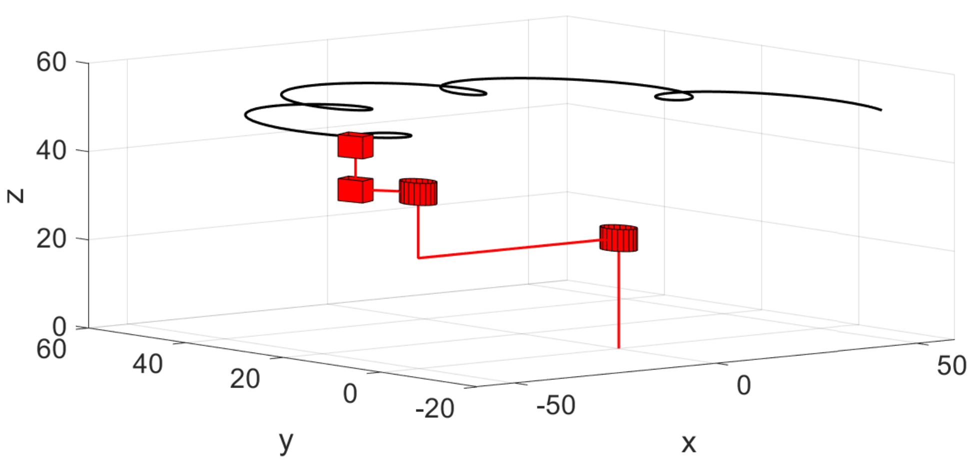
(a) $x - z$ view



(b) $y - z$ view



(c) $x - y$ view



(d) Isometric view

Figure 3.5: Final position of the robot and trajectory plot

Report 4

5 DOF robot

The 5-axis robot (previously described for report 2) with an articulated arm (3 d.o.f.) and a wrist (2 d.o.f.) is considered. The following hypotheses are assumed:

- Base 0 is taken equivalent to a parallelepiped with homogeneous distributed mass m0 = 18 kg.
- Body 1 is taken equivalent to a parallelepiped with homogeneous distributed mass equal to m1 = 10.5 kg.

  Body 1 attached to the base 0 by means of a revolute joint with vertical axis and d.o.f. q1.
- Body 2 is taken equivalent to a parallelepiped with homogeneous distributed mass equal to m2 = 2 kg.

  Body 2 attached to the body 1 by means of a revolute joint with horizontal axis and d.o.f. q2.
- Body 3 is taken equivalent to a parallelepiped with homogeneous distributed mass equal to m3 = 6 kg.

  Body 3 attached to the body 2 by means of a revolute joint with horizontal axis and d.o.f. q3.
- Body 4 is taken equivalent to a parallelepiped with homogeneous distributed mass equal to m4 = 2 kg.

  Body 4 attached to the body 3 by means of a revolute joint with horizontal axis and d.o.f. q4.

- Body 5 is taken equivalent to a cylinder with homogeneous distributed mass equal to m5 = 0.5 kg.

  Body 5 attached to the body 4 by means of a revolute joint with d.o.f. q5.

CP Wrist centre point at the intersection of axes 4 and 5.

EE End effector centre point at the distal end of body 5 in the center of the wrist interface plate.

A payload PL is attached to the wrist interface plate as figure 3 shows; to define its inertial characteristics the cylindrical portion is neglected and the payload is taken equivalent to a parallelepiped with homogeneous distributed mass equal to mL = 1 kg.

4.1 Reference frames

According with the DH convention, reference frames can be defined for each body and then represented on both views previously showed.

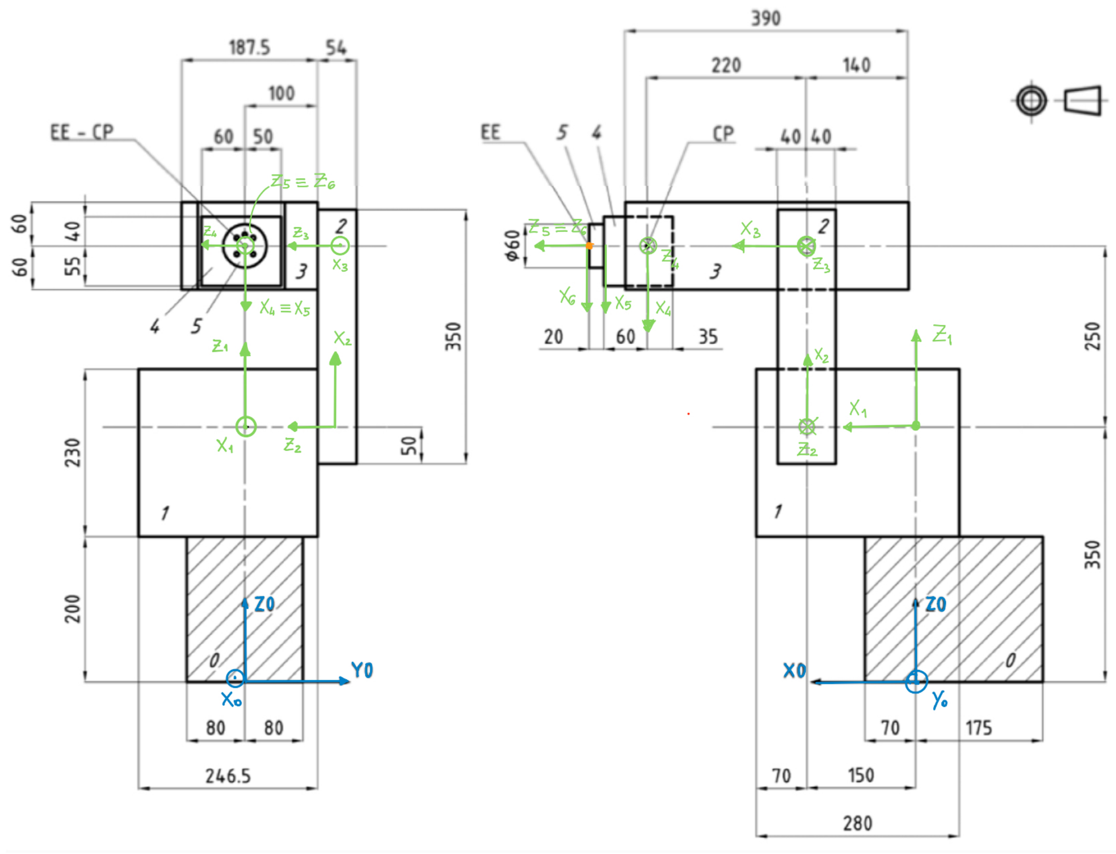


Figure 4.1: Reference frame

It can be observed that CP Wrist corresponds to the center of reference frame 4, while EE corresponds to the center of reference frame 6. This choice of reference frame allows to plot properly the trajectory of the two point, assuming they are coincident with two reference system already defined.

[m]	α	a	d	θ
1	0	0	0,35	q_1
2	90°	0,15	-0,127	90° + q_2
3	0	0,25	0	-90° + q_3
4	0	0,22	0,127	-90° + q_4
5	-90°	0	0,106	q_5
EE	0	0	0,102	0

Figure 4.2: Denavit-Hartenberg convention

4.2 Angles, speeds and accelerations of the joints

Considering a movement on a provided time vector of 2 seconds:

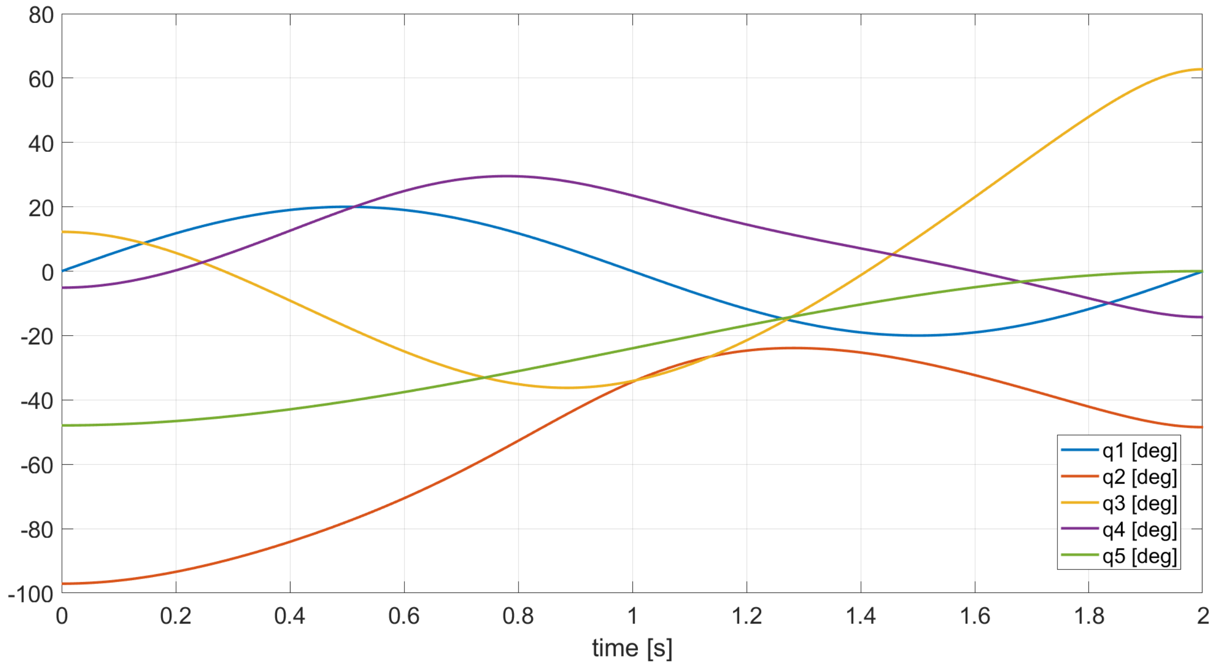


Figure 4.3: Angular displacement

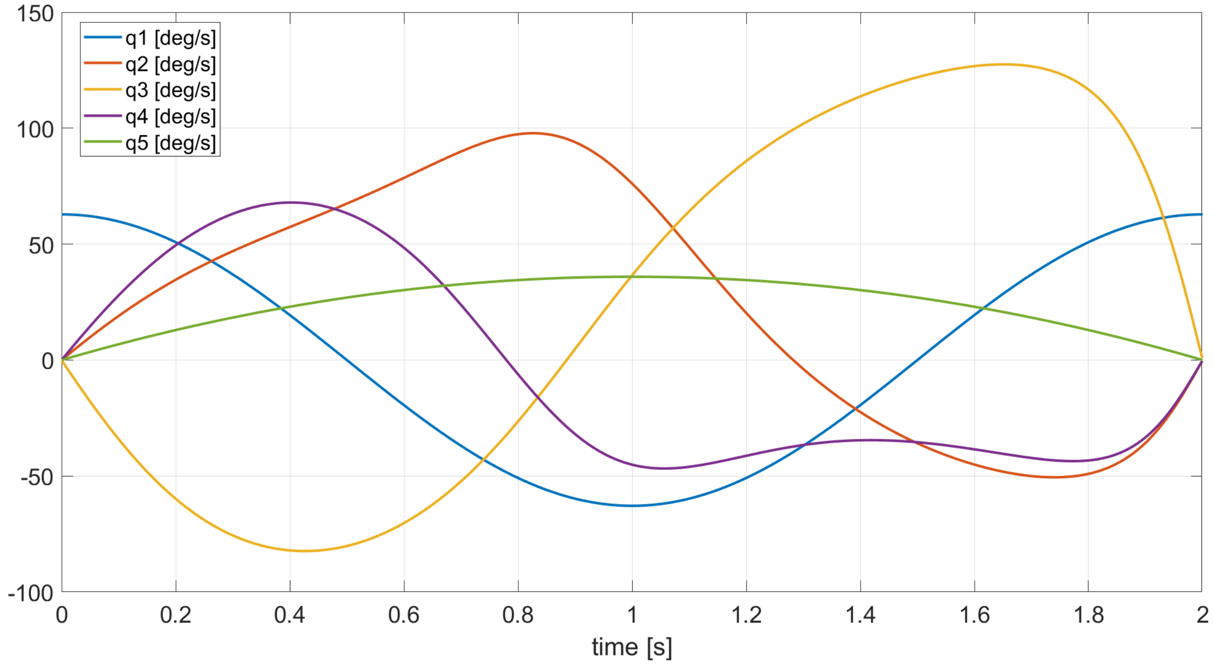


Figure 4.4: Angular speeds

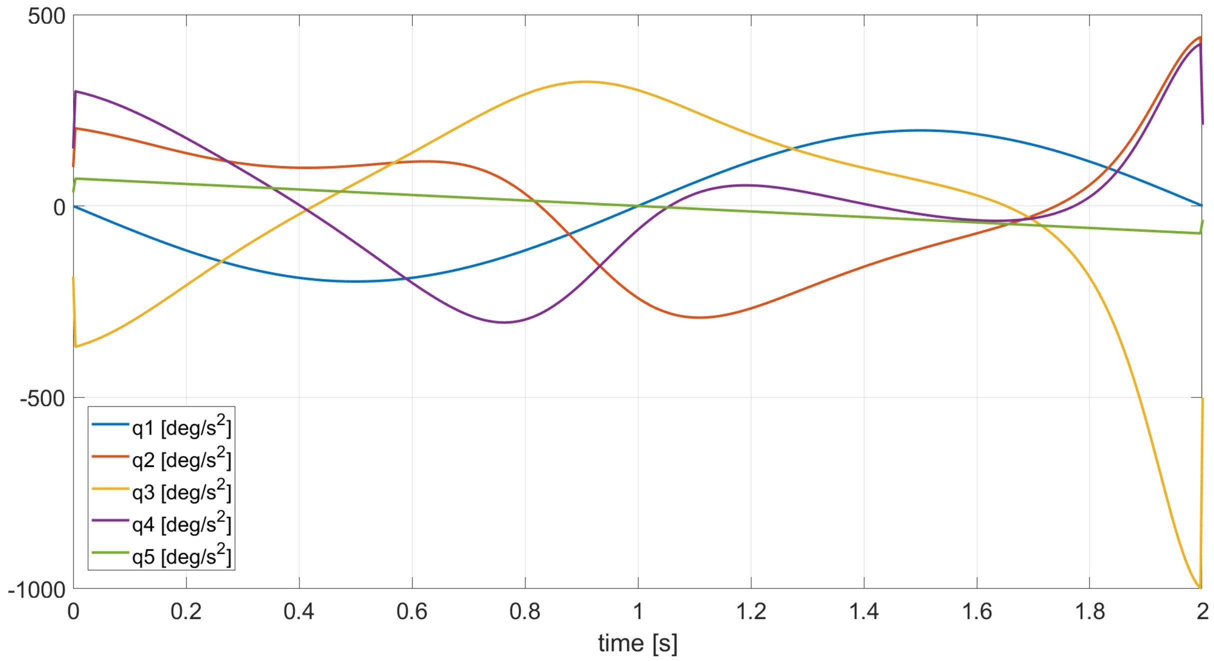


Figure 4.5: Angular accelerations

4.3 Initial and final positions of wire frame

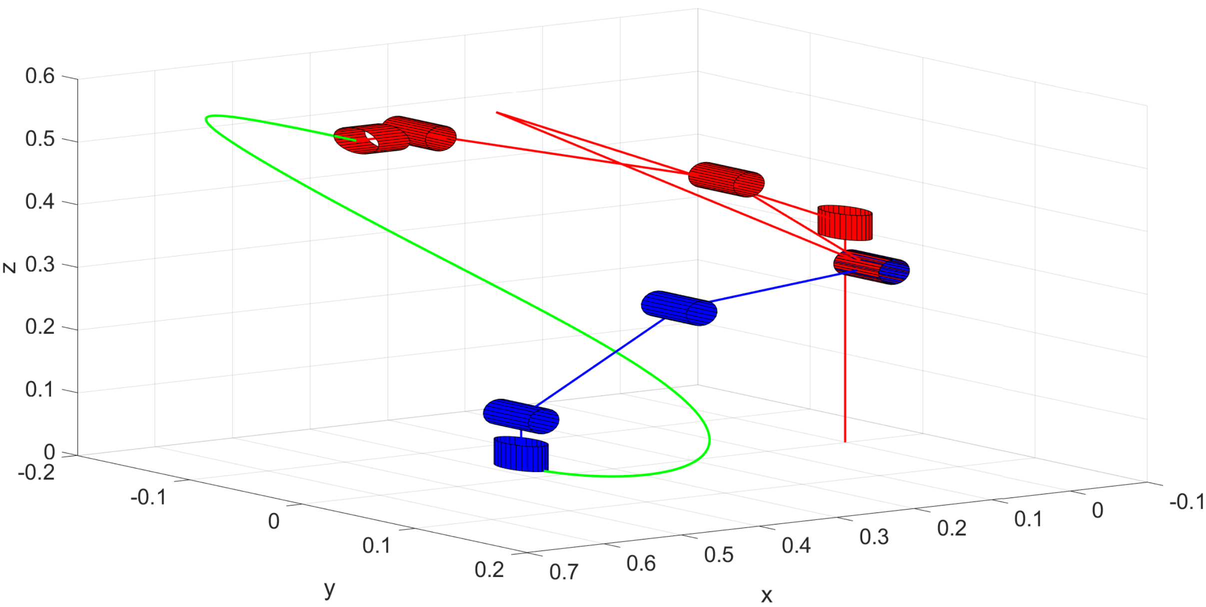


Figure 4.6: Initial (blue) and final (red) positions of the wire frame

It is appreciable that the trajectory plot of the payload (going from the initial to the final position) corresponds to the one already showed in figure 1.3.

Since time vector, positions, velocities, and accelerations are the same already provided for the Adams report, it is possible to use figure 1.3 as a counterproof of the solution.

4.4 Trajectory of CP and EE

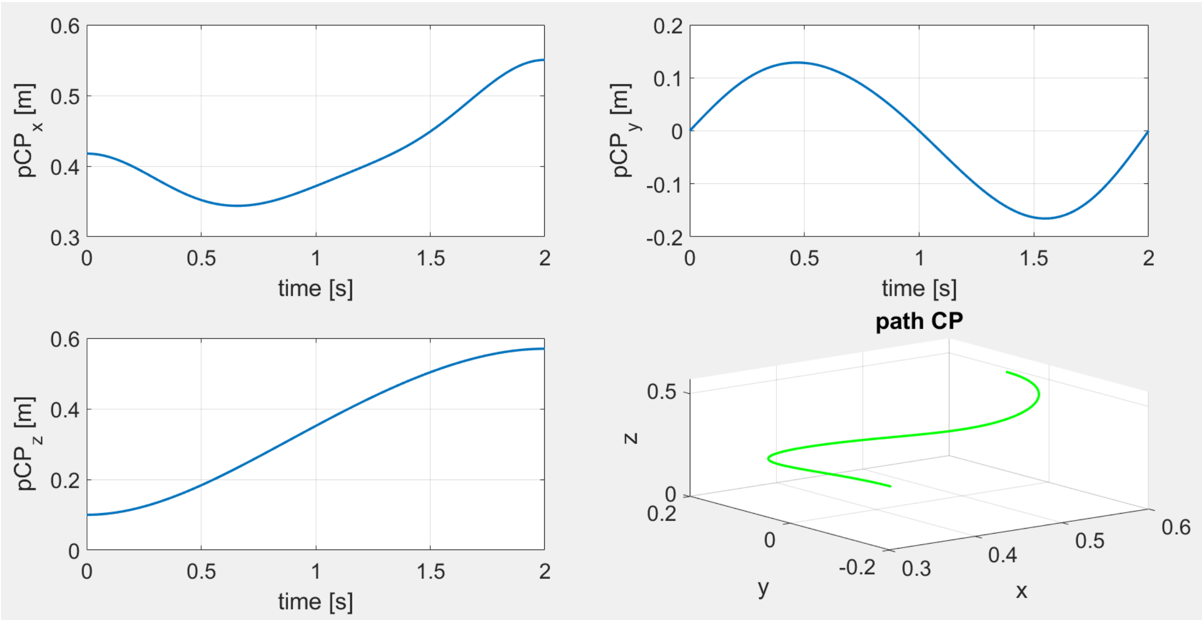


Figure 4.7: Trajectory of CP Wrist

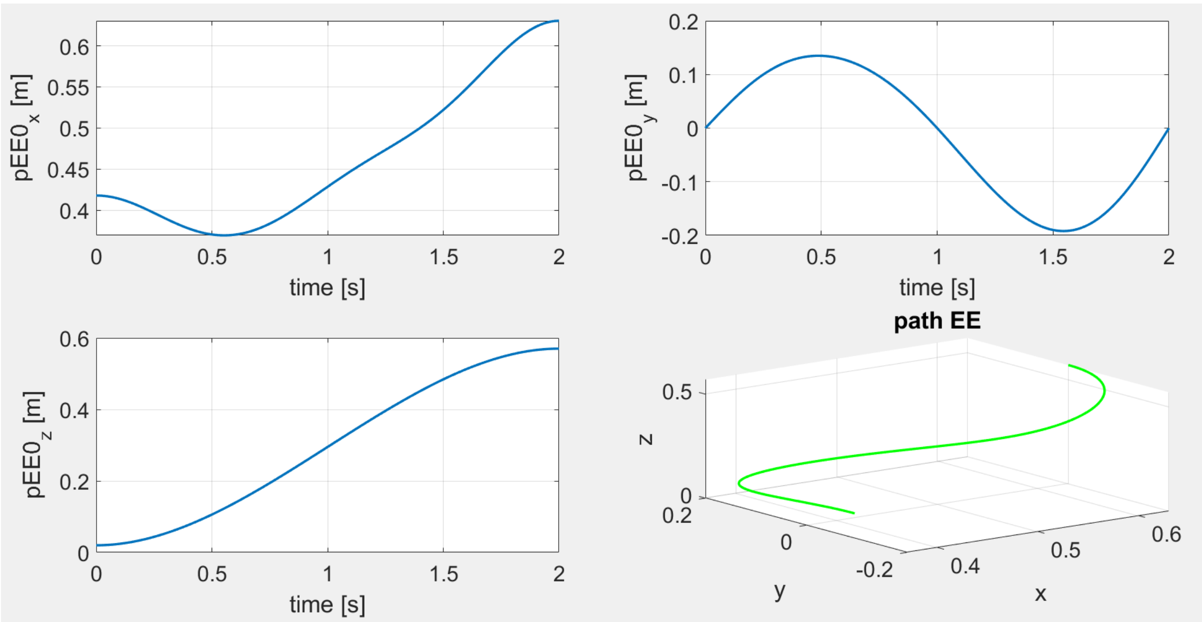


Figure 4.8: Trajectory of the payload attached to EE

4.5 Frame 5: center of mass

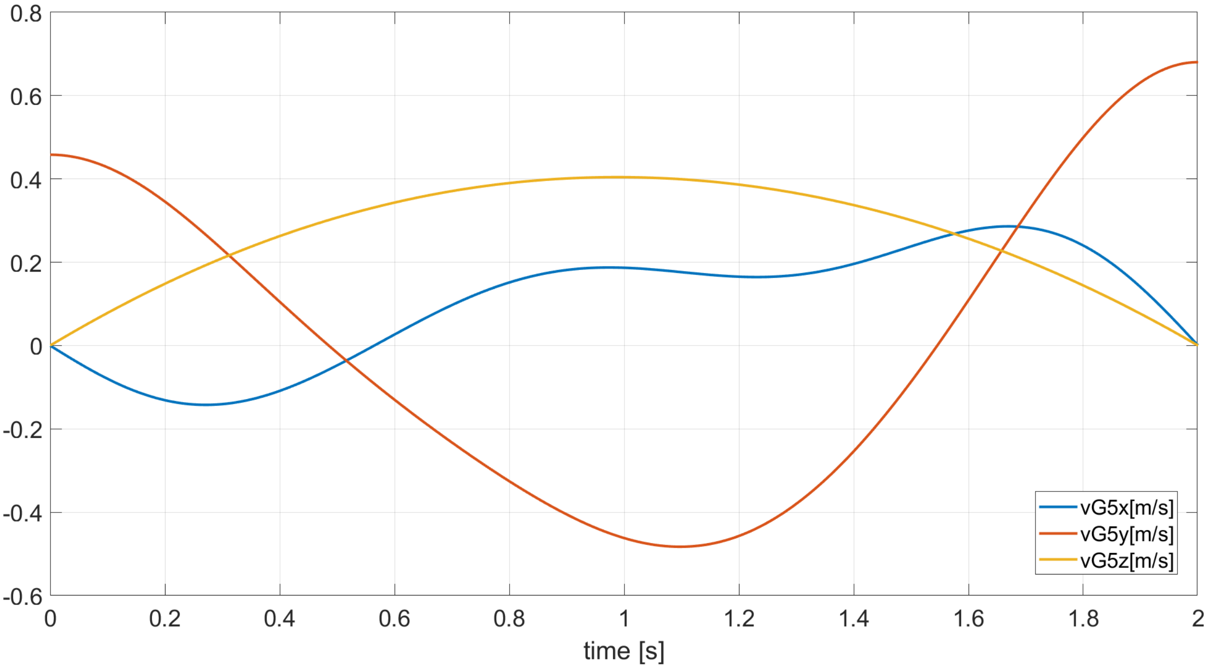


Figure 4.9: Speeds of center of mass for frame 5

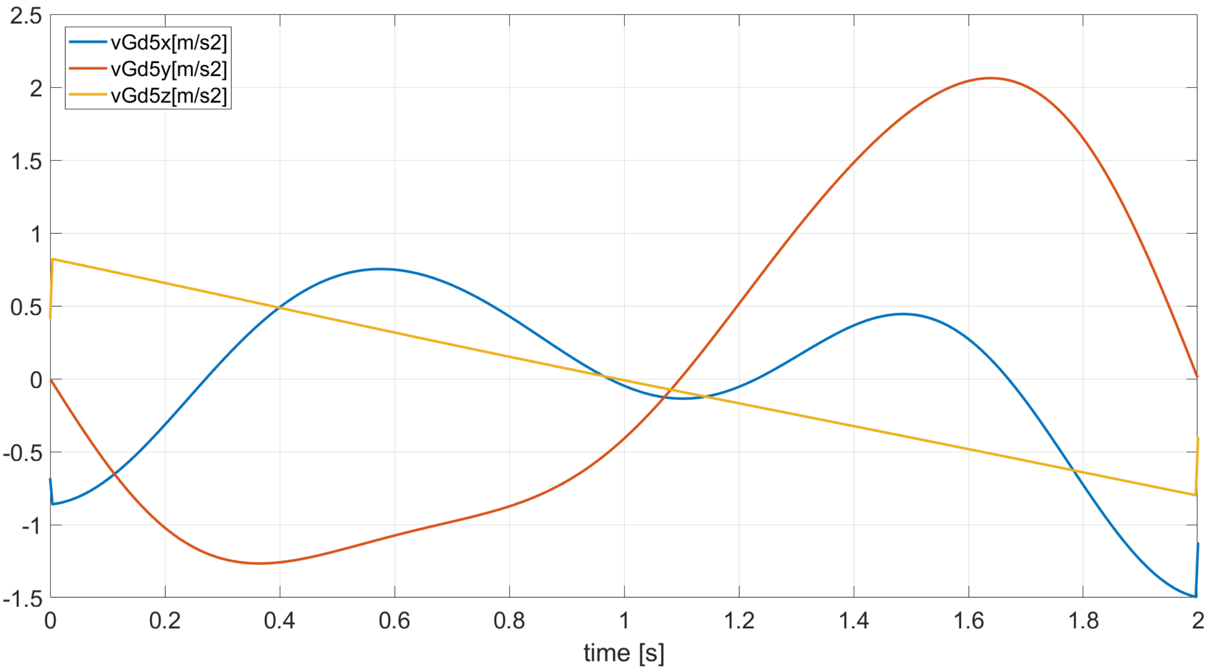


Figure 4.10: Accelerations of center of mass for frame 5

4.6 Inertial matrices of bodies and payload

For each body, U,V,W are the dimensions with respect to directions x,y,z of the body reference system.

MATLAB script

```
% mass of links [kg]
m0=18; m1=10.5; m2=2; m3=6; m4=2; m5=0.5; mL=1;

% Central inertial tensor:
U1=280e-3; V1=246.5e-3; W1=230e-3;% dimensions of link 1 [m] wrt
 ↪ his ref.system
I1=m1/12*diag([V1^2+W1^2 U1^2+W1^2 U1^2+V1^2]);% central inertial
 ↪ tensor link 1

U2=350-3; V2=80e-3; W2=54e-3; %dimensions of link 2 [m] wrt his
 ↪ ref.system
I2=m2/12*diag([V2^2+W2^2 U2^2+W2^2 U2^2+V2^2]);% central inertial
 ↪ tensor link 2

U3=390e-3; V3=120e-3; W3=187.5e-3;
I3=m3/12*diag([V3^2+W3^2 U3^2+W3^2 U3^2+V3^2]);

U4=95e-3; V4=95e-3; W4=110e-3;
I4=m4/12*diag([V4^2+W4^2 U4^2+W4^2 U4^2+V4^2]);

L5=20e-3; D5=60e-3; %cylindrical body
I5=m5*diag([((D5^2)/16)+((L5^2)/12)  ((D5^2)/16)+((L5^2)/12)
 ↪ D5^2/8]);

U6=230e-3; V6=80e-3; W6=30e-3; %payload attached to EE
I6=mL/12*diag([V6^2+W6^2 U6^2+W6^2 U6^2+V6^2]); %payload is assumed
 ↪ as a parallelepiped of mass mL
```

$$I_1 = \begin{bmatrix} 0.09945447 & 0 & 0 \\ 0 & 0.1148875 & 0 \\ 0 & 0 & 0.12176697 \end{bmatrix} \text{kg}\cdot\text{m}^2 \tag{4.1}$$

$$I_2 = \begin{bmatrix} 0.00155267 & 0 & 0 \\ 0 & 0.02055417 & 0 \\ 0 & 0 & 0.02113483 \end{bmatrix} \mathrm{kg{\cdot}m^2} \tag{4.2}$$

$$I_3 = \begin{bmatrix} 0.02477812 & 0 & 0 \\ 0 & 0.09362813 & 0 \\ 0 & 0 & 0.08325 \end{bmatrix} \mathrm{kg{\cdot}m^2} \tag{4.3}$$

$$I_4 = \begin{bmatrix} 0.00352083 & 0 & 0 \\ 0 & 0.00352083 & 0 \\ 0 & 0 & 0.00300833 \end{bmatrix} \mathrm{kg{\cdot}m^2} \tag{4.4}$$

$$I_5 = \begin{bmatrix} 0.00012917 & 0 & 0 \\ 0 & 0.00012917 & 0 \\ 0 & 0 & 0.000225 \end{bmatrix} \mathrm{kg{\cdot}m^2} \tag{4.5}$$

$$I_6 = \begin{bmatrix} 0.00060833 & 0 & 0 \\ 0 & 0.00448333 & 0 \\ 0 & 0 & 0.00494167 \end{bmatrix} \mathrm{kg{\cdot}m^2} \tag{4.6}$$

4.7 Actuator torques

Solving the inverse dynamics using function dynam_en02, it is possible to calculate the actuator torques $\tau_1, \tau_2, \tau_3, \tau_4, \tau_5$

MATLAB script

```
%inverse dynamics
[F6,M6]=dynam_en02([0 0 0]',[0 0 0]', mL, vGd6, w6, wd6, l6, b6,
    I6, eye(3),A06);
[F5,M5]=dynam_en02(F6,M6,m5,vGd5,w5,wd5,l5,b5,I5,A65,A05);  %
    applied force on body 5 by body 4
[F4,M4]=dynam_en02(F5,M5,m4,vGd4,w4,wd4,l4,b4,I4,A54, A04); %
    applied force on body 4 by body 3
[F3,M3]=dynam_en02(F4,M4,m3,vGd3,w3,wd3,l3,b3,I3,A43, A03); %
    applied force on body 3 by body 2
[F2,M2]=dynam_en02(F3,M3,m2,vGd2,w2,wd2,l2,b2,I2,A32, A02); %
    applied force on body 2 by body 1
```

```
[F1,M1]=dynam_en02(F2,M2,m1,vGd1,w1,wd1,l1,b1,I1,A21, A01); %
 ↪  applied force on body 1 by body 0 (basement)
I0=eye(3);
[F0,M0]=dynam_en02(F1,M1,m0,vGd0,w0,wd0,l0,b0,I0,A10, eye(3)); %
 ↪  applied force on basement by the floor

tau1(i)=k1'*F1*delta1+k1'*M1*(1-delta1);
tau2(i)=k2'*F2*delta2+k2'*M2*(1-delta2);
tau3(i)=k3'*F3*delta3+k3'*M3*(1-delta3);
tau4(i)=k4'*F4*delta4+k4'*M4*(1-delta4);
tau5(i)=k5'*F5*delta5+k5'*M5*(1-delta5);
```

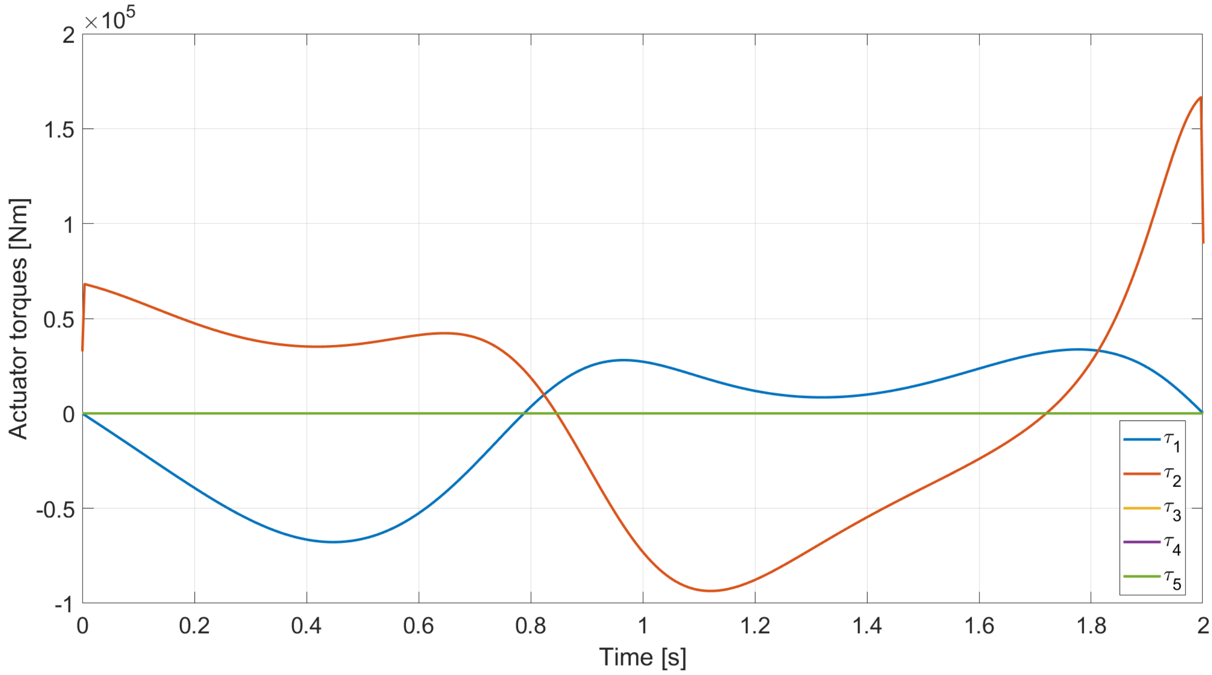


Figure 4.11: Actuator torques

4.8 Forces exerted from the basement

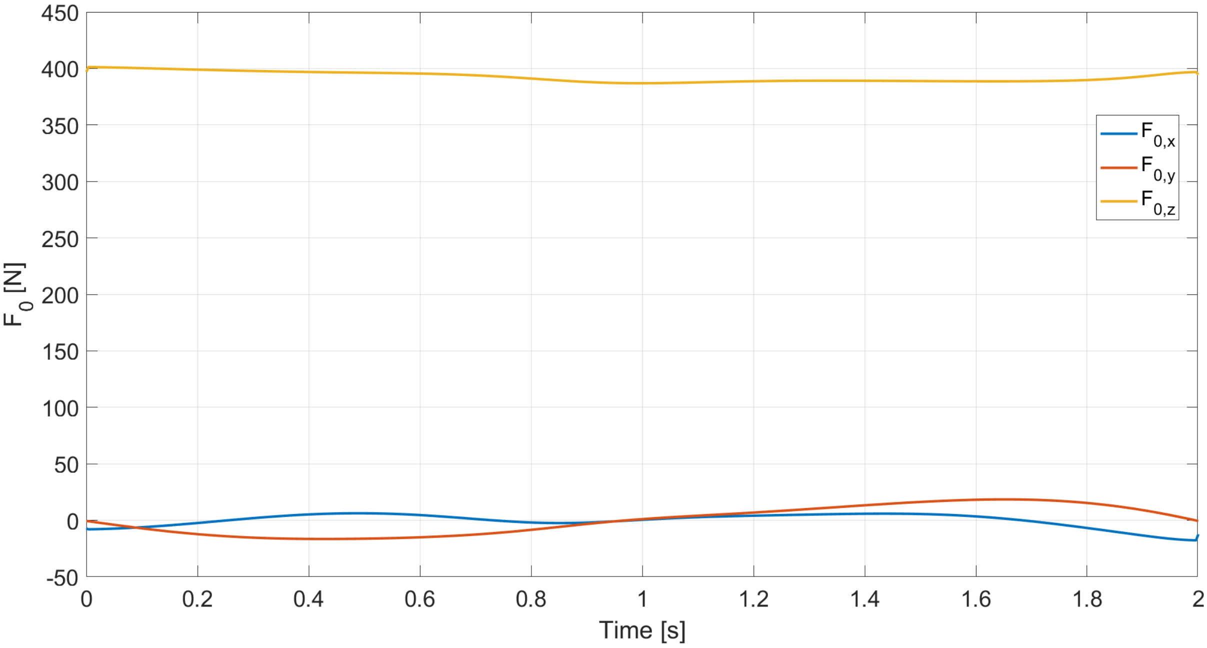


Figure 4.12: Forces exerted from the basement