MECH 544 - Robotics

Project III

Author:
Berk Güler (ID: 0077076)
Volkan Aydıngül (ID: 0075359)

Date: January 15, 2021

Contents

1 Introduction

It is an immense fact that the robots have become a great interest. Thanks to the their numerous capabilities and advantages in terms of efficiency and reliability, their existence in everyday life is inevitable. Another important aspect of the robots is their ability to be programmed. However, to be able to program them, a certain set of knowledge should be inherited beforehand. For example, to infer the characteristics of the robot, one should analyze the forward and inverse kinematics of a robot. Then, if the intended task is to move the robot from a start position to a goal position, one should also incorporate the motion-planning algorithms. However, to be able fuse these different operations, one should definitely design a control system which both take into account the dynamical properties of the system and the desired/planned motion.

In this project, the main task is to analyze the *PHANToM (Model 1.0) haptic interface* and design a controller which provides a real time trajectory tracking capability to the haptic interface. The real camera image of the PHANToM (Model 1.0) haptic interface can be observed in Figure 1.

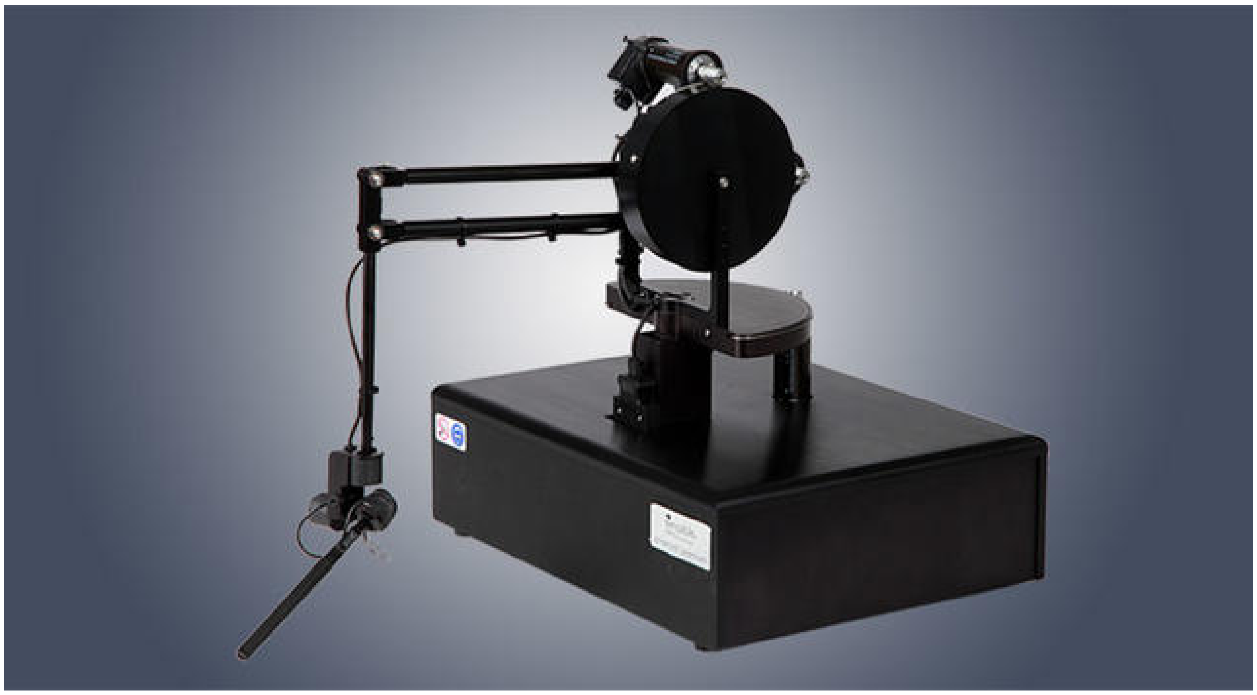

Figure 1: Real camera image of Phantom Model 1.0

2 Model Generation

2.1 Forward Kinematics

It is a good practice to start from constructing a DH table using prefabricated convention. To be able to do this, the first step is to assign the coordinate frames onto the joints. The declared coordinate frames can be investigated in Figure 2.

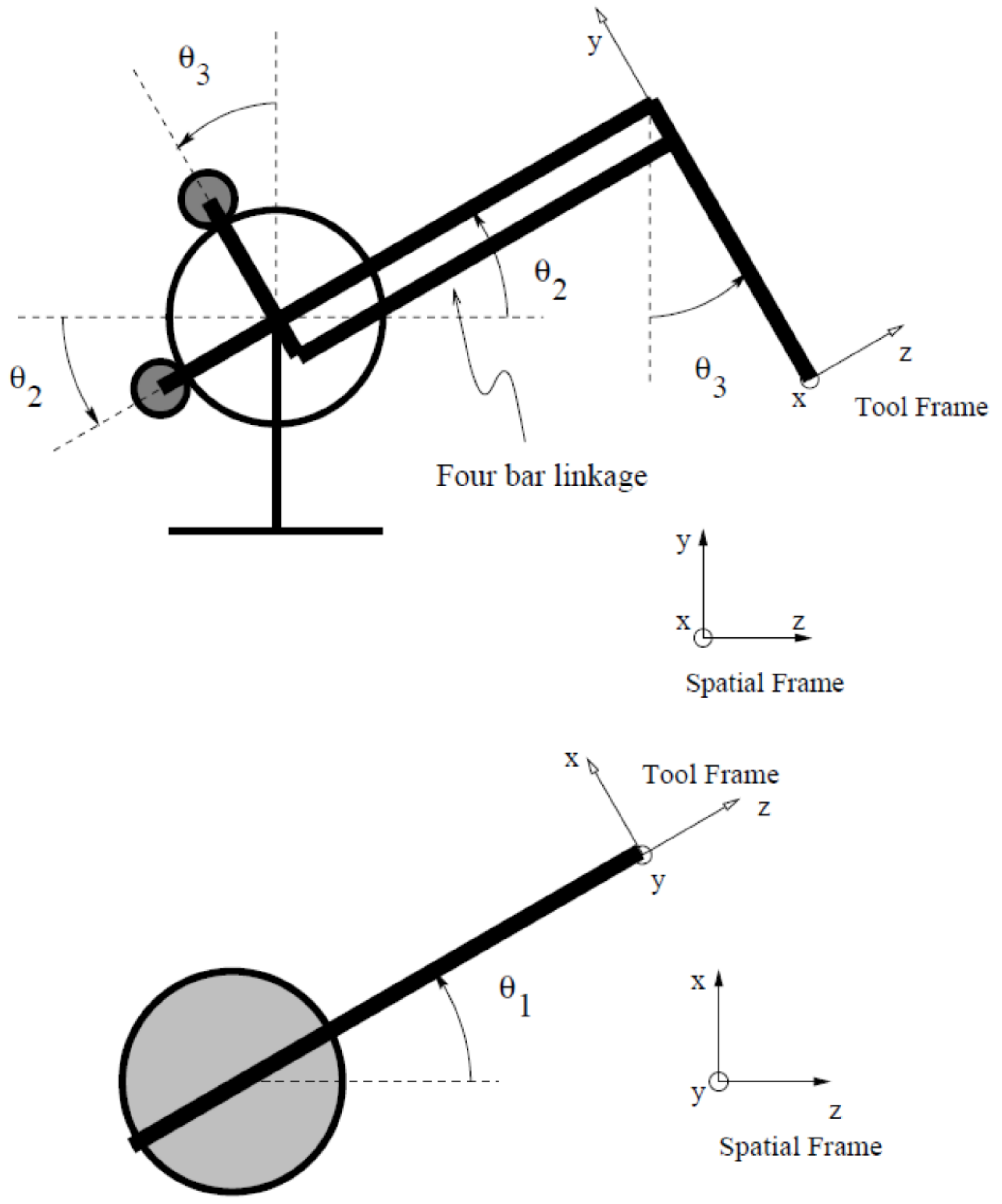

Figure 2: Top and Side View of the Phantom Model 1.0

As can be seen in Figure 2, the movement of the links are not maintained by its own actuators, rather, there is a more coupled behavior. For now, let's assume that the system in the question has the architecture of the one in the Figure 3.

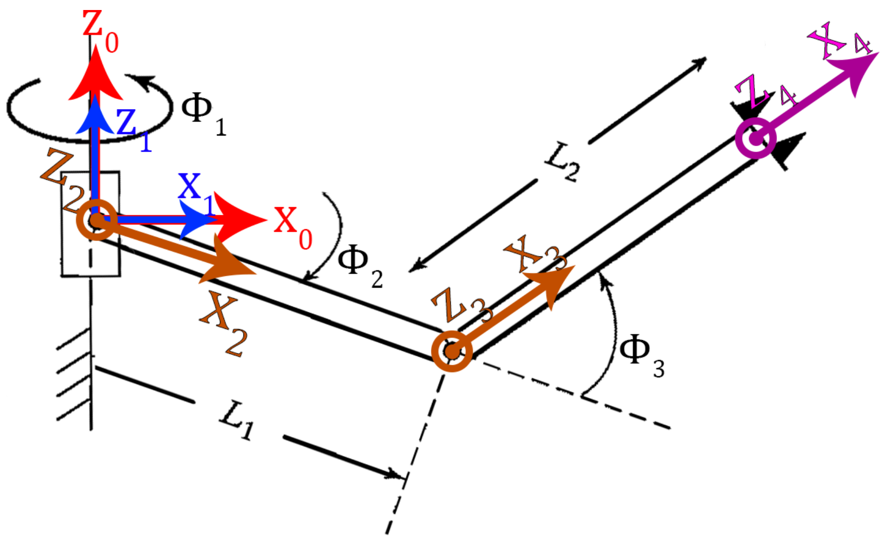

Figure 3: The new constructed scheme of the system

In this new scheme, to begin with, the differences which may occur due to the actuation technique is ignored. These differences will be added to the computation architecture at the end of the derivation. Therefore, the angle set of θ is converted to the new set Φ. From now on, the derivation will be based on Φ, and, the conversion from Φ to θ will be introduced later.

Finally, DH table should be constructed, which can be investigated in Table 1

$\mathbf{i}$	$\alpha_{\mathbf{i-1}}$	$\mathbf{a_{i-1}}$	$\theta_{\mathbf{i}}$	$\mathbf{d_i}$
1	0	0	Φ_1	0
2	$\frac{\pi}{2}$	0	Φ_2	0
3	0	L_1	Φ_3	0
4	0	L_2	0	0

Table 1: DH Table

To be able to define forward kinematic of the given robotic system, firstly, the successive transformation should be introduced. It is known that, the transformation between two successive coordinate frame can be defined as following:

$$ {}^{i-1}_{i}T = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & a_{i-1} \\ \sin\theta_i\cos\alpha_{i-1} & \cos\theta_i\cos\alpha_{i-1} & -\sin\alpha_{i-1} & -\sin\alpha_{i-1}d_i \\ \sin\theta_i\sin\alpha_{i-1} & \cos\theta_i\sin\alpha_{i-1} & \cos\alpha_{i-1} & \cos\alpha_{i-1}d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

In this manner, one can easily produce the subsequent transformation matrices as follows:

$$ {}^{0}_{1}T = \begin{bmatrix} \cos\Phi_1 & -\sin\Phi_1 & 0 & 0 \\ \sin\Phi_1 & \cos\Phi_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{1}_{2}T = \begin{bmatrix} \cos\Phi_2 & -\sin\Phi_2 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ \sin\Phi_2 & \cos\Phi_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{2}_{3}T = \begin{bmatrix} \cos\Phi_3 & -\sin\Phi_3 & 0 & L_1 \\ \sin\Phi_3 & \cos\Phi_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{3}_{4}T = \begin{bmatrix} 1 & 0 & 0 & L_2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

To be able to obtain overall transformation matrix (${}^{0}_{4}T$), the following operation must be executed:

$$ {}^{0}_{4}T = {}^{0}_{1}T\,{}^{1}_{2}T\,{}^{2}_{3}T\,{}^{3}_{4}T $$

Finally, the relation in which the set of joint angles Φ_i and link lengths L_i are connected to the *end-effector* position can be constructed.

$$ \begin{bmatrix} {}^{4}x_{ORG} \\ {}^{4}y_{ORG} \\ {}^{4}z_{ORG} \end{bmatrix} = \begin{pmatrix} \cos(\Phi_1)\ (L_2\ \cos(\Phi_2+\Phi_3) + L_1\ \cos(\Phi_2)) \\ \sin(\Phi_1)\ (L_2\ \cos(\Phi_2+\Phi_3) + L_1\ \cos(\Phi_2)) \\ L_2\ \sin(\Phi_2+\Phi_3) + L_1\ \sin(\Phi_2) \end{pmatrix} $$

Note that, it is not the very final result which should be used in inverse kinematics calculations. Before further proceeding, this form should be converted to the one which aligns with the subject robotic system, Phantom Model 1.0 in this case.

Now, the set of Φ angles should be converted to the θ counterparts. Therefore, it is suitable to write:

$$\Phi_1 = \theta_1$$

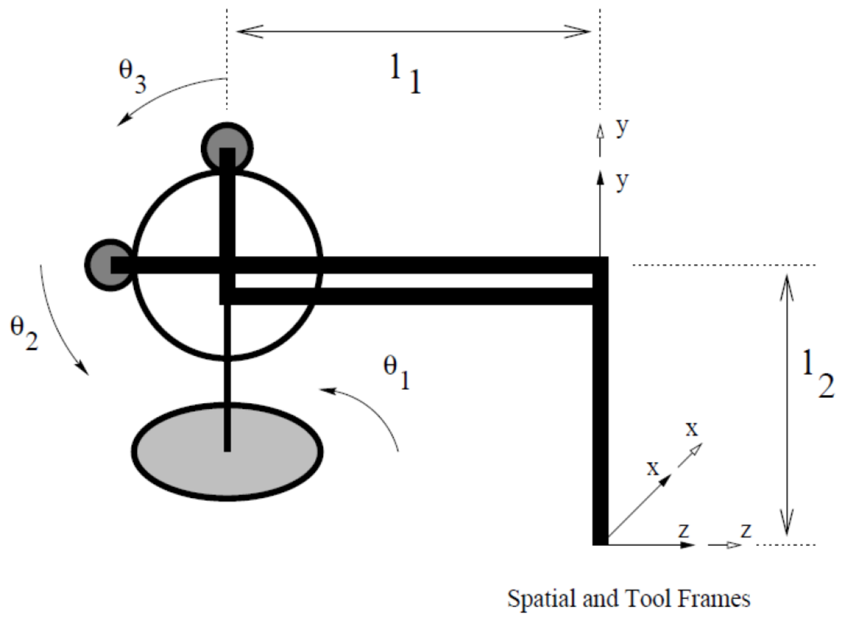

Figure 4: Representative sketch of Phantom Model 1.0

Moreover, from Figure 3 and Figure 2, it is apparent that the θ definition is consistent with the new scheme. Therefore,

$$\Phi_2 = \theta_2$$

Final and the most tricky part is to write Φ_3 in terms of θ. To do this, one should observe the following Figure 5, carefully. Right now, it is appropriate to note that Φ_3 angle **should** define the angle between X_2 and X_3, according to the formal DH definition. Observing, Figure 5, one can easily conclude that:

$$\Phi_3 = \theta_3 - \theta_2 - \frac{\pi}{2}$$

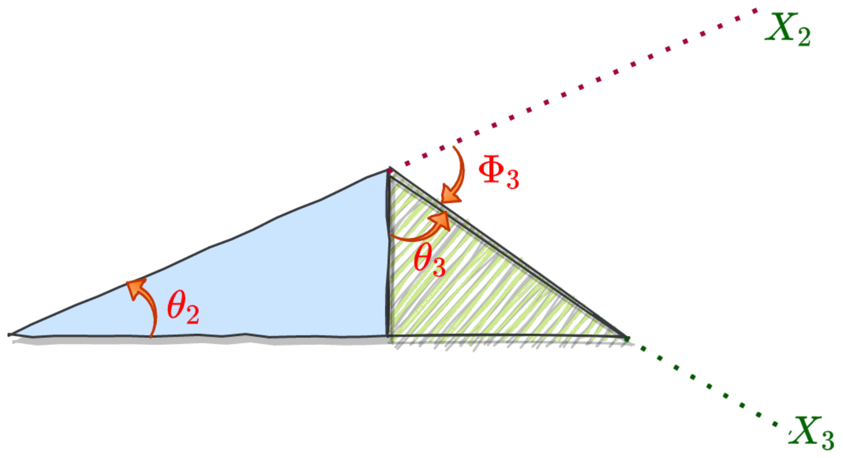

Figure 5: The triangle constituted by angle links and angle definitions

Before jumping into last part, the final process that should be take place is to orient everything with respect to the given reference frame. In Figure 4, the given reference frame can be investigated. By quick inspection, it can be inferred that the base frame can be obtained by following transformations, which is applied through reference frame.

- L_2 translation through Y_{REF} axis.
- $-L_1$ translation through Z_{REF} axis.
- -90^o rotation about Z_{REF} axis.
- -90^o rotation about Y_{REF} axis.

To sum up, the following relation is obtained:

$$ {}^{REF}_{0}T = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & L_2 \\ 1 & 0 & 0 & -L_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Finally, plugging $\Phi - \theta$ conversions into equations and applying final transformation of ${}^{REF}_{0}T$, the following relation is obtained:

$$ {}^{REF}_{4}T = {}^{REF}_{0}T {}^{0}_{1}T {}^{1}_{2}T {}^{2}_{3}T {}^{3}_{4}T $$

$$ \begin{bmatrix} {}^4x_{ORG} \\ {}^4y_{ORG} \\ {}^4z_{ORG} \end{bmatrix} = \begin{bmatrix} {}^4x_{EE} \\ {}^4y_{EE} \\ {}^4z_{EE} \end{bmatrix} = \begin{pmatrix} \sin(\theta_1)\ (L_1\ \cos(\theta_2) + L_2\ \sin(\theta_3)) \\ L_2 - L_2\ \cos(\theta_3) + L_1\ \sin(\theta_2) \\ L_1\ \cos(\theta_1)\ \cos(\theta_2) - L_1 + L_2\ \cos(\theta_1)\ \sin(\theta_3) \end{pmatrix} $$

2.2 Inverse Kinematics

If the position of the end effector is defined according to the coordinate frame shown in Figure 4, the required joint angle values can be found using the inverse kinematics solutions. In contrast to the forward kinematics, there is more than one solution to the problem of finding the desired joint configurations solved by using inverse kinematics algorithm and various approaches are used according to the purpose. For instance, there are several solutions and different methods to solve the inverse kinematics problem for the Phantom Model 1.0 robot that will be examined in the later stages. In generally, there are two approach to solve inverse kinematics problems, namely analytical method (in closed form) and numerical method (in iterative form).

As shown in Figure 7 and Figure 8, the robot's joint angles can be clearly shown in the three-dimensional Cartesian coordinate system. For this reason, the problem of inverse kinematics is solved by geometric approximation. Geometric analysis is made according to the robot's base frame for convenience. Therefore, a transformation must be made between the given coordinate frame and the robot's base frame. The axes are chosen the same to avoid rotations, the rest of the translation process can be calculated with vectoral approach as seen in following figure.

$$P_{ee}^{rb} = -P_{rb}^{o} + P_{ee}^{o}$$

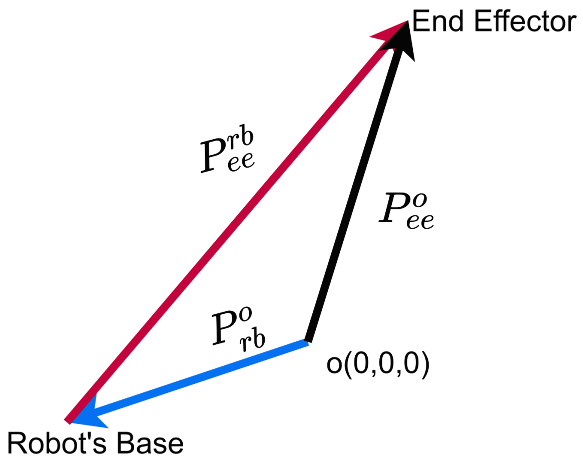

Figure 6: Translations of the frames

As shown in Figure 4, the translation vector between the given coordinate frame and the base frame to be taken into account when performing inverse kinematics calculations is as follows:

$$P_{rb}^{o} = \begin{bmatrix} 0 \\ L_2 \\ -L_1 \end{bmatrix}$$

$$P_{\text{ee}}^{\text{rb}} = \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = - \begin{bmatrix} 0 \\ L_2 \\ -L_1 \end{bmatrix} + \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

As mentioned above, inverse kinematics solutions can have more than one solution. When choosing between these solutions, the environment and the availability of the robot are considered. For instance, if the joint limits of the Phantom Premium 1.0 robot are ignored, the system has a total of 4 different solutions, but it should be noted that the system has only 1 solution due to its mechanical restrictions.

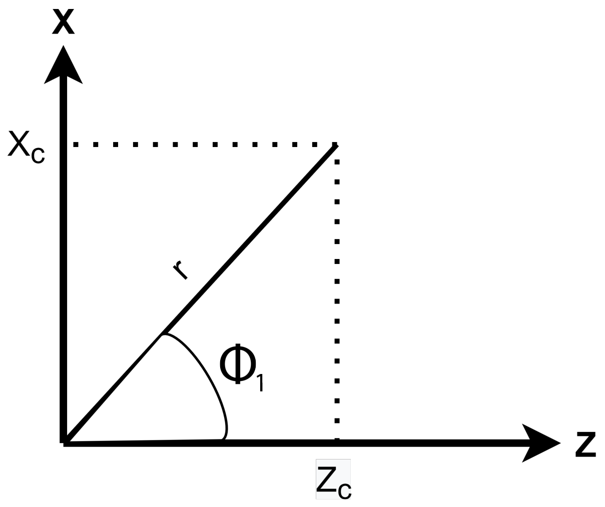

Figure 7: The Top View of the Phantom Model 1.0

The Φ angles introduced in the forward kinematics section will also be used when calculating the inverse kinematics. As shown in Figure 7, one can easily derive the Φ_1 angle like as follows:

$$\Theta_1 = atan2(X_c, Z_c)$$

$$\Theta_1 = -atan2(X_c, Z_c)$$

$$\Theta_1 = \pi - atan2(X_c, Z_c)$$

As seen above, two different results were obtained for theta 1 angle. At the same time, the following equation can be obtained for the variable r

$$r = \sqrt{X_c^2 + Z_c^2}$$

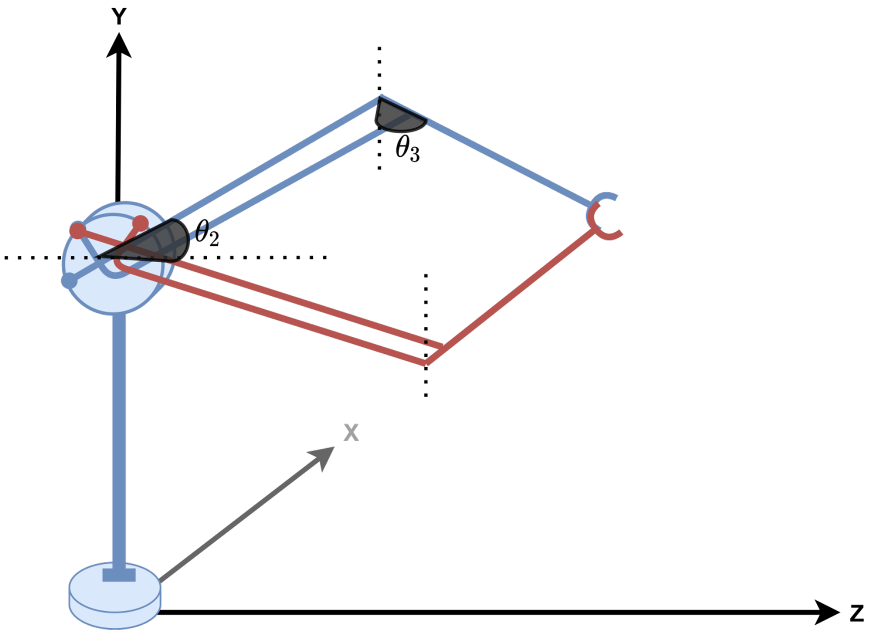

Figure 8: The side view of the Phantom Model 1.0

As seen in Figure 8, Elbow Up and Elbow Down configurations are available for Θ_2 and Θ_3 angles. Solution shown with blue link color represents elbow up, solution shown with red link color represents elbow down. However, the elbow down solution is not available for the Phantom Premium Model 1.0 due to joint limitations.

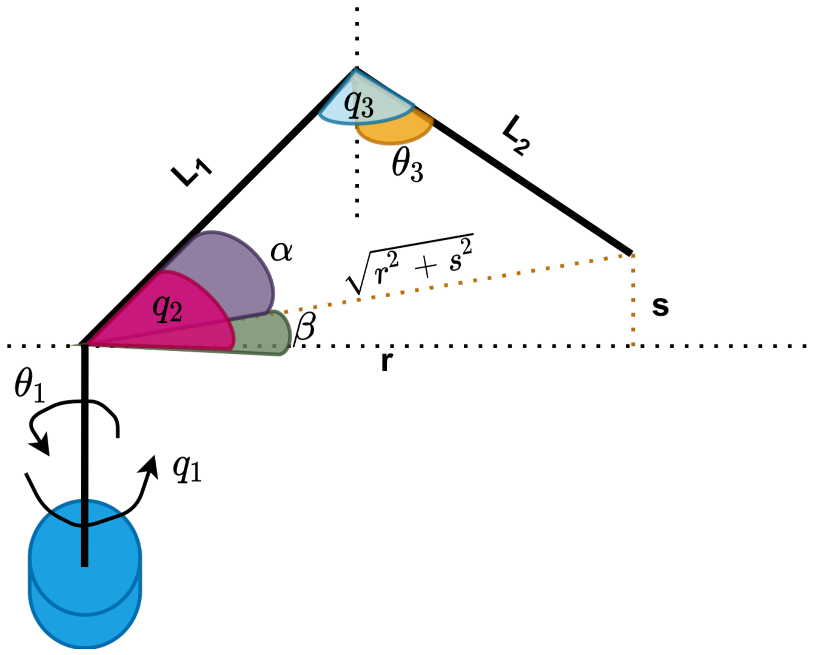

Figure 9: The side view of the Phantom Model 1.0 with auxiliary angle and length

In order to solve the Θ_2 and Θ_3 angles, it is necessary to calculate the auxiliary angles and lengths. Equations of angles and lengths seen in Figure 9 are given below.

$$s = Y_c$$

Θ_2 consists of two angles, α and β. The cosinus theorem is applied to find the α angle as follows:

$$\theta_2 = \alpha + \beta$$

$$L_2^2 = L_1^2 + r^2 + s^2 + 2L_1\sqrt{r^2+s^2}cos(\alpha)$$

$$\alpha = cos^-1(\frac{-L_2^2 + L_1^2 + r^2 + s^2}{2L_1\sqrt{r^2+s^2}})$$

$$\beta = atan2(s, r)$$

$$\theta_2 = cos^-1(\frac{-L_2^2 + L_1^2 + r^2 + s^2}{2L_1\sqrt{r^2+s^2}}) + atan2(s, r)$$

For determining the Θ_3, the cosinus theorem is applied in triangle with sides L_1, L_2, and $\sqrt{r^2+s^2}$.

$$\theta_3 = cos^-1(\frac{L_1^2 + L_2^2 - (r^2+s^2)}{2L_1L_2}) + \theta_2 - \frac{\pi}{2}$$

2.3 Jacobian Matrix Calculation

In the Calculus, The Jacobian is a multidimensional form of the derivate. Also, they are useful for linear transformations. Jacobian Matrix represents the change of end effector's position with respect to small changes in each joint angle. Taking the derivative of forward kinematic gives the Jacobian Matrix. The partial derivative of forward kinematic is as follows:

$$J = \begin{bmatrix} L_1\cos(\theta_1)\ \cos(\theta_2) + L_2\sin(\theta_3)\cos(\theta_1) & -L_1\sin(\theta\ _1)\sin(\theta_2) & L_2\sin(\theta_1)\cos(\theta_3) \\ 0 & L_1\cos(\theta_2) & L_2\sin(\theta_3) \\ -\left(L_1\sin(\theta_1)\cos(\theta_2) + L_2\sin(\theta_1)\sin(\theta_3)\right) & -L_1\ \sin(\theta_2)\ \cos(\theta_1) & L_2\cos(\theta_1)\cos(\theta_3) \end{bmatrix}$$

Jacobians in the force domain yields following equation, and these equality provides a mapping between force and torques. It means that one can easily determines the each joint torques from exerted forces on the end effector of the robot.

$$T = J^T F$$

2.4 Forward Dynamic

Rigid-body equations of motion consists of inertia matrix, Coriolis Centripetal matrix,gravity vector, joint velocity and joint acceleration components. Those components provides a torques of each joint, it can be said that the rigid-body equations of motion maps motion to torque.

$$T = M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q)$$

For this project, dynamic model of PHANToM Model 1.0 haptic interface was given. One can derive the joint positions from the rigid-body equations with given torque data.

$$\ddot{q} = \frac{(T - C(q,\dot{q})\dot{q} - g)}{M(q)}$$

Right now, below equation provides acceleration of the joints. Taking integral of the acceleration results a joint velocity, and taking integral of the acceleration gives a joint positions.

2.5 Trajectory Planner

In this specific project plan, it is asked to design a motion planner that incorporates the *line segments* as main element of the route. To design a trajectory in a 3D, one can make use of the parametric line equation in 3D.

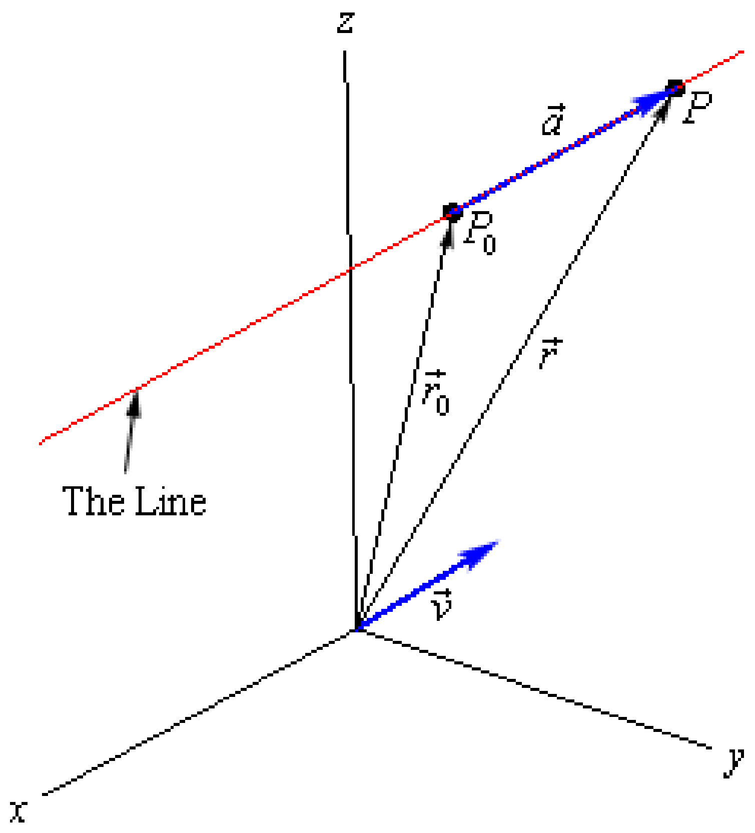

Figure 10: Line representation in 3D

As can be seen Figure 10, to be able to define a line in 3D, one can use an initial point and a vector which is parallel to the subject line. In this way, the parametric equation can be constructed as the following:

$$\mathbf{p} - \mathbf{p_0} = \mathbf{v}t$$

$$\mathbf{p} = \mathbf{p_0} + \mathbf{v}t$$

Actually, a further inspection on the parametric equation easily unveil that the parametric form of the line equation is nothing but the kinematic equation that yields displacement with the usage of the velocity. Therefore, in the project, this approach is used, and the further detail can be found in *Simulink project* file.

2.6 Control System

In the robot control, following block diagram can be designed to the trajectory following control.

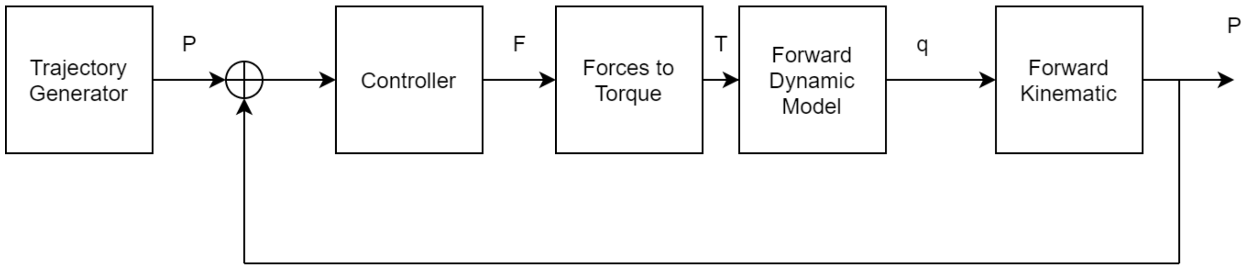

Figure 11: Block Diagram of the Trajectory Following Control

In the 11, details of the Trajectory Generator block mentioned in the Trajectory Planner section, and this block provides a desired positions of the end-effector in the Cartesian coordinate system. Controller takes the error signal and publishes meaningful controller signal. For this control problem, proportional integral derivative controller was selected. Thanks to the Jacobian Matrix, force signal can easily mapping to the torque signal. With making some manipulation of the dynamical model of the robot, torque signal can be converted to joint positions (aka configurations), and finally Forward Kinematic Block provides a feedback to the controller by mapping the joint positions to Cartesian positions.

2.6.1 Proportional Integral Derivative (PID) Controller

PID Controller is the most common controller in the industrial applications. The most important reasons for this are that it is easy to apply and its performance results in success in most plants. PID Controller consists of Kp, Ki, and Kd parameters. Each of them has different meaning in terms of the controlling the plant. The proportional term allows the error to affect the controller signal directly. The integral term transfers the integral of the error, that is, the sum of the errors up to that point, to the controller signal. The derivative term transfers the instantaneous rate of change of the error to the control signal. As a result, the PID can be shown with the following block diagram and transfer function.

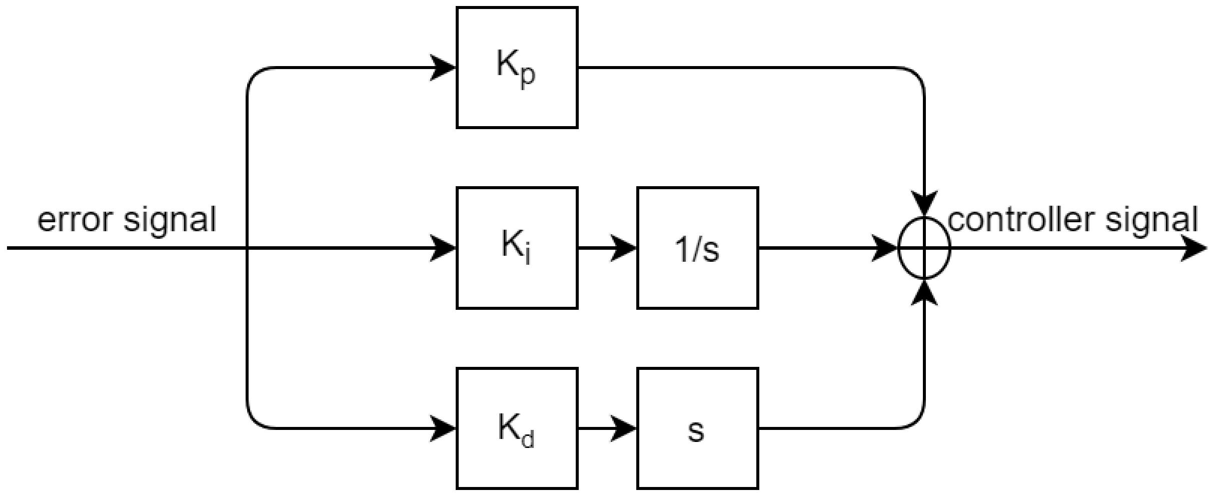

Figure 12: Block Diagram of PID Controller

$$u(t) = K_p\, e(t) + \frac{K_i}{s} e(t) + K_d\, s\, e(t)$$

In the implementation of the PID Controller on Trajectory Following Control scheme, controller parameters were tuned by manually. As a result, following parameters were found:

$$\begin{array}{c|ccc} PID_x & K_p = 100 & K_i = 25 & K_d = 6.85 \\ PID_y & K_p = 200 & K_i = 100 & K_d = 4 \\ PID_z & K_p = 31 & K_i = 14.5 & K_d = 1.85 \end{array}$$

3 Results

$$Task\ Completion\ Time = 2.1sec$$

$$Average\ Tracking\ Error = \begin{bmatrix} -3.3548 \times 10^{-4} \\ -2.8519 \times 10^{-4} \\ -0.0020 \end{bmatrix} m$$

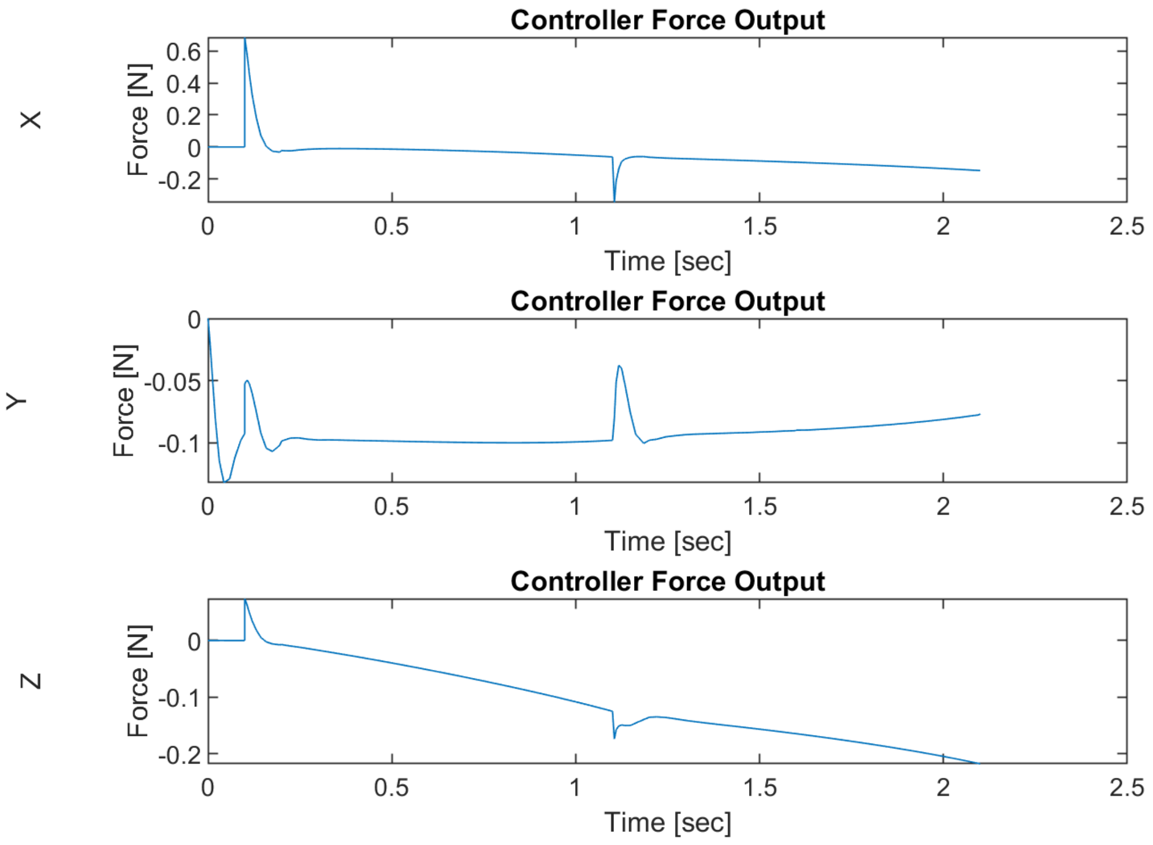

Figure 13: Force Output of the Controller

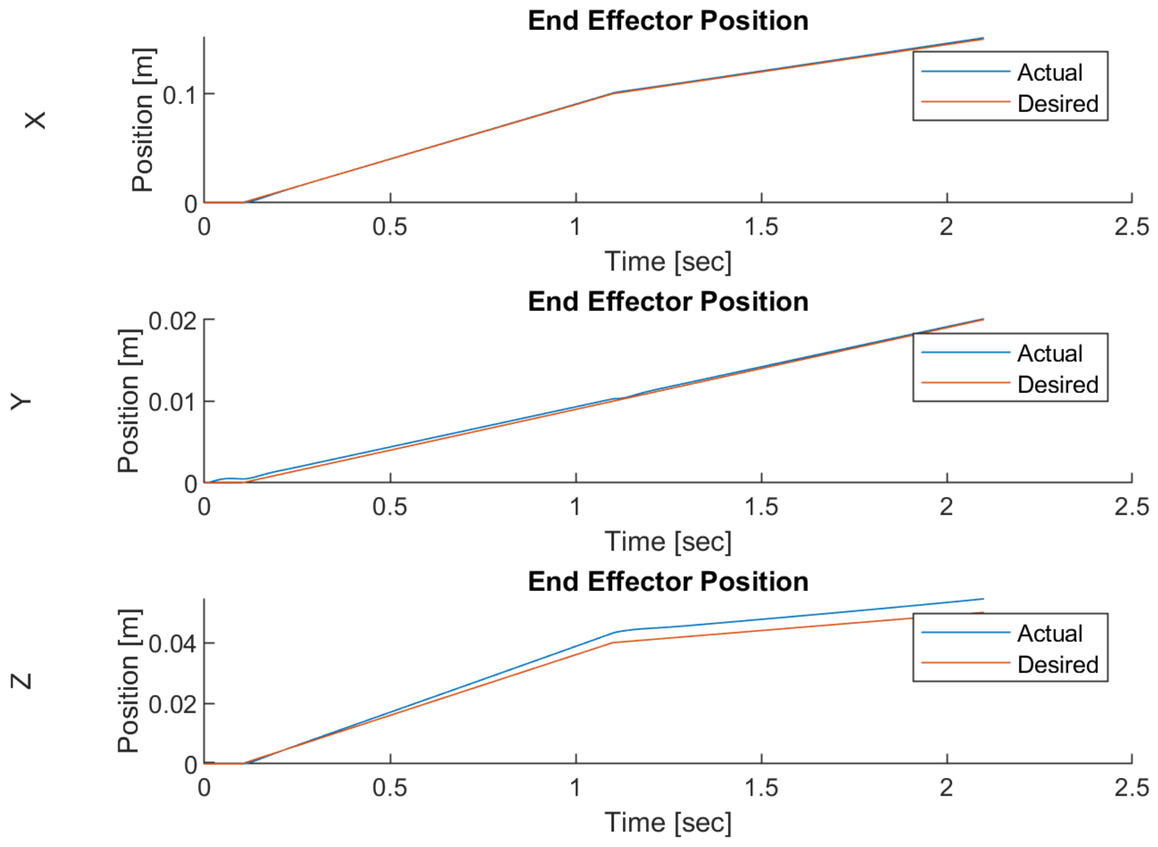

Figure 14: End Effector Position

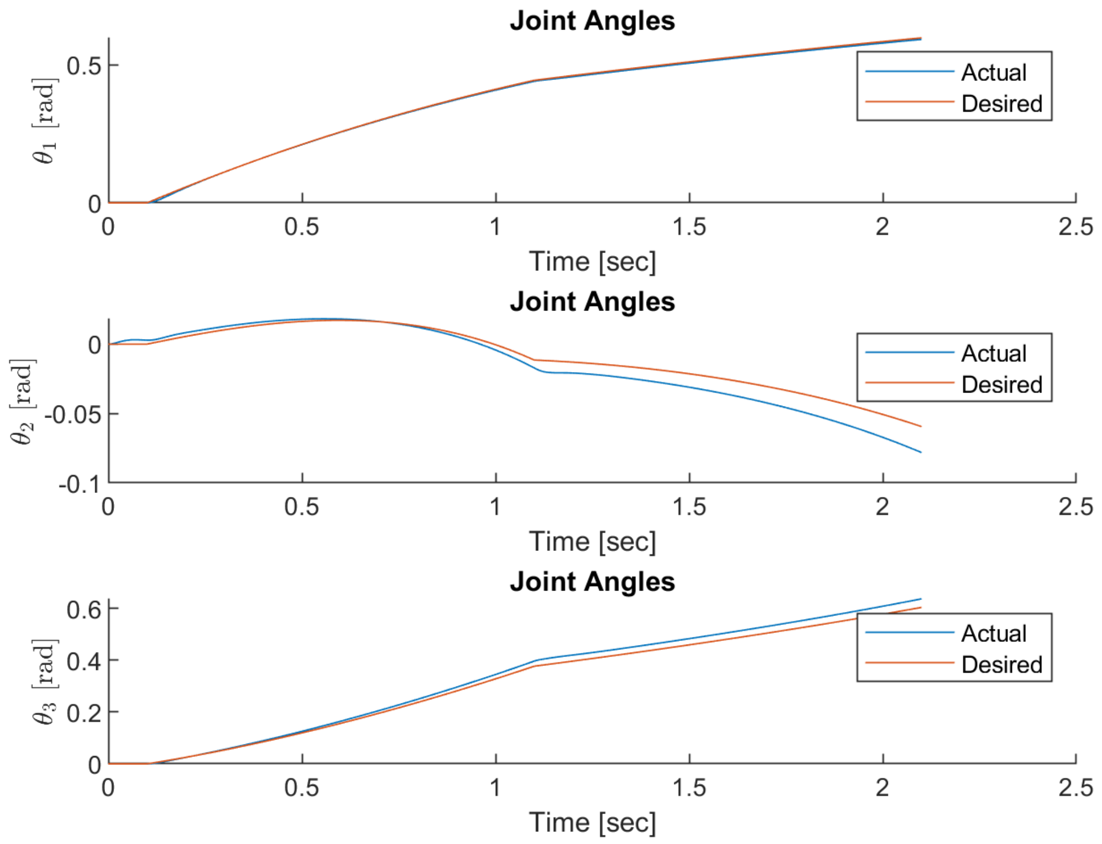

Figure 15: Joint Angles

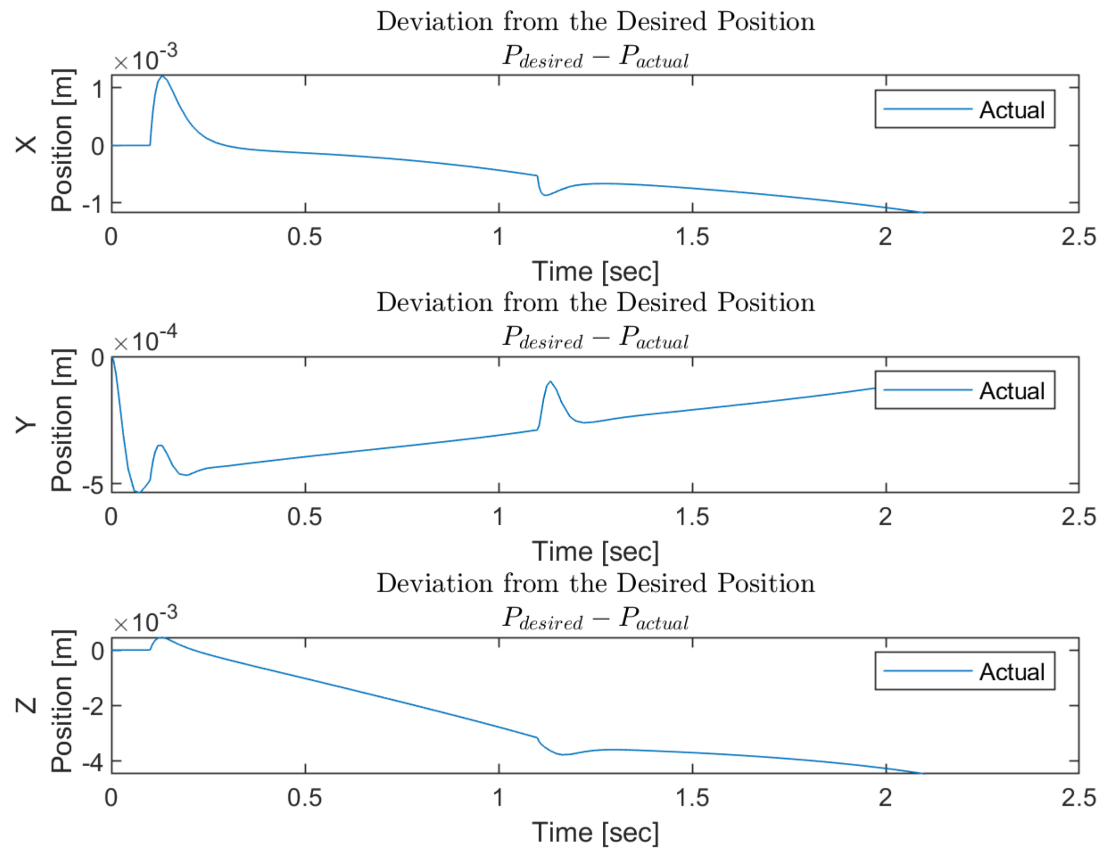

Figure 16: Difference Between Desired and Actual End Effector Position

4 Conclusion

All in all, in this project, the dynamical analysis of the *PHANToM (Model 1.0) haptic interface* was investigated, then, based on the obtained dynamical properties, the suitable controller design was implemented. Firstly, the forward and inverse kinematics of the *PHANToM (Model 1.0) haptic interface* was obtained. Then, based on the inherent properties of the robot, the dynamical model was extracted. The trajectory planning was constructed based on the line segments via using the knowledge of the analytical representation of the line in 3D, and the adaptation of this to the kinematic equation. Finally, the whole system was being controlled via using the *PID controller*. The PID tuning operation was executed based on the experience and *trial-and-error*.