Modelling and Controlling of a 3-DOF Linear Parallel Delta Robot

Bachelor Thesis

GERMAN UNIVERSITY IN CAIRO

FACULTY OF ENGINEERING AND MATERIAL SCIENCE

MECHATRONICS ENGINEERING

Author:

Mohammed Ashraf

Supervisor:

Dr. Ayman El-Badawy

This is to certify that:

(i) The thesis comprises only my original work toward the Bachelor Degree of Science (B.Sc.) at the German University in Cairo (GUC),

(ii) Due acknowledgement has been made in the text to all other material used

_____

Mohammed Ashraf

12/06/2022

Abstract

This paper describes a linear parallel delta robot, which is one of the most common parallel manipulator models. The main component of the machine is a three-DoF parallel robot with stepper motors that controls the position of the moving platform. Because of the parallel robot's complex structure and connected motion, a dynamics study of the parallel robot with three translational DoF (linear Delta robot) is performed in order to aid its development. The parallel robot's position, velocity, and acceleration are investigated. A simplified dynamic model of the parallel robot based on the notion of virtual work is constructed by introducing a mass distribution component. The simulation results are then compared to an ADAMS model to confirm their accuracy. Several control methods are introduced and results are compared to determine the better control method used on the robot. At last, a section with future recommendations to be implemented and tested are mentioned at the end of the paper.

Contents

1 List of Abbreviations **5**

2 Introduction **10**

3 Literature Review **11**

4 Robot Manipulators **14**

4.1 History 19

4.2 Serial Manipulators 20

4.3 Parallel Manipulators 23

5 3D Model **24**

6 Kinematic Analysis **27**

6.1 Frames of Reference 28

6.2 Inverse Kinematics 31

6.3 Position Analysis 32

6.4 Velocity Analysis 33

6.5 Acceleration Analysis 34

6.6 Verification 36

6.7 Open-Loop Kinematics Results 38

7 Dynamics **40**

8 Simulation results and validation **43**

9 PD Controller **44**

10 PID **50**

11 Computed Torque Control **56**

12 Comparison **63**

13 Conclusion **69**

14 Future Works **70**

1 List of Abbreviations

DoF	Degree of Freedom
PD	Proportional Derivative
PID	Proportional Integral Derivative
CTC	Computed Torque Control
RMSE	Root Mean Square Error
PD+	Proportional Derivative with gravity compensation
MPC	Model Predictive Control

List of Figures

1 Serial Robot Manipulator 15

2 Relationship between input torques and joint motion 15

3 Inverse Kinematics 15

4 Forward Kinematics 16

5 Relationship between input torques and moving platform motion 16

6 Joint Space Control 18

7 Cartesian Space Control 18

8 Serial Manipulator 21

9 Serial Robot Manipulator 21

10 Serial Robot Manipulator used for Welding 22

11 Serial Robot Manipulator for Assembly 22

12 Hexapod Parallel Robot 23

13 Top View 25

14 Side View 26

15 Isometric View 26

16 Fixed Base 28

17 Moving Platform 29

18 Kinematic Diagram of 3-DoF Delta Robot 29

19 Loop Closure Equation 32

20 Simulink Model 38

21 Kinematics Helix Trajectory 39

22 Control block diagram for the virtual prototype 43

23 PD Feedback Loop 44

24 PD Helix Trajectory 45
25 Top View 46
26 Side View 46
27 PD MATLAB X-Trajectory 47
28 PD MATLAB Y-Trajectory 47
29 PD ADAMS X-Trajectory 48
30 PD ADAMS Y-Trajectory 48
31 PD MATLAB VS ADAMS X-Trajectory 49
32 PD MATLAB VS ADAMS Y-Trajectory 49
33 PID Feedback Loop 50
34 PID Helix Trajectory 51
35 Side View 52
36 PID MATLAB X-Trajectory 52
37 PID MATLAB Y-Trajectory 53
38 PID ADAMS X-Trajectory 53
39 PID ADAMS Y-Trajectory 54
40 PID MATLAB VS ADAMS X-Trajectory 54
41 PID MATLAB VS ADAMS Y-Trajectory 55
42 Computed Torque Control 58
43 CTC Helix Trajectory 59
44 CTC MATLAB X-Trajectory 59
45 CTC MATLAB Y-Trajectory 60
46 CTC ADAMS X-Trajectory 60
47 CTC ADAMS Y-Trajectory 61

48 CTC MATLAB VS ADAMS X-Trajectory 61
49 CTC MATLAB VS ADAMS Y-Trajectory 62
50 PD Position Error 63
51 PD Velocity Error 64
52 PID Position Error 64
53 PID Velocity Error 65
54 CTC Position Error 65
55 CTC Velocity Error 66
56 X Trajectory Error 66
57 Y Trajectory Error 67
58 Z Trajectory Error 67

List of Tables

1 Prismatic-Input Delta Robot Parameters 30

2 Measured Parameters 36

3 Forward Kinematics Test 37

4 Inverse Kinematics Test 37

5 Controller errors in m 68

2 Introduction

Parallel robots have been employed in a variety of industries, in production lines, product assembly, surgery, hybrid polishing, 3D printing, aerospace industry and more. The advantage of parallel robots over serial robots is that they can tackle more complex jobs than serial robots, but they are far more complex in return. Furthermore, one of the key advantages of parallel robots is the ability to keep the motors fixed in the base, allowing for a significant reduction in the active movable mass of the robot structure. When using direct drive, it is necessary to keep the motors on the robot base; consequently, parallel robots are best suited to direct-drive actuation. They also have a high degree of stiffness. These features make manipulations more precise and faster. However, such advantages do not come cheap, and they are restricted. However, they frequently have a restricted work space, which is defined as the intersection of the individual work spaces of each serial arm that makes up the workspace. Clavel's discovery of the DELTA robot in 1988, a type of new parallel robot primarily suited to handle light things, helped to overcome this constraint. Codourey created a direct-drive version of the DELTA robot in 1991 to maximise its capabilities. Direct-drive robots have the following advantages: simple mechanics, no backlash, less friction and noise, and increased stiffness (Asada and Yousef-Toumi, 1987). Due to mechanical coupling and inertial variations that are directly sensed on each motor axis, this approach demands more sophistication at the controller level [1]. Parallel robots have a smaller work space volume than serial robots, have less dexterity, and have more single configurations. Furthermore, the presence of singularities reduces the work space available to parallel manipulators. As a result, parallel manipulators' potential benefits cannot be completely exploited [2]. In order to solve the shortcomings of manipulators, L. Wang, J. Wu, and J. Wang suggested redundant parallel manipulators in [3]. Because of their simple kinematics and uncoupled motions, three-Degree of Freedom (DoF) linear parallel manipulators have been employed to address these flaws.

3 Literature Review

Over the last few decades, the development of robot-dynamic models has piqued researchers' curiosity. The main challenge is coming up with a solution that is sufficiently reflective of the real system while still being simple to calculate in real time for use in the control algorithm. The challenge is significantly more complicated for parallel structures than it is for serial robotics, owing to the analytical complexity posed by joint-variable interactions. Although there has been research into approaches that can be used to model such robots in general, the results do not lend themselves to real-time processing. Cutting out the closed-chain mechanism at passive joints and first considering the dynamics of the tree robot thus generated is a method that has been investigated extensively.

For the kinematic modelling and analysis, Stock.M and Miller.K published an optimal kinematic design of spatial parallel manipulators with different configurations of a linear delta robot [4]. K.Youssef and A.El-Badawy published the kinematic and dynamic modelling and simulation of a six-DoF Stewart Platform using multibody dynamics approach [5]. Mehrafrooz.B, Mohammadi.M and Masouleh.M published a paper on the kinematic sensitivity of the revolute and prismatic joint three-DoF delta robots [6]. Wang.L and Hsieh.J published a paper on the extreme reaches and reachable workspace analysis of general parallel robot [7].

The closure criterion is then loosened by applying the d'Alembert virtual work principle to the mean of the manipulator's Jacobian matrix (Kokkinis and Stoughton 1991; Nakamura 1991; Wang and Chen 1994), or by using Lagrange multipliers (Kleinfinger 1986). The direct application of the Newton-Euler method, as well as methods based on the virtual work principle (Clavel 1991; Zhang and Song 1993; Codourey and Burdet 1997), Lagrange formalism (Fujimoto et al. 1991; Miller and Clavel 1992; Lebret, Liu, and Lewis 1993; Pang and Shahinpoor 1994), Hamilton's

equations (Miller 1993) or other particular methods (Devaquet 1992, Zanganeh 1997). Almost everyone agrees that a comprehensive model that accounts for the masses and inertia of all the linkages leads to extremely complex solutions that are inefficient for control. Some authors have proposed parallel algorithms (Guglielmetti 1994, Gosselin 1996) that take advantage of the unique geometry of parallel structures to speed up computing. However, as Ji (1993) shown, the mass and inertia of the legs might be overlooked when considering the platform and motor dynamics. This hypothesis simplifies dynamics, allowing them to be utilized in a real-time model-based control technique.

This study focuses on Delta, a three-DoF linear parallel robot that is currently the fastest manipulator available. An offer of a closed form formulation of the Delta's entire inverse dynamics model, which allows for both analysis and implementation of an appropriate control system. The following sections will discuss and derive the equations of motions of the parallel robot by applying the d'Alembert virtual work principle for the dynamics. Inverse kinematics can be utilized to develop a system that is both efficient and effective. When using the parallel manipulator with rapid acceleration, high speed operation, and/or high load application, the rigid-body dynamic characteristic should be explored [8].

For control methods of general parallel manipulators, R. Kelly, B. Santibanez and A. Loria published a book regarding several techniques for parallel manipulators ranging from a simple linear control theories to complex non-linear control theories with Lyapunov verifications [9]. W. Khalil and O. Ibrahim [10] and P. Yen and C. Lai [11]. Y. Li and Q. Xu proposed a robust Computed Torque Control and Passivity-based robust control scheme [12]. Wu.M, Mei.J, Ni.J and Hu.W [13] developed PD, PD+ and CTC controllers with the implementation of disturbance observer to account for the disturbances injected into the system. Li.I, Chiang.H and Lee.L developed a three-DoF Linear delta

robot with an LDR controller implemented [14].

4 Robot Manipulators

As shown in Fig.1, a robotic manipulator is made up of rigid links joined by joints. The manipulator's one end is connected to a fixed base, and the other end is connected to an end effector or a tool. Each manipulator joint is linked to an actuator, such as a motor. The actuators apply torques or forces to the joints, which causes them to move. As shown in Fig.2, the term dynamics explains the relationship between joint motion and input forces and torques in robotics. The movements of the joints cause the links to move, allowing the end effector to accomplish a task. The moving platform's motion is determined by the manipulator's joint motion and mechanical structure, and robot kinematics specifies the link between them. An inverse kinematics model is used to calculate the joint motion given the motion of the moving platform, while a forward kinematics model is used to determine the motion of the moving platform given the joint motion. Figure 3 and 4 shows block diagrams illustrating the ideas of forward and inverse kinematics. As a result of the robot dynamics and kinematics, the motion of the robot's moving platform is related to the input torques, as shown in Fig. 5. A desired motion of the end effector must first be planned or specified for the robot manipulator to complete a task. The task space refers to the coordinates in which the manipulator task is defined, whereas the joint space refers to the vector space in which the joint displacements are defined. A robot controller is designed to deliver appropriate input torques to the robot joints in order for the moving platform's output motion to follow the desired or specified motion in task space.

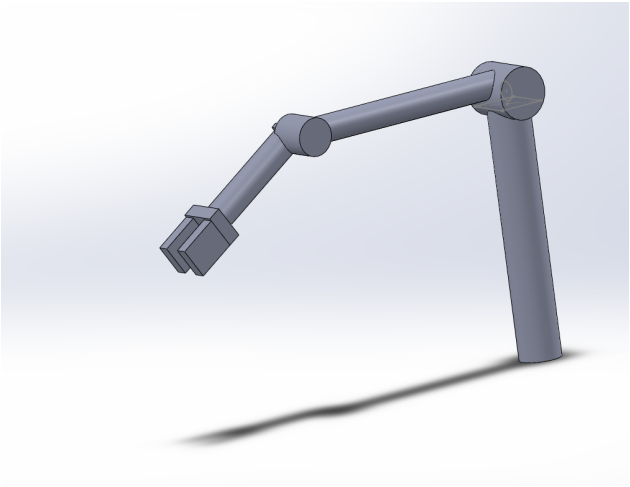

Fig. 1: Serial Robot Manipulator

Fig. 2: Relationship between input torques and joint motion

Fig. 3: Inverse Kinematics

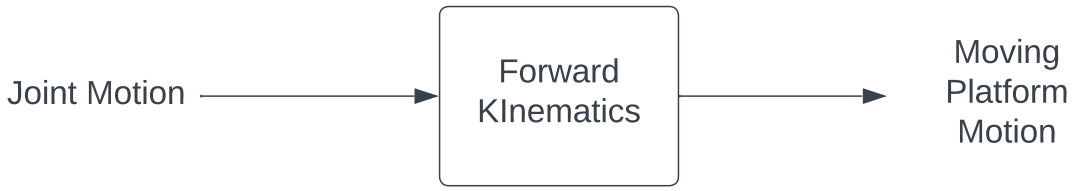

Fig. 4: Forward Kinematics

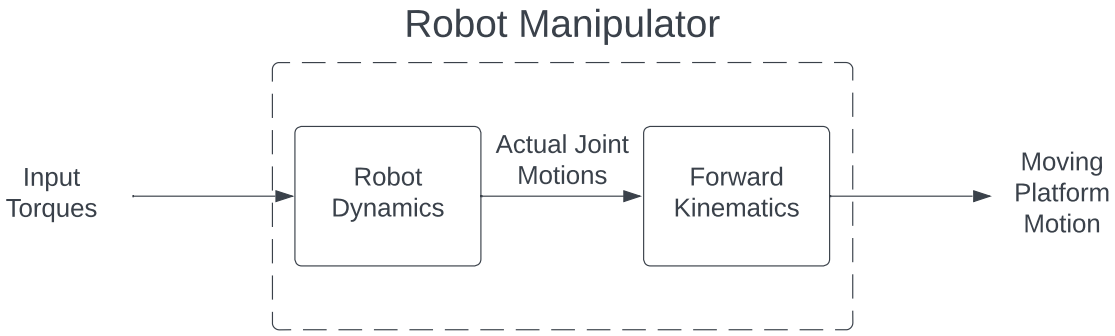

Fig. 5: Relationship between input torques and moving platform motion

The inverse kinematics model can be used to compute the equivalent desired motion in joint space given a desired motion of a moving platform in task space, such as Cartesian space. The desired joint motion in joint space is set as a reference input to the controller and compared to the actual joint motion to create a feedback error in joint space in the joint-space control method. Figure 6 depicts a block diagram of a typical joint-space control approach. The desired motion of the moving platform in Cartesian space is employed directly as the reference input of the control system in the Cartesian-space control method, as shown in Fig. 7. The forward kinematics equation can be used to calculate the motion of the moving platform, which can then be compared to the reference input to generate a feedback error in Cartesian space. Because an exact kinematics model is necessary for Cartesian-space control, this method is vulnerable to kinematics modelling errors [15].

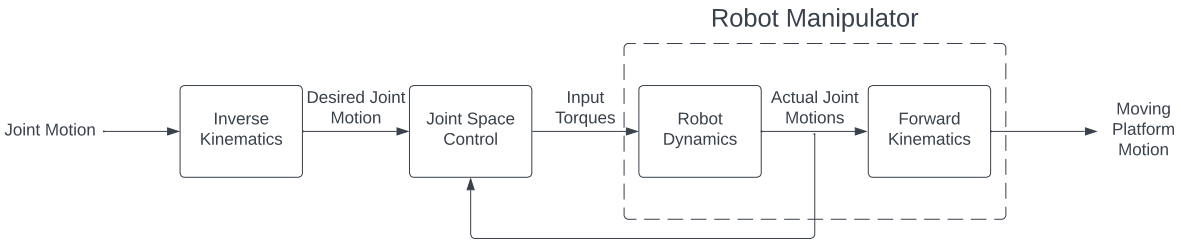

Fig. 6: Joint Space Control

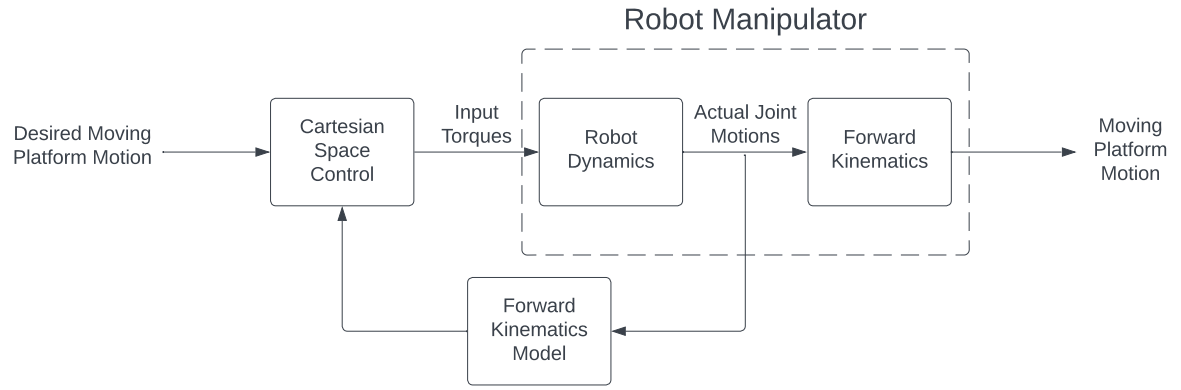

Fig. 7: Cartesian Space Control

4.1 History

In the previous two decades, robotics has seen a great increase in activity, both in terms of research and in terms of capturing the public's imagination with its seemingly limitless and diversified possibilities. Robots have evolved technologically during this time, from simple pick-and-place, painting, and welding robots to more sophisticated assembly robots for placing integrated circuit chips into printed circuit boards, and mobile carts for parts handling and delivery. Several aspects of robotic automation have now become standard on the manufacturing floor, and the field is poised for a fresh explosion in areas such as hazardous settings, minimally invasive surgery, and micro electro-mechanical systems. After WWII, work on remotely controlled mechanical manipulators for handling radioactive material began at Argonne and Oak Ridge National Laboratories, which led to the development of today's robots. The master-slave mechanism consisted of a master manipulator that was steered by the user through a sequence of moves before being reproduced by the slave unit. A set of mechanical links connected the slave unit to the master. These linkages were eventually replaced by either electric or hydraulic powered coupling in "teleoperators" as these machines are called, made by General Electric and General Mills. Force feedback to keep the slave manipulator from crushing glass containers was also added to the teleoperators in 1949. Computer Numerically Controlled (CNC) machine tools for precision machining of low-volume, high-performance airplane parts were developed along with the introduction of teleoperators. The first robots, invented by George Devol in 1954, replaced the teleoperator's master manipulator with the CNC machine tool controller's programmability. In the 1980s, many efforts were made to improve the performance of industrial robots in fine manipulation tasks: active methods using feedback control to improve positioning accuracy and program compliance, and passive methods involving a mechanical redesign of the arm. The majority of the research focused on deciphering algorithms for robot control, trajectory planning, and sensor aggregation. Efficient recursive Lagrangian and computational

strategies for determining the gravity and Coriolis force terms in robot dynamics were among the first active control methods devised. Parallel to this, a project was underway to precisely linearize the dynamics of a multi-joint robot utilizing state feedback and a technique known as computed torque. While this method was computationally intensive, it offered the advantage of providing accurate constraints on the manipulator's transient performance. It required accurate cancellation, which had to be done either roughly or adaptive in practice. Other research, known as hybrid or compliant control, focused on establishing position/force control systems for robots in touch with the environment. Robot links became lighter and had harmonic drives instead of gear trains in their joints as the search for more precisely controllable robot manipulators progressed [16].

4.2 Serial Manipulators

The most typical industrial robots are serial manipulators, which consist of a number of linkages connected by motor-actuated joints that stretch from a base to an end-effector. A serial robot, in its most basic form, is made up of a series of rigid links connected by joints. Robots with solely revolute or prismatic joints with orthogonal, parallel, and/or intersecting joint axes have been developed for manufacturing and control reasons. This type of structure is known as an open chained mechanism. The figures below show different applications for this type of robot manipulators.

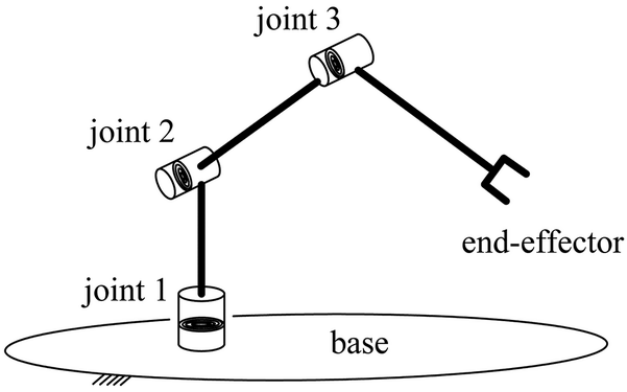

Fig. 8: Serial Manipulator

Fig. 9: Serial Robot Manipulator

Fig. 10: Serial Robot Manipulator used for Welding

Fig. 11: Serial Robot Manipulator for Assembly

4.3 Parallel Manipulators

Parallel manipulators are made up of several kinematic chains that are coupled in a parallel fashion. To move a shared point, the kinematic chains act together. This common point, which is also known as the end effector, usually comprises of a manipulator that performs a specific duty. A parallel manipulator is considered a closed chained mechanism since the kinematic chains eventually connect to a single point. Parallel manipulators often have actuators at or near the system's base, as opposed to serial manipulators, which have actuators at each joint. The dynamic model is highly complex in nature, and there are numerous instances of singularities that must be mapped out and avoided in order to retain system control, which is one of the disadvantages of this sort of design. From the Stewart platform or Hexapod Parallel Robot depicted in Fig (), which is used in aircraft motion simulators, to the Delta robot, which is utilized in packaging facilities, parallel robots come in a broad variety of designs and applications. This explains why no definitive answer can be given as to which controller best suits the functioning of all parallel robots.

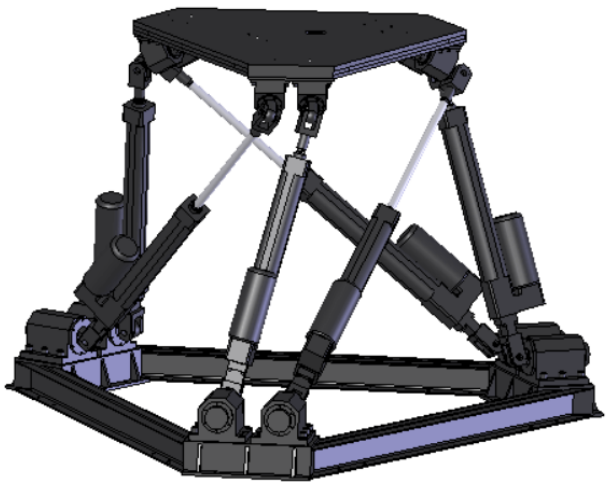

Fig. 12: Hexapod Parallel Robot

5 3D Model

The linear Delta robot is composed mainly of a fixed base, three kinematics chains and a moving platform. The moving platform is supported by three identical serial chains. Each chain represents a four-bar mechanism and each chain connects the moving platform to the base by a prismatic joint, a revolute joint, and a spherical joint in sequence. A prismatic joint connects the parallelogram mechanism and the base. Thus, the moving platform is attached to the base by three identical PRS linkages. Through a combination of the actuation to the three prismatic joints, the output of the moving platform can be obtained. A fixed Cartesian frame Ox,y,z is assigned at the centered point O of the base frame, and a moving frame Px,y,z on the triangle moving platform at the centered point P, along with the y- and v-axes perpendicular to the platform, and the x- and z-axes parallel to the u- and w-axes, respectively. Both $A_1A_2A_3$ and $B_1B_2B_3$ are assigned to be equilateral triangles so as to obtain a symmetric work space of the manipulator. The delta robot dimensions were measured and the robot was drawn on SOLIDWORKS. The model helps us simulate the motion of the robot further down the line in both SOLIDWORKS and ADAMS dynamics simulation Figs. (14), (13) and (15) show different views of the final 3D model.

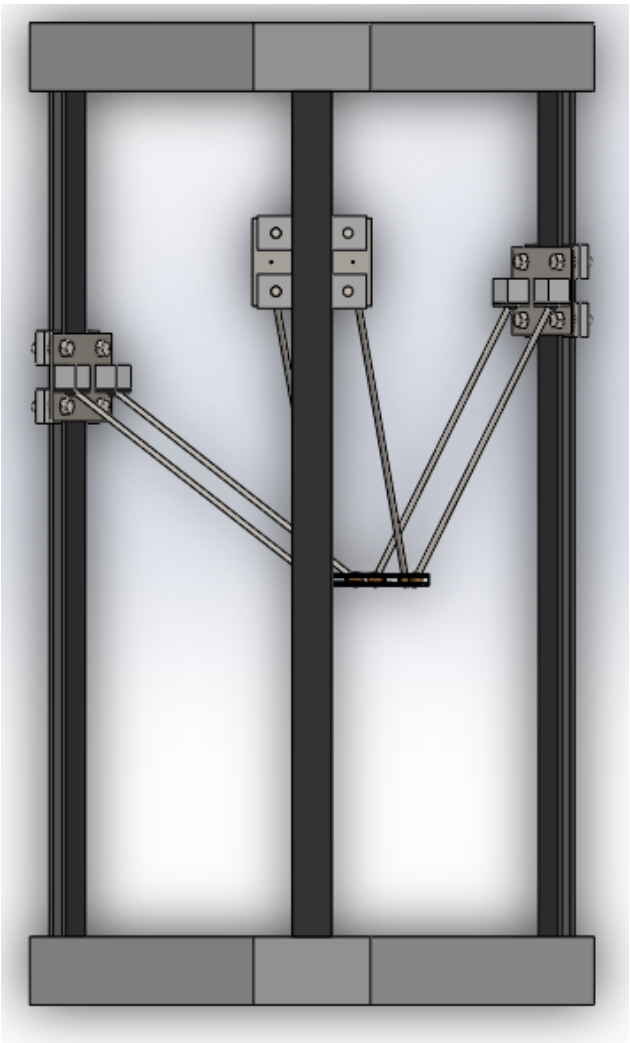

Fig. 13: Top View

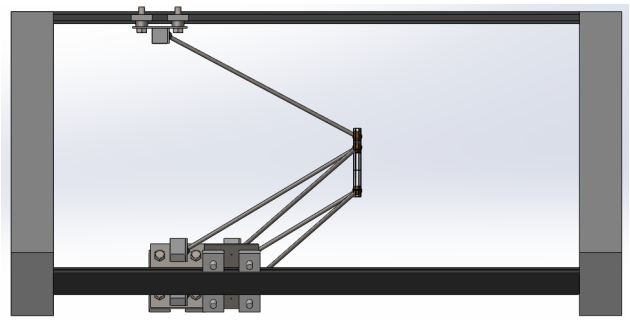

Fig. 14: Side View

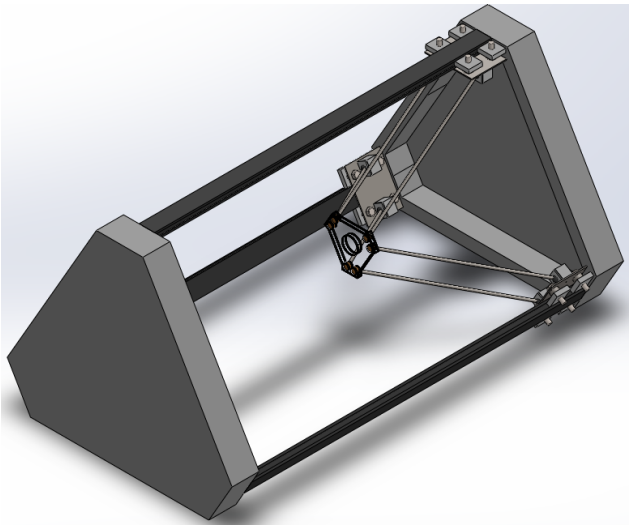

Fig. 15: Isometric View

6 Kinematic Analysis

Mobility

Mobility, also known as External Degrees-of-Freedom, is the number of inputs, in or case the torques, that should be applied to the relevant joints in order to manipulate the motion of the corresponding joints (the prismatic joints).

The ends of all arms are connected directly to either the travelling plate or the corresponding actuator by universal joints. Grübler's formula states:

$$M = 6(n-j-1)\sum_{i=1}^{j} f_{\mathrm{i}} \tag{1}$$

where n is the number of links of the manipulator including the base, j is the number of joints, and joint i has a connectivity (degree of motion) of f_{i} (universal joints = 2, prismatic joints = 1).

The characteristics of the manipulator are:

A total of 12 universal joints (2 for each leg).

A total of 3 prismatic joints.

A total of 14 links (including the base).

Applying Grübler's formula:

$$M = 6(14-15-1)+2(12)+1(3) = 15 \tag{2}$$

which is not correct. An assumption will be taken that each pair of legs will be treated as one. Taking this into consideration, the number of links can be reduced to 11 and the number of joints can be reduced to a total of 12. Applying Grübler's formula:

$$M = 6(11-12-1)+2(6)+1(3) = 3 \tag{3}$$

6.1 Frames of Reference

In order to facilitate the analysis of our robot, we will establish two frames of reference: first is a global frame, *O-XYZ*, which is located in the centre of the base of the robot. The Z-axis is perpendicular to both the base and the XY plane and points towards the end effector platform. The second frame is a local frame titled *P*, located at the centre of the platform. The Z-axis is perpendicular to the platform and the XY plane and points to the second triangular base. Fig 18 shows a schematic of the 3-DoF delta robot showing the aforementioned coordinate systems along with some important points to aid with the analysis of the system.

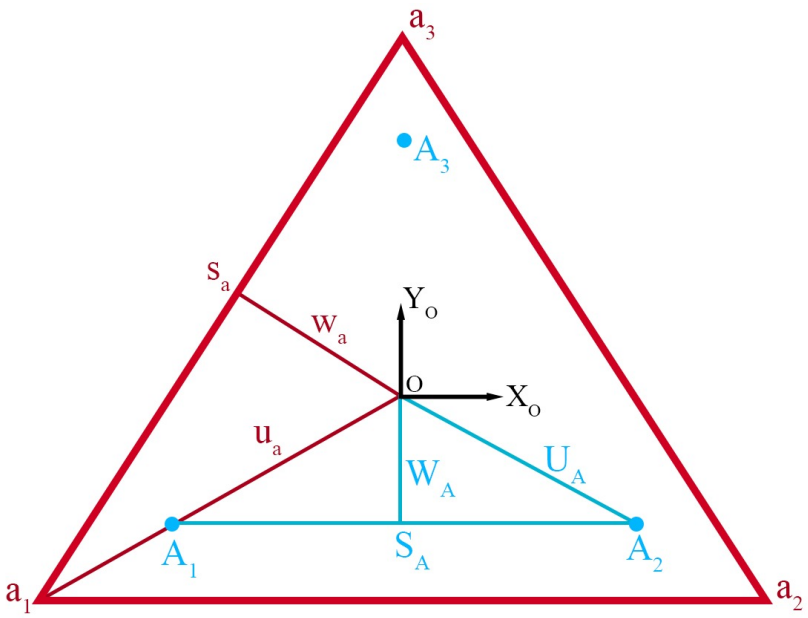

Fig. 16: Fixed Base

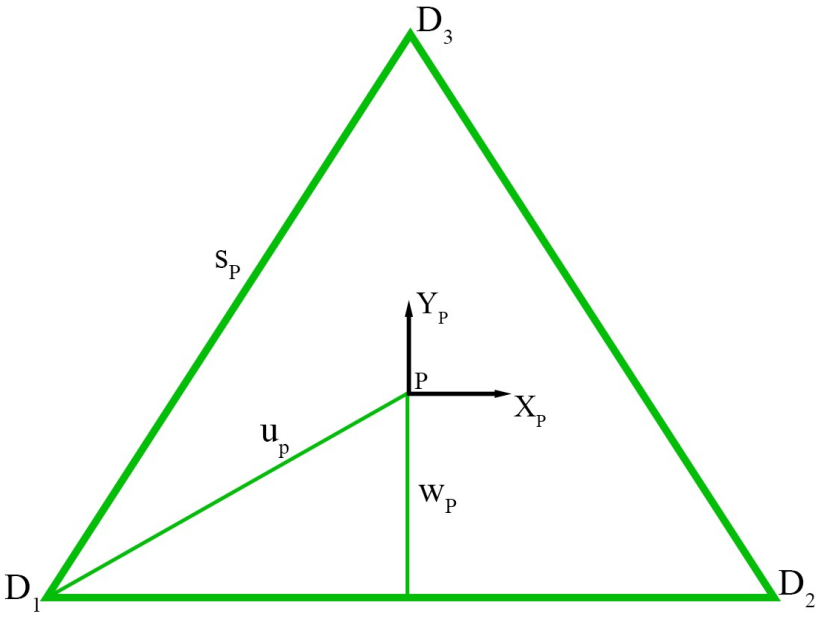

Fig. 17: Moving Platform

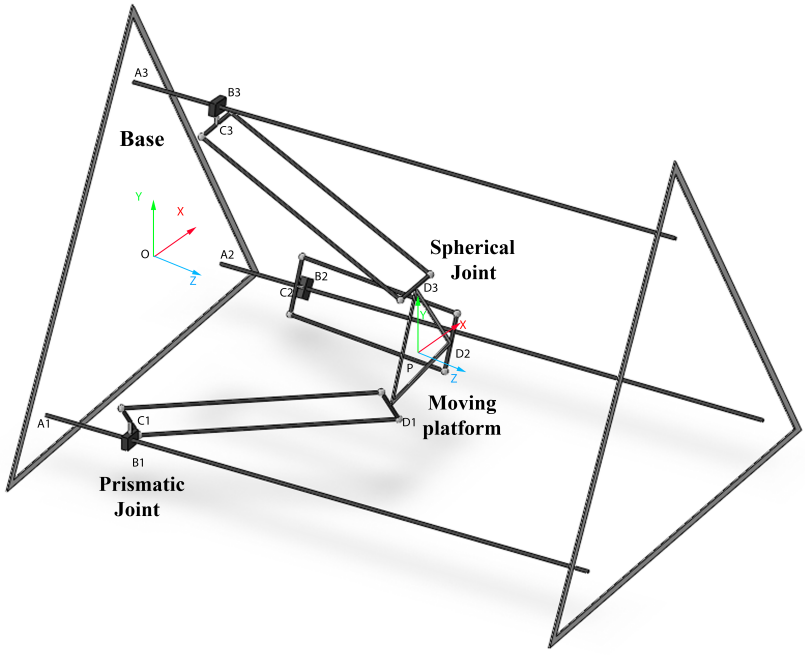

Fig. 18: Kinematic Diagram of 3-DoF Delta Robot

Symbol	Meaning	Value (mm)
s_a	Base triangle side length	700
w_a	Planar distance from O to base side	202
u_a	Planar distance from O to base vertex	404
H	Frame height	870
S_A	Prismatic joints triangle side length	485
W_A	Planar distance from O to prismatic joint triangle side	140
U_A	Planar distance from O to prismatic joint triangle vertex	280
s_p	Platform triangle side length	170
w_p	Planar distance from P to platform triangle side	49
u_b	Planar distance from P to platform triangle vertex	98
L_{min}	Minimum prismatic joint length	70
L_{max}	Maximum prismatic joint length	580
l	Parallelogram link length	365

Table 1: Prismatic-Input Delta Robot Parameters

6.2 Inverse Kinematics

The world coordinate system O is connected to each prismatic joint's guide way via vector a_i and the position of the i-th prismatic joint from the base point of guide way is shown by vector d_i. The c_i connects the center of the i-th prismatic joint to the i-th spherical joint of the i-th cart. The vector l_i is along and equal to the links of the i-th arm and b_i is a vector attached to the moving platform which connects the i-th spherical joint of the i-th arm to the end-effector's reference point.
Assuming that the position of the end-effector is denoted by P, a vector loop closure equation can be written for the i-th arm as follows:

$$P + b_i = a_i + d_i \mathbf{u_i} + c_i + L\mathbf{l_i} \tag{4}$$

For the sake of simplification, the constant terms in Eq. 4 are grouped together and ζ_i is defined as

$$\zeta_i = a_i - b_i + c_i \tag{5}$$

By rearranging the loop closure equation and assuming d is along the z-axis, then:

$$\begin{bmatrix} 0 \\ 0 \\ d_i \end{bmatrix} + \begin{bmatrix} l_{ix} \\ l_{iy} \\ l_{iz} \end{bmatrix} = \lambda_i = \begin{bmatrix} \lambda_{ix} \\ \lambda_{iy} \\ \lambda_{iz} \end{bmatrix} \tag{6}$$

where l_{ix}, l_{iy} and l_{iz} are the components of the vector l_i in x-, y- and z-direction and i is

$$\lambda_i = P - \zeta_i \tag{7}$$

Considering the fact that all the links in each arm have the same length,

$$l_{\mathrm{i}} = \sqrt{{l_{\mathrm{ix}}}^2 + {l_{\mathrm{iy}}}^2 + {l_{\mathrm{iz}}}^2} \tag{8}$$

where l_{i} denotes the length of the links of the i-th arm. By plugging Eq. 6 into Eq. 8, the position of the i-th cart can be expressed as

$$d_{\mathrm{i}} = \lambda_{\mathrm{iz}} \pm \sqrt{{l_{\mathrm{i}}}^2 - {\lambda_{\mathrm{ix}}}^2 - {\lambda_{\mathrm{iy}}}^2} \tag{9}$$

6.3 Position Analysis

From Fig. 19, the position loop closure equation for the *i*-th kinematic chain is

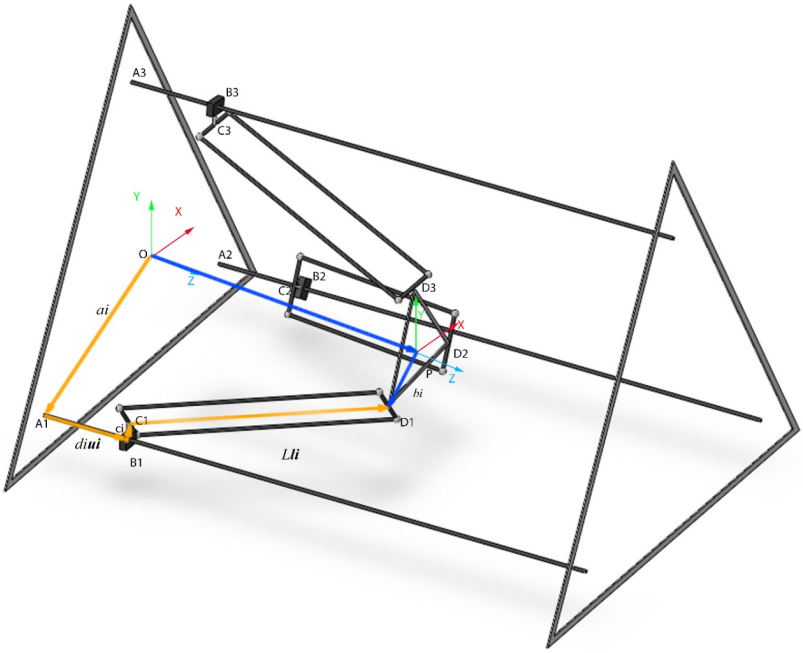

Fig. 19: Loop Closure Equation

$$P + b_{\mathrm{i}} = a_{\mathrm{i}} + d_{\mathrm{i}}\mathbf{u_i} + c_{\mathrm{i}} + L\mathbf{l_i} \tag{10}$$

where P represents the position vector of the moving platform with respect to frame O-XYZ, b_i is the position vector of the offset between the centre of the moving platform and the link, a_i is the position vector of point A_i with respect to O-XYZ, d_i and u_i is the displacement of the i-th slider with respect to A_i and its unit vector, c_i is the position vector between the prismatic joint and the parallelogram-shaped link, also known as the "carriage offset." Lastly, L and l_i are the length and unit vector of the i-th link.

6.4 Velocity Analysis

Taking the derivative of both sides of (10) with respect to time yields

$$\dot{P} = \dot{d_i}\mathbf{u_i} + L\omega_i \times \mathbf{l_i} \tag{11}$$

where $\dot{P}$ is the velocity vector of the platform, $\dot{d_i}$ is the velocity of the i-th prismatic joint, and ω_i is the angular velocity of the i-th link with respect to O. Taking the dot product of (11) with $\mathbf{l}_i^T$ yields the following equation

$$\mathbf{l}_i^T \dot{P} = \mathbf{l}_i^T \mathbf{u}_i \dot{d_i} + L\mathbf{l}_i^T \omega_i \times \mathbf{l_i} \tag{12}$$

We can observe that $L\mathbf{l}_i^T(\omega_i \times \mathbf{l_i}) = 0$. This allows us to simplify (12) into

$$\mathbf{l}_i^T \dot{P} = \mathbf{l}_i^T \mathbf{u}_i \dot{d_i} \tag{13}$$

Now making $\dot{d_i}$ subject gives us this equation

$$\dot{d_i} = \frac{\mathbf{l}_i^T \dot{P}}{\mathbf{l}_i^T \mathbf{u}_i} = J_{di}\dot{P} \tag{14}$$

where $J_{di}\dot{P}$ is the Jacobian matrix of the i-th kinematic chain and is equal to

$$J_{di} = \frac{\mathbf{l}_i^T}{\mathbf{l}_i^T \mathbf{u}_i} \tag{15}$$

This procedure can be repeated for the two other kinematic chains as delta robots have identical kinematic chains. This would yield the following equation for the entire robot

$$\begin{bmatrix} \dot{d}_1 \\ \dot{d}_2 \\ \dot{d}_3 \end{bmatrix} = \begin{bmatrix} J_{d1} \\ J_{d2} \\ J_{d3} \end{bmatrix} \dot{P} \tag{16}$$

6.5 Acceleration Analysis

Taking the derivative of 11 on both sides with respect to time results in

$$\ddot{P} = \ddot{d}_i \mathbf{u}_i + L(\alpha_i \times \mathbf{l}_i) + L\omega_i \times (\omega_i \times \mathbf{l}_i) \tag{17}$$

where $\ddot{P}$ is the acceleration vector of the platform, $\ddot{d}_i$ is the acceleration of the i-th prismatic joint and α_i is the angular acceleration of the i-th link with respect to O. Taking the dot product of (17) with $\mathbf{l}_i$, we notice a similarity as in (11) where $L\mathbf{l}_i(\alpha_i \times \mathbf{l}_i) = 0$. We can also note that $L\mathbf{l}_i\omega_i \times (\omega_i \times \mathbf{l}_i) = -L|(\omega_i \times \mathbf{l}_i)|^2$. We can now simplify and rearrange (17) into

$$\mathbf{l}_i^T \ddot{P} = \mathbf{l}_i^T \mathbf{u}_i \ddot{d}_i - L|(\omega_i \times \mathbf{l}_i)|^2 \tag{18}$$

From (11) and (14), we can deduce the following equation

$$\omega_{\mathrm{i}} \times \mathbf{l}_{\mathbf{i}} = \frac{1}{L}(I_3 - \mathbf{u}_i J_{di})\dot{P} \tag{19}$$

where I_3 is a 3x3 identity matrix. This allows us to further simplify (18) to

$$\mathbf{l}_i^T \mathbf{u}_i \ddot{d}_i = \mathbf{l}_i^T \ddot{P} + L|(\omega_i \times \mathbf{l}_i)|^2 = \mathbf{l}_i^T \ddot{P} + \frac{1}{L}\dot{P}^T (I_3 - \mathbf{u}_i J_{di})^T (I_3 - \mathbf{u}_i J_{di})\dot{P} \tag{20}$$

Now the acceleration of the i-th joint is expressed as

$$\ddot{d}_i = \frac{\mathbf{l}_i^T}{\mathbf{l}_i^T \mathbf{u}_i}\ddot{P} + \dot{P}^T \frac{1}{L(\mathbf{l}_i^T \mathbf{u}_i)}(I_3 - \mathbf{u}_i J_{di})^T (I_3 - \mathbf{u}_i J_{di})\dot{P} \tag{21}$$

This equation can be simplified into

$$\ddot{d}_i = J_{di}\ddot{P} + \dot{P}^T H_i \dot{P} \tag{22}$$

where H is the Hessian matrix of the i-th.chain and is equal to

$$H_i = \frac{1}{L(\mathbf{l}_i^T \mathbf{u}_i)}(I_3 - \mathbf{u}_i J_{di})^T (I_3 - \mathbf{u}_i J_{di}) \tag{23}$$

The equation for the whole parallel robot then becomes,

$$\begin{bmatrix} \ddot{d}_1 \\ \ddot{d}_2 \\ \ddot{d}_3 \end{bmatrix} = \begin{bmatrix} J_{d1} \\ J_{d2} \\ J_{d3} \end{bmatrix} \ddot{P} + \begin{bmatrix} \dot{P}^T & 0 & 0 \\ 0 & \dot{P}^T & 0 \\ 0 & 0 & \dot{P}^T \end{bmatrix} \begin{bmatrix} H_1 & 0 & 0 \\ 0 & H_2 & 0 \\ 0 & 0 & H_3 \end{bmatrix} \begin{bmatrix} \dot{P} \\ \dot{P} \\ \dot{P} \end{bmatrix} \tag{24}$$

Let

$$\ddot{d} = J_d \ddot{P} + V H \hat{V} \tag{25}$$

$$V = \begin{bmatrix} \dot{P}^T & 0 & 0 \\ 0 & \dot{P}^T & 0 \\ 0 & 0 & \dot{P}^T \end{bmatrix}, H = \begin{bmatrix} H_1 & 0 & 0 \\ 0 & H_2 & 0 \\ 0 & 0 & H_3 \end{bmatrix}, \hat{V} = \begin{bmatrix} \dot{P} \\ \dot{P} \\ \dot{P} \end{bmatrix} \tag{26}$$

where $\ddot{d} \in R^n$, is the prismatic joint acceleration, J_d are the Jacobian matrices for serial links i = 1, 2, 3. is the moving platform acceleration and H is the Hessian matrix which is a partial derivative matrix.

6.6 Verification

To verify the accuracy of the model, several number of lengths for the prismatic joints on the hardware setup were measured and then recorded the position of the end platform. The lengths and coordinates were then tested in the inverse and forward kinematics functions on MATLAB and compared to the measurements.

L1 (cm)	L2 (cm)	L3 (cm)	Platform Coordinates
0	0	0	(0, 0, -31.6)
6	0	6	(5, -3.5, -34)
30	33.5	30	(-2, 0.4, -61)
33	33	35	(0, -2, -63)
17	32	24	(-11, -2, -51.5)
35	35	39	(0, -5, -64)

Table 2: Measured Parameters

L1 (cm)	L2 (cm)	L3 (cm)	Platform Coordinates
0	0	0	(0, 0.006731, -31.64)
6	0	6	(6, -3.472, -34.65)
30	33.5	30	(-3.36, 1.946, -62)
33	33	35	(0, -2.225, -64.46)
17	32	24	(-13.2, -0.5, -52.3)
35	35	39	(0, -3.981, -64.22)

Table 3: Forward Kinematics Test

Platform Coordinates	L1	L2	L3
(0, 0, -31.6)	-0.04455	-0.04455	-0.03874
(5, -3.5, -34)	4.5	-0.4286	5.08
(-2, 0.4, -61)	28.55	30.55	29.2
(0, -2, -63)	30.85	30.85	32.6
(-11, -2, -51.5)	16	27.7	23.16
(0, -5, -64)	31.33	31.33	35.82

Table 4: Inverse Kinematics Test

6.7 Open-Loop Kinematics Results

The trajectory equations were given by

$$x = r_x cos(\omega t), \quad y = r_y sin(\omega t), \quad z = z_o + vt \tag{27}$$

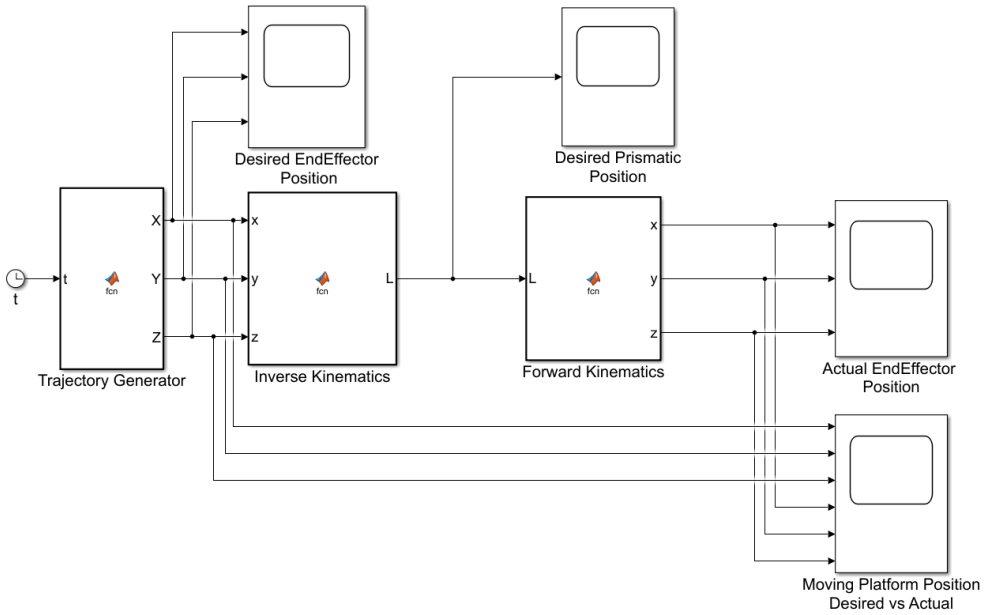

Fig. 20: Simulink Model

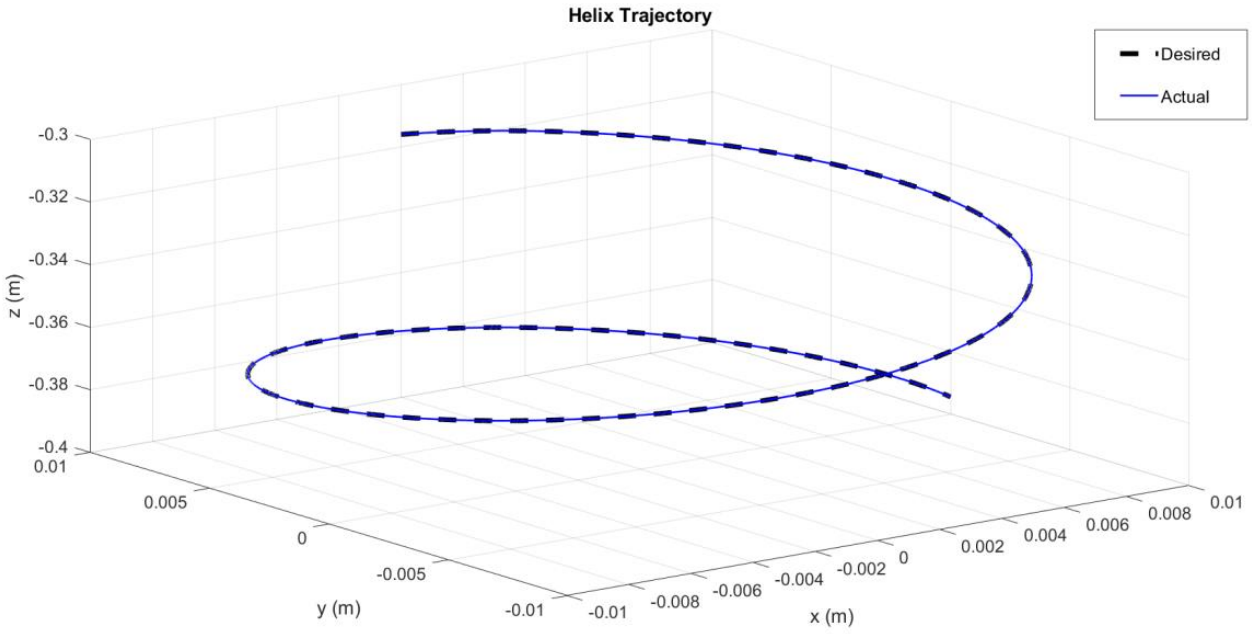

Fig. 21: Kinematics Helix Trajectory

7 Dynamics

The dynamic analysis for the parallel robot tries to solve the driving forces of the three prismatic joints. For a dynamic model, the inertia force and moment of each moving body around the crucial point should be computed. A mass distribution factor is introduced in this study. As a result, the rotational influence of the leg is avoided, simplifying the dynamic analysis. Each leg's bulk can be separated into two portions and placed at its two extremity points, according to [21]. On the moving platform, one component with the proportion of w (0 - 1) can be allocated, and on the slider in the prismatic joints, the other part with the proportion of (1 - w). Leg rotational inertia can be avoided by using this method [17].

Assume that m_p, m_s, and m_l denote the masses of the slider, moving platform and leg link respectively,

$$\overline{m_p} = m_p + 6wm_l \tag{28}$$

$$\overline{m_s} = m_s + 2(1-w)m_l \tag{29}$$

where $\overline{m_p}$ and $\overline{m_s}$ are the equivalent masses of the moving platform and the slider, respectively.

Assume that $\boldsymbol{f} = [f_1, f_2, f_3]^T$ is a vector that represents the actuator forces acting on the prismatic joints through the motor, $\Delta\boldsymbol{d} = [\Delta\boldsymbol{d}_1, \Delta\boldsymbol{d}_2, \Delta\boldsymbol{d}_3]^T$ and $\Delta\boldsymbol{P} = [\Delta X, \Delta Y, \Delta Z]^T$ are the virtual linear displacement vectors of the sliders and the moving platform. Then, according to the principle of virtual work,

$$\boldsymbol{f}\Delta\boldsymbol{d} - \boldsymbol{G_s}^T\Delta\boldsymbol{d} - \boldsymbol{F_s}^T\Delta\boldsymbol{d} - \boldsymbol{G_p}^T\Delta\boldsymbol{P} - \boldsymbol{F_p}^T\Delta\boldsymbol{P} = 0 \tag{30}$$

where $\boldsymbol{G_s}$ is the gravity forces acting on the sliders, $\boldsymbol{F_s}$ is the inertial forces of the sliders, $\boldsymbol{G_p}$ is the gravity forces acting on the moving platform, and $\boldsymbol{F_p}$ is the inertial forces of the moving platform. They can be expressed as following equations

Rearranging (17), an equation for $\dot{P}$ is obtained as following,

$$\dot{P} = {J_\mathrm{d}}^{-1}\dot{d} \tag{31}$$

Thus it can lead to,

$$\Delta\boldsymbol{P} = {J_\mathrm{d}}^{-1}\Delta\boldsymbol{d} \tag{32}$$

By substituting (32) into (30),

$$(\boldsymbol{f} - \boldsymbol{G_s}^T - \boldsymbol{F_s}^T - \boldsymbol{G_p}^T{J_\mathrm{d}}^{-1} - \boldsymbol{F_p}^T{J_\mathrm{d}}^{-1})\Delta\boldsymbol{d} = 0 \tag{33}$$

For any virtual displacement $\Delta\boldsymbol{d}$,

$$\boldsymbol{f} = \boldsymbol{F_s} + {J_\mathrm{d}}^{-T}\boldsymbol{F_p} + \boldsymbol{G_s} + {J_\mathrm{d}}^{-T}\boldsymbol{G_p} \tag{34}$$

Substituting the inertial forces into (34),

$$\boldsymbol{f} = \boldsymbol{M_s}\ddot{d} + {J_\mathrm{d}}^{-T}\boldsymbol{M_p}\ddot{p} + \boldsymbol{G_s} + {J_\mathrm{d}}^{-T}\boldsymbol{G_p} \tag{35}$$

where

$$\boldsymbol{M_s} = \begin{bmatrix} \overline{m_{\mathrm{s}}} & 0 & 0 \\ 0 & \overline{m_{\mathrm{s}}} & 0 \\ 0 & 0 & \overline{m_{\mathrm{s}}} \end{bmatrix}, \boldsymbol{M_p} = \begin{bmatrix} \overline{m_{\mathrm{p}}} & 0 & 0 \\ 0 & \overline{m_{\mathrm{p}}} & 0 \\ 0 & 0 & \overline{m_{\mathrm{p}}} \end{bmatrix} \tag{36}$$

According to (25),

$$\ddot{P} = J_d^{-1}(\ddot{d} - VH\hat{V}) \tag{37}$$

Finally, substituting (32) and (37) into (35), the dynamic model of the parallel robot in general form is derived as

$$\boldsymbol{f} = M(d)\ddot{d} + C(d,\dot{d})\dot{d} + G(d) \tag{38}$$

The above equation is the generalized form of the dynamic model. The actuation force is divided into three terms. The first term, $M(d)\ddot{d}$ represents the generalized mass inertia matrix. The second term, $C(d,\dot{d})\dot{d}$, represents the centrifugal and Coriolis acceleration forces matrix. The third term, $G(d)$, represents the generalized gravitational forces and torques vector. When a trajectory of the moving platform is entered, the input vectors $d, \dot{d}$ and $\ddot{d}$ are calculated via the kinematic model. After that, the actuation force $\boldsymbol{f}$ to control the robot can be acquired via the dynamic model.

8 Simulation results and validation

Combining MATLAB/Simulink and ADAMS allowed for a co-simulation on the virtual prototype of the model. The control algorithm is run in MATLAB during the simulation to create command forces, which are then exported to the ADAMS environment and applied to the virtual prototype's actuators at each time cycle. The virtual prototype's outputs (position and velocity of the moving platform) are measured by "sensors" in ADAMS and then given back to the controller in MATLAB for the next command signal calculation. The co-simulation continues until the end of the simulation time, and the input and output variables for the virtual prototype are shown in 22. The simulation results reveal that the Inverse Dynamics Control approach described above produces nearly no steady-state errors. In practise, however, the payload and dynamic characteristics may not be known precisely.

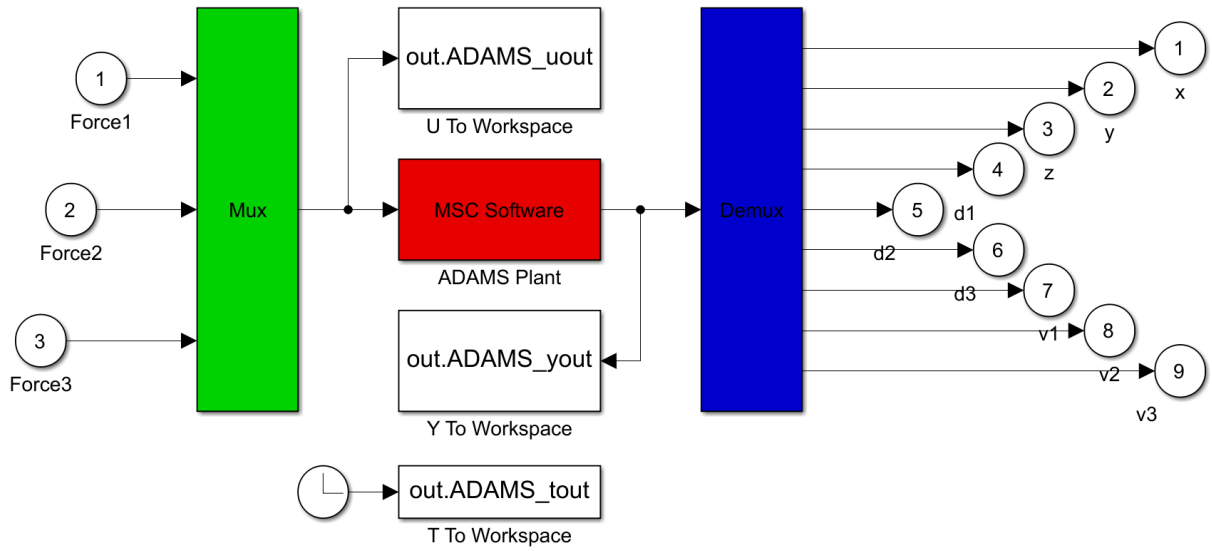

Fig. 22: Control block diagram for the virtual prototype

9 PD Controller

The simplest closed-loop controller for controlling robot manipulators is proportional control with velocity feedback. This control strategy's conceptual application is widespread in motor angular position control.The proportional control plus velocity feedback equation is as follows:

$$f = K_{\mathrm{p}}(d_{\mathrm{d}} - d) + K_{\mathrm{v}}(\dot{d}_{\mathrm{d}} - \dot{d}) \tag{39}$$

where $K_{\mathrm{p}}K_{\mathrm{v}} \in R^{nxn}$ are the proportional and derivative square matrices which are positive definite.

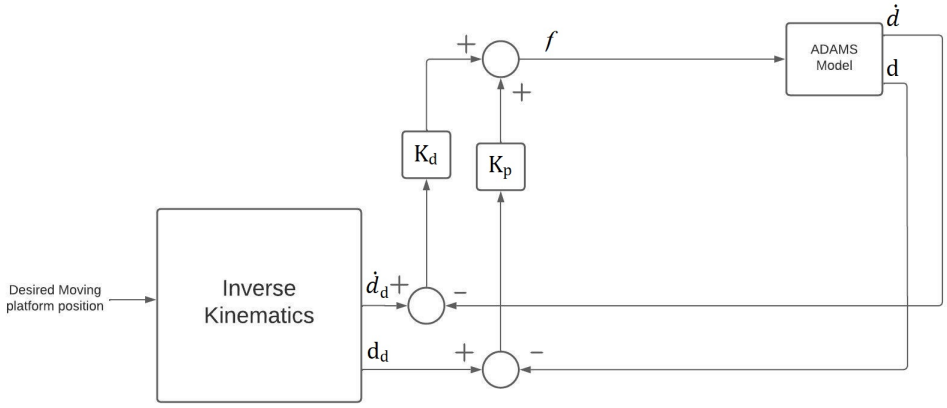

Fig. 23: PD Feedback Loop

We can achieve the position control goal with PD control for robots whose models do not include the gravitational factor (i.e. g(q) = 0). The tuning procedure for PD control is simple in this situation because all that is required is for the design matrices K_{p} and K_{v} to be square matrices and positive.

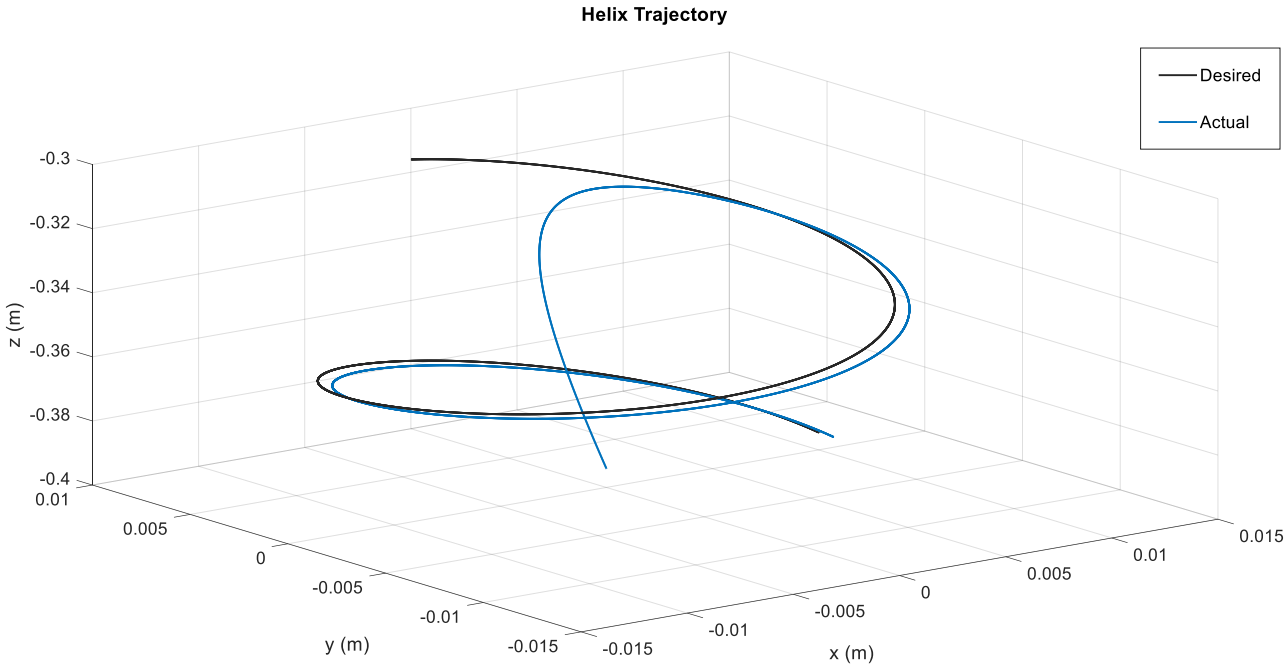

Fig. 24: PD Helix Trajectory

The side and the top view of the helix trajectory are shown below in Fig. 25 and Fig. 26. As it can be seen, there is a steady state error which inhibits the parallel manipulator from following the desired trajectory. The values of the errors and the plots of the desired and actual trajectory are shown below as well.

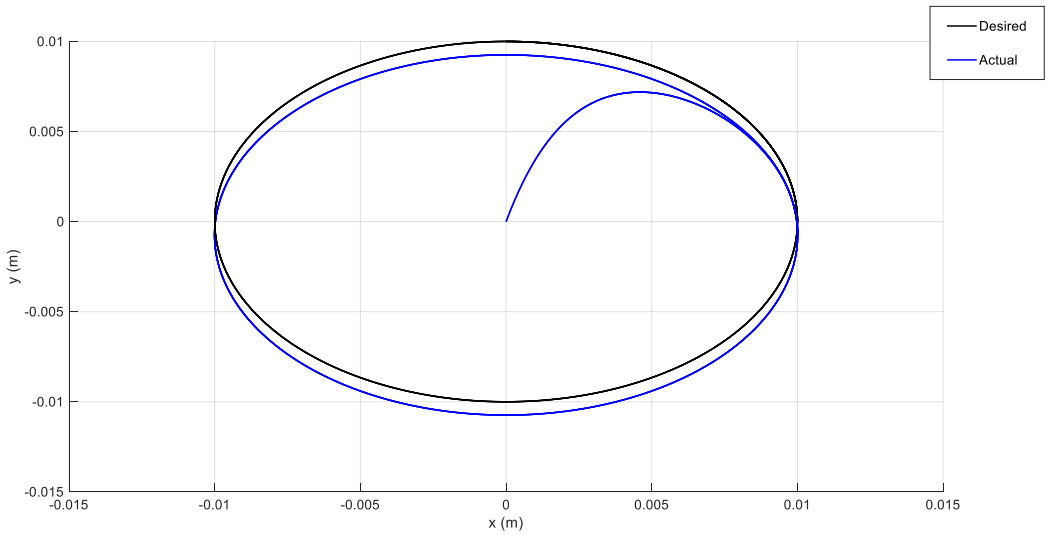

Fig. 25: Top View

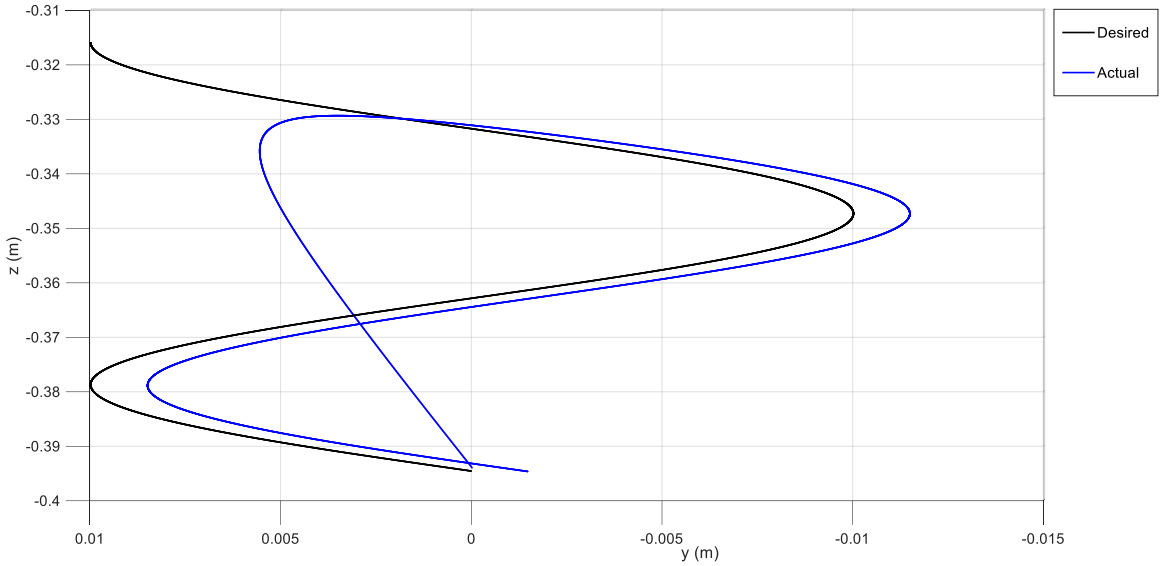

Fig. 26: Side View

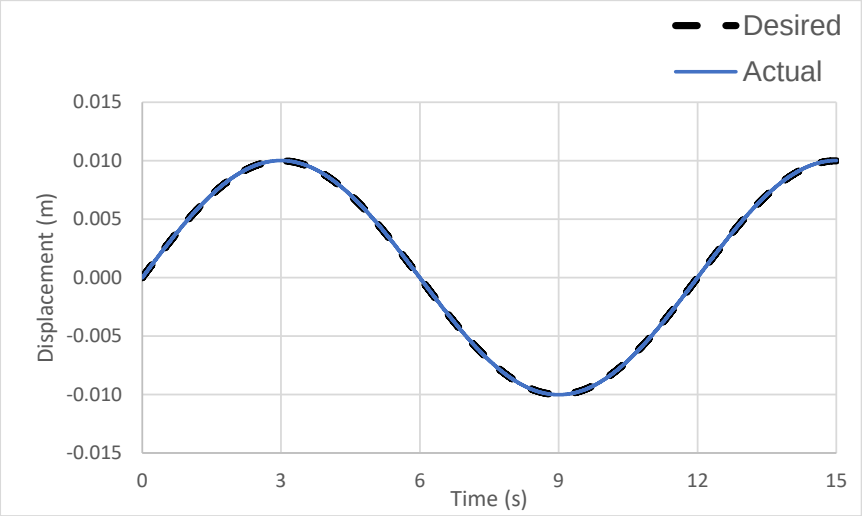

Fig. 27: PD MATLAB X-Trajectory

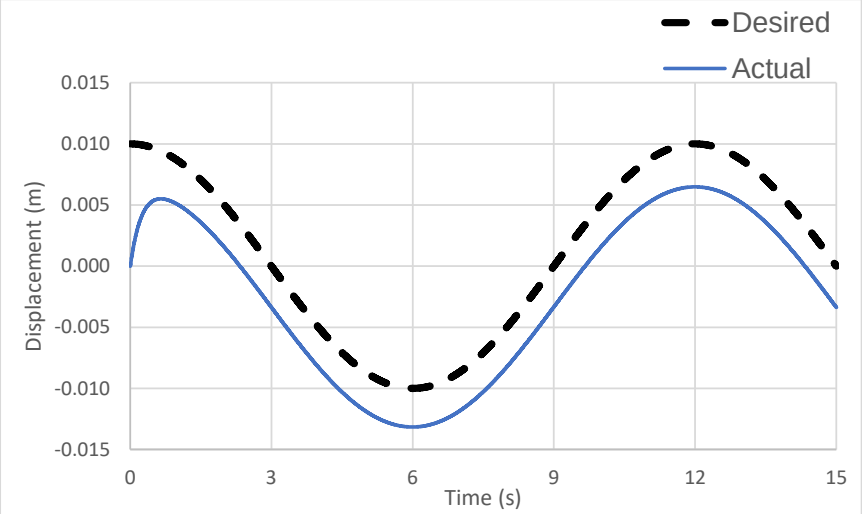

Fig. 28: PD MATLAB Y-Trajectory

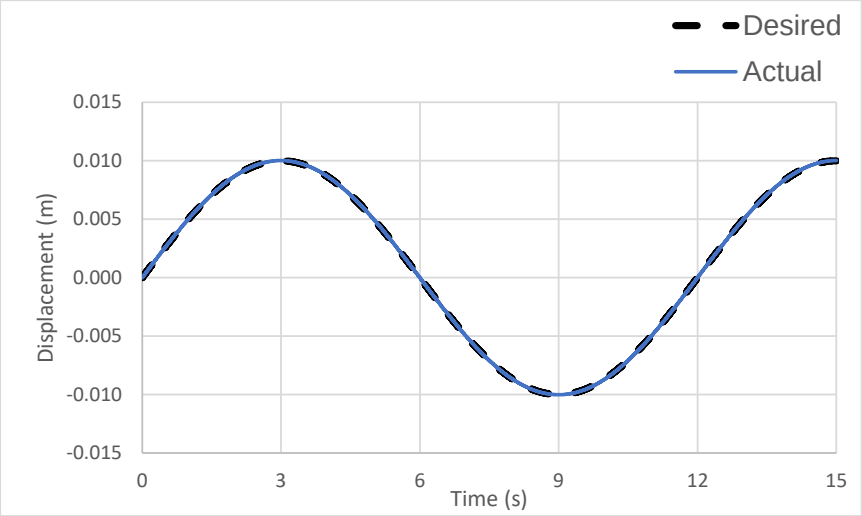

Fig. 29: PD ADAMS X-Trajectory

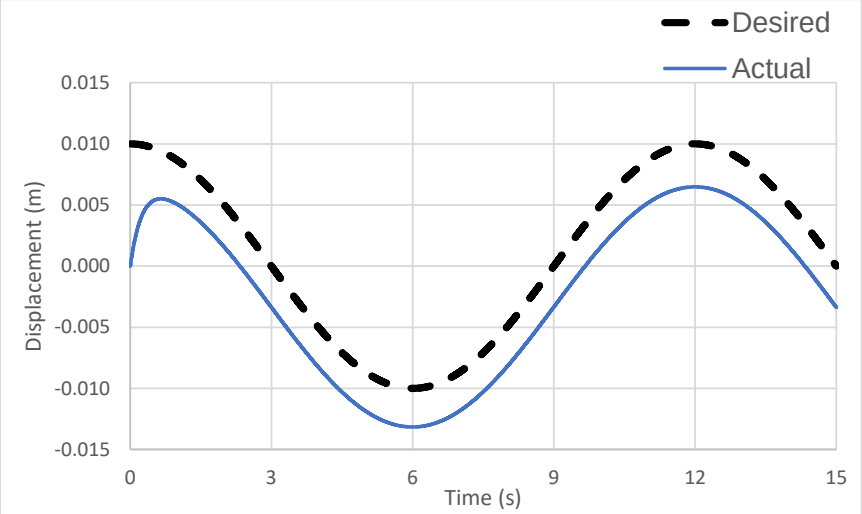

Fig. 30: PD ADAMS Y-Trajectory

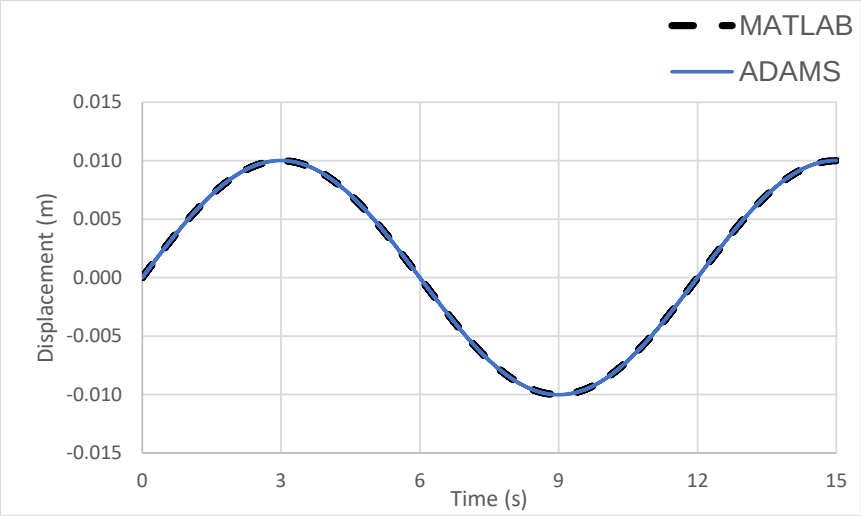

Fig. 31: PD MATLAB VS ADAMS X-Trajectory

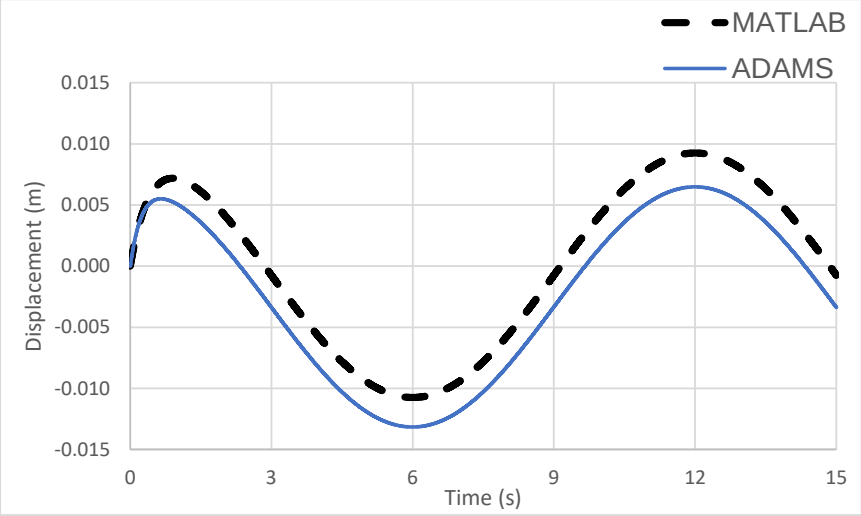

Fig. 32: PD MATLAB VS ADAMS Y-Trajectory

10 PID

When the robot model contains the vector of gravity torques (i.e. $g(d) \neq 0$), and especially when $g(d_{\mathrm{d}}) \neq 0$, where $\mathrm{d_d}$ is the prismatic joint desired position, the position control target cannot be reached using a basic PD control law. In fact, it is possible that the position error q tends to a constant vector, but that this vector is never equal to the vector 0 R^n. Then, from the standpoint of automatic control and in order to achieve the position control goal, it seems reasonable to add an Integral component to the PD control to push the position error to zero. The use of Proportional Integral Derivative (PID) control on robot manipulators is justified by the following.

The PID control law is given by

$$f = K_{\mathrm{p}}\tilde{d} + K_{\mathrm{v}}\dot{\tilde{d}} + K_{\mathrm{i}}\int_0^t \tilde{d}(t)dt \tag{40}$$

where $K_{\mathrm{p}}, K_{\mathrm{v}}, K_{\mathrm{i}} \in R^{nxn}$ are the position, velocity and integral gains, respectively.

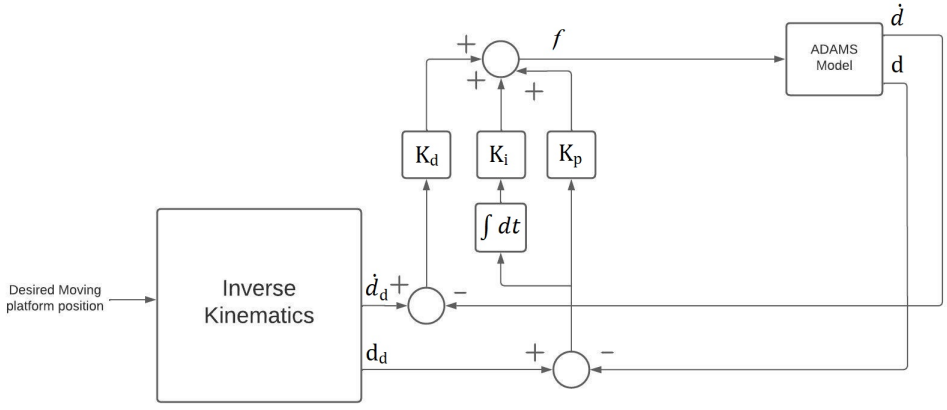

Fig. 33: PID Feedback Loop

PID controllers are now used to control the majority of industrial robot manipulators. The widespread usage of robot manipulators in everyday applications demonstrates the high performance that PID control may achieve in a wide range of applications. In contrast to PD control, however, the tuning

technique for PID controllers, i.e., the procedure for selecting appropriate positive definite matrices K_p, K_v, and K_i, is far from simple. The results for the PID controller are shown below.

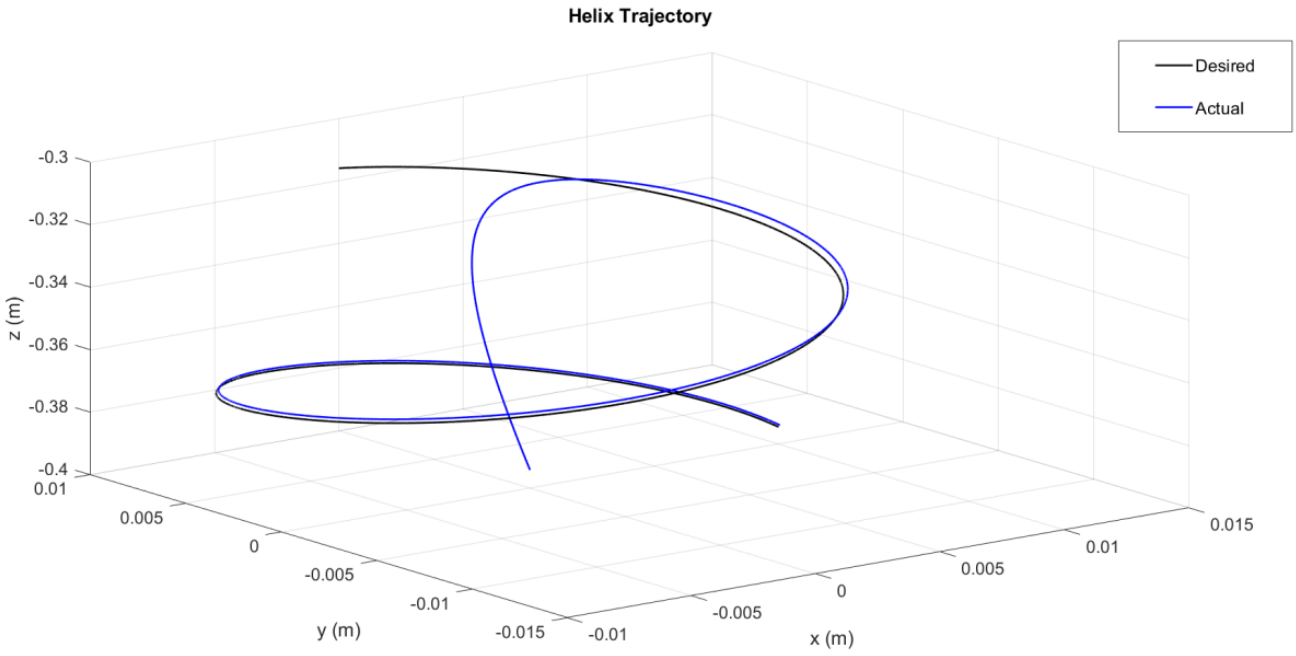

Fig. 34: PID Helix Trajectory

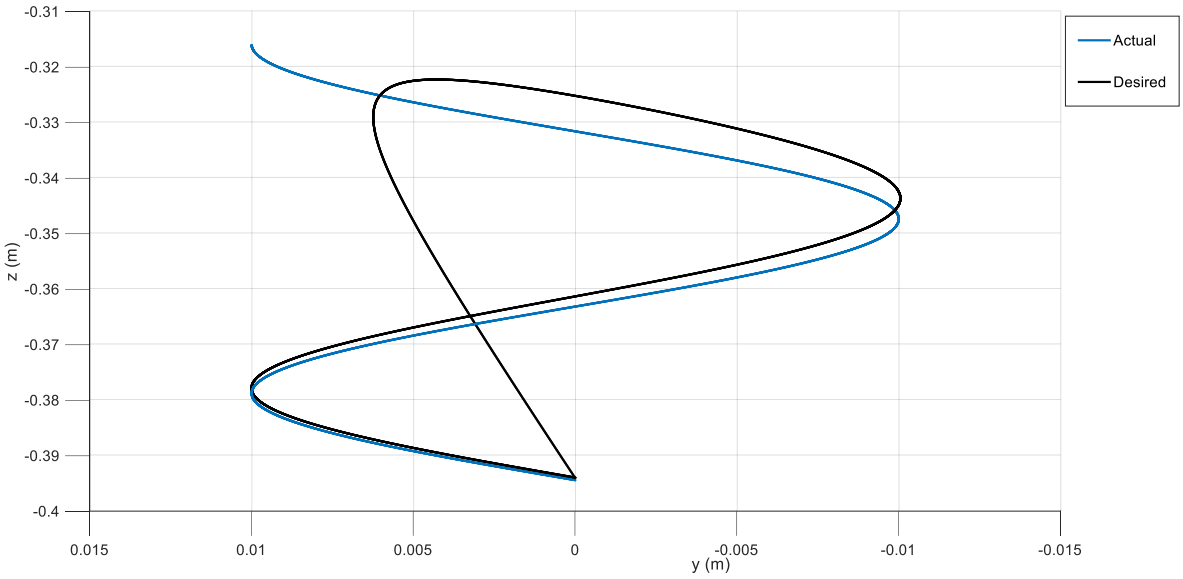

Fig. 35: Side View

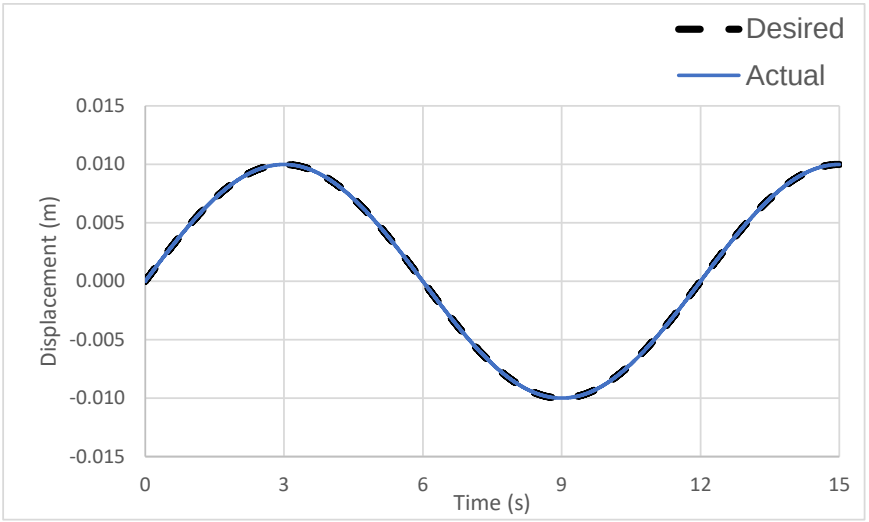

Fig. 36: PID MATLAB X-Trajectory

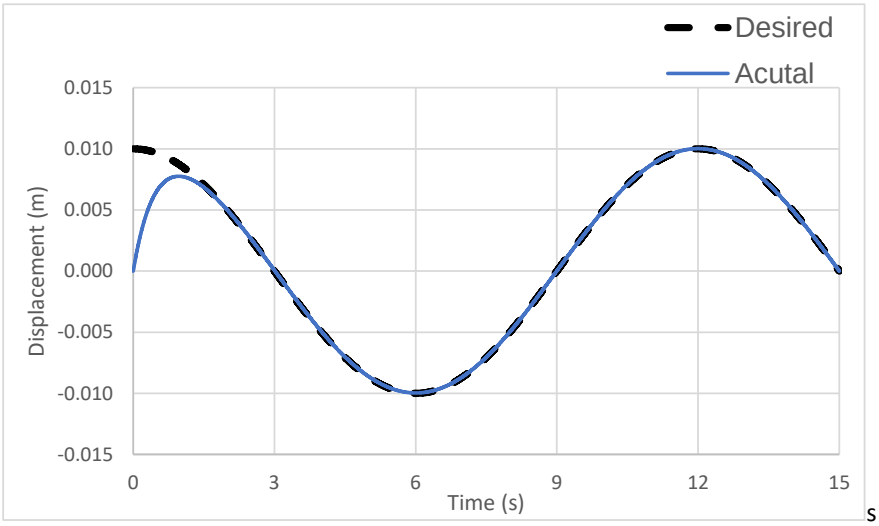

Fig. 37: PID MATLAB Y-Trajectory

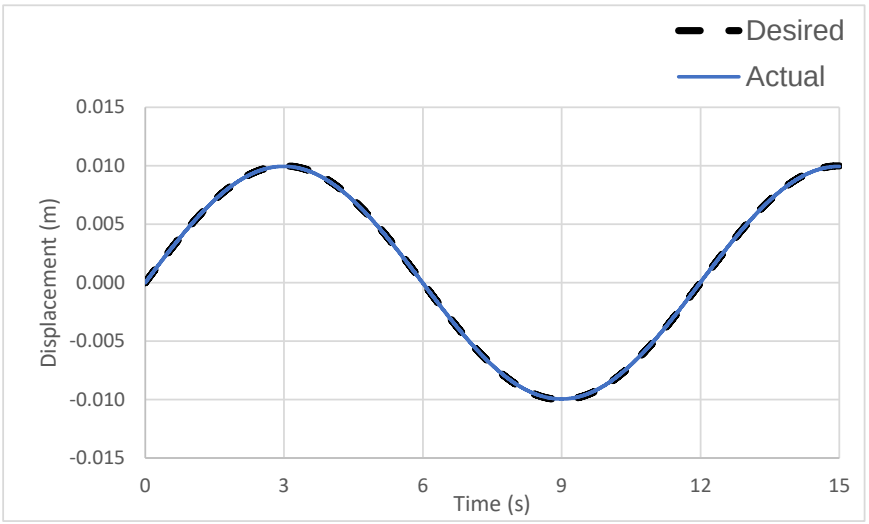

Fig. 38: PID ADAMS X-Trajectory

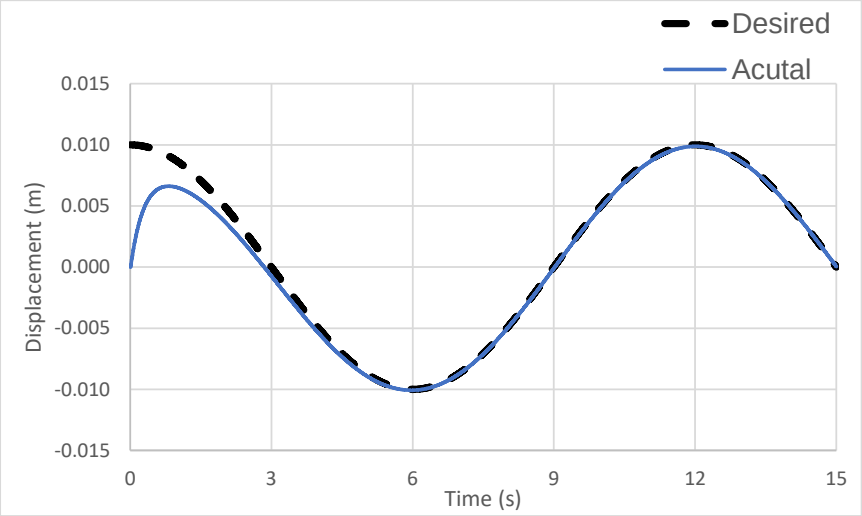

Fig. 39: PID ADAMS Y-Trajectory

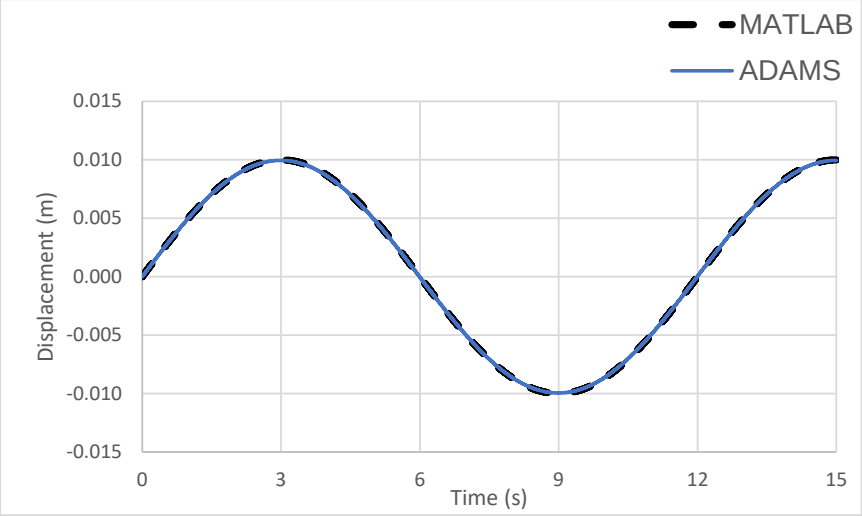

Fig. 40: PID MATLAB VS ADAMS X-Trajectory

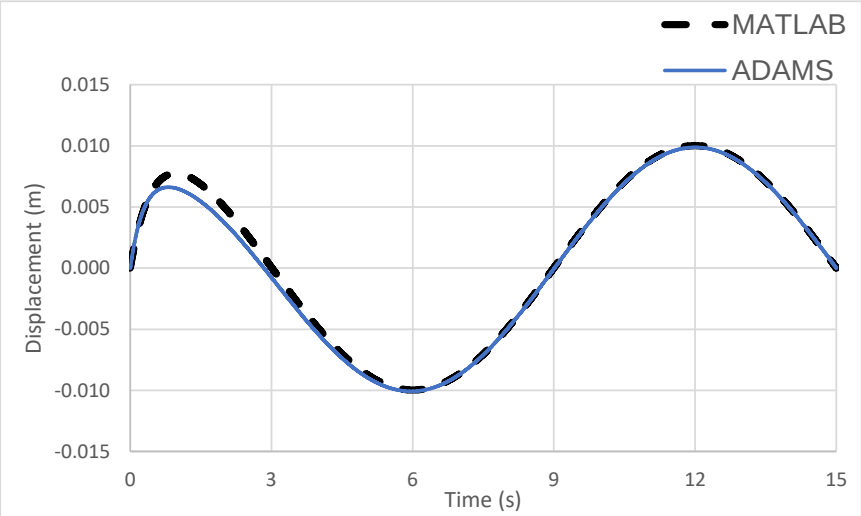

Fig. 41: PID MATLAB VS ADAMS Y-Trajectory

11 Computed Torque Control

Computed-torque control was one of the earliest model-based motion control systems for manipulators, in which the knowledge of the matrices is explicitly used for M(q), $C(q,\dot{q})$ and of the vector G(q). For the Computed Torque Control method in joint space, given a set of vectorial bounded functions q_d, $\dot{q_d}$ and $\ddot{q_d}$ referred to as desired joint positions, velocites and accleration of the moving platform, respectively. It is required to find a vectorial function f such that the positions q, associated to the robot's joint coordinates follow q_d accurately [18].

The task space will be used in the experiment, where the moving platform coordinates are converted to the prismatic joints' coordinates through the inverse kinematics.

$$f = M(d)\ddot{d} + H(d,\dot{d}) \tag{41}$$

where $H(d,\dot{d}) = C(d,\dot{d})\dot{d} + G(d)$

The vector of displacement tracking error then becomes:

$$e = q_d - q \tag{42}$$

Differentiating (42) twice leads to the errors in velocity and acceleration equations

$$\dot{e} = \dot{q_d} - \dot{q} \tag{43}$$

$$\ddot{e} = \ddot{q_d} - \ddot{q} \tag{44}$$

To eliminate the nonlinear term and track the desired trajectory, define the computed-torque control law:

$$f = M(\ddot{q}_{\mathrm{d}} - u) + H \tag{45}$$

where u is the outer loop feedback control input.

Substituting the robot dynamics:

$$M(\ddot{q}_{\mathrm{d}}) + H = M(\ddot{q}_{\mathrm{d}} - u) + H \tag{46}$$

$$M(\ddot{q}_{\mathrm{d}} - \ddot{q}) = Mu \tag{47}$$

$$\ddot{e} = u \tag{48}$$

For a PD feedback with an outer loop feedback control input u,

$$u = -k_{\mathrm{d}}\dot{e} - k_{\mathrm{p}}e \tag{49}$$

The control law becomes

$$f = M(\ddot{q}_{\mathrm{d}} + k_{\mathrm{d}}\dot{e} + k_{\mathrm{p}}e) + H \tag{50}$$

Then the closed-loop dynamics are

$$M(\ddot{q}_{\mathrm{d}} + k_{\mathrm{d}}\dot{e} + k_{\mathrm{p}}e) = 0 \tag{51}$$

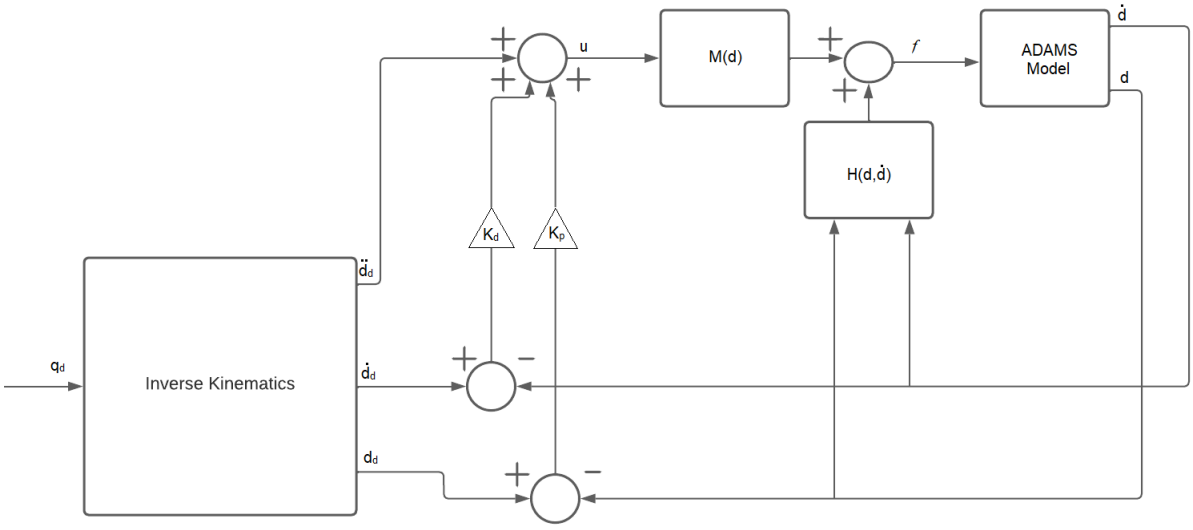

Fig. 42: Computed Torque Control

The error is asymptotically stable as long as k_{d} and k_{p} are all positive. The control law (50) contains the terms $k_{\mathrm{d}}\dot{q} + k_{\mathrm{p}}q$ which are of the PD type. However, these terms are actually pre-multiplied by the inertia matrix $M(q_{\mathrm{d}} - q)$. Therefore this is not a linear controller as the PD, since the position and velocity gains are not constant but they depend explicitly on the position error q.

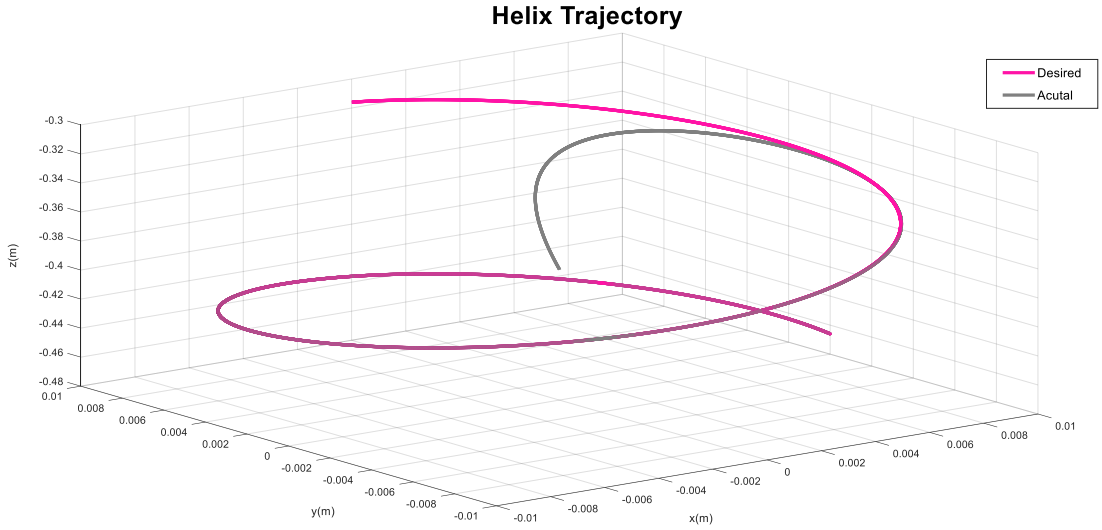

Fig. 43: CTC Helix Trajectory

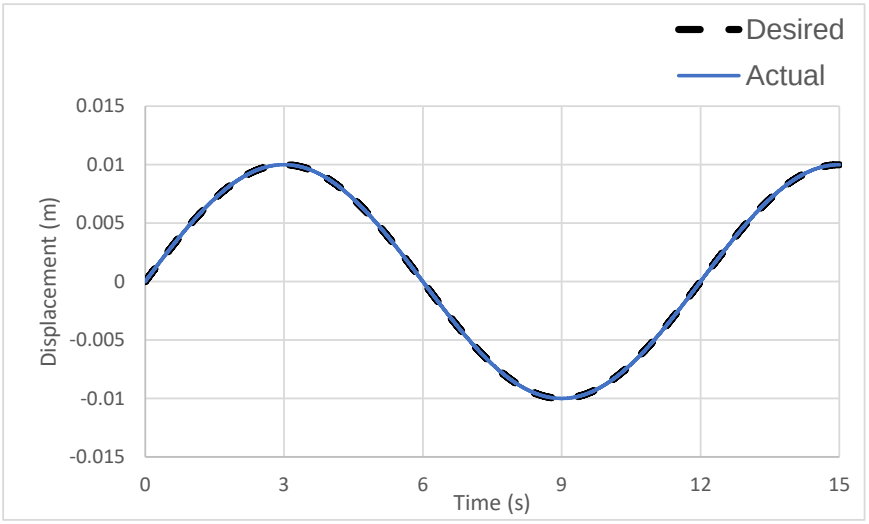

Fig. 44: CTC MATLAB X-Trajectory

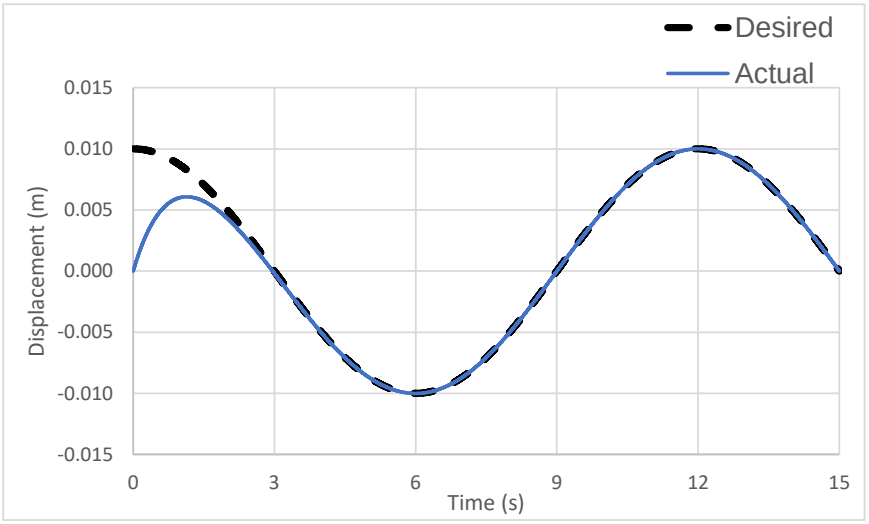

Fig. 45: CTC MATLAB Y-Trajectory

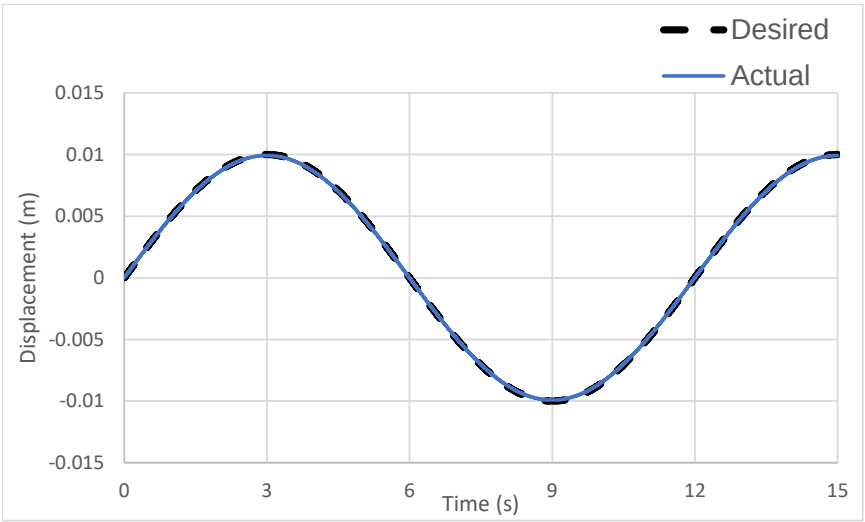

Fig. 46: CTC ADAMS X-Trajectory

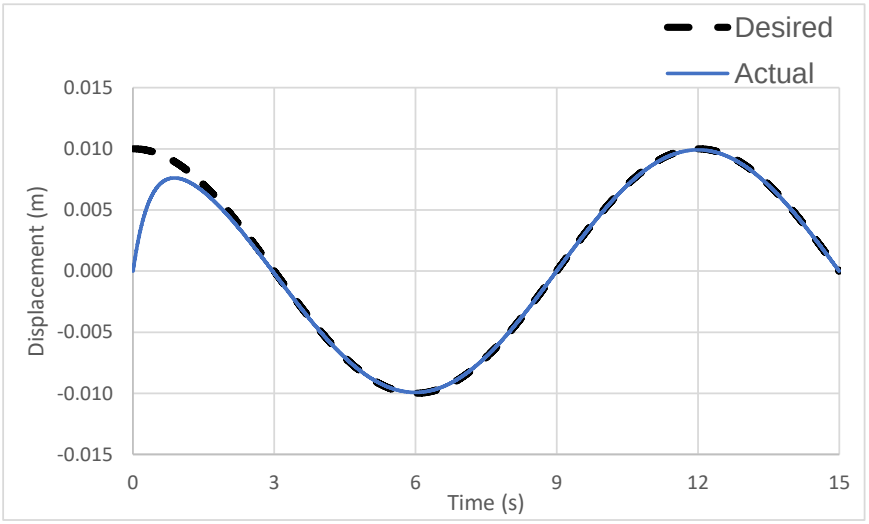

Fig. 47: CTC ADAMS Y-Trajectory

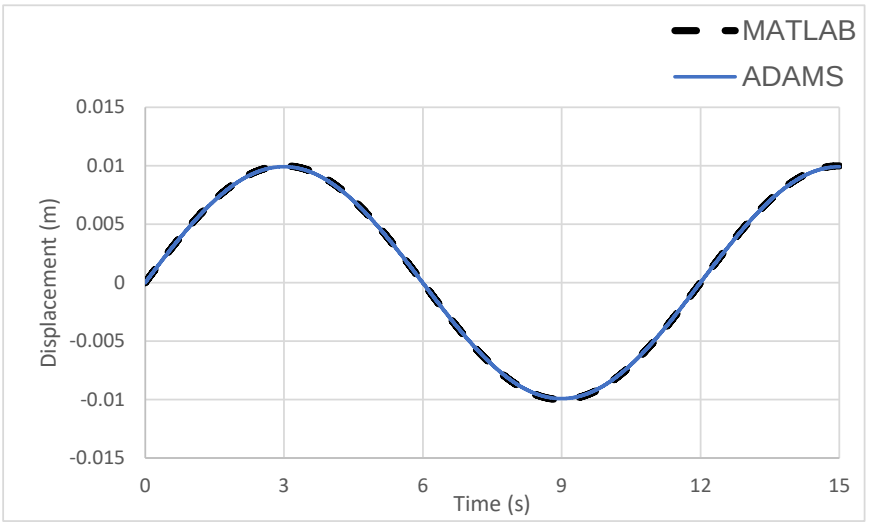

Fig. 48: CTC MATLAB VS ADAMS X-Trajectory

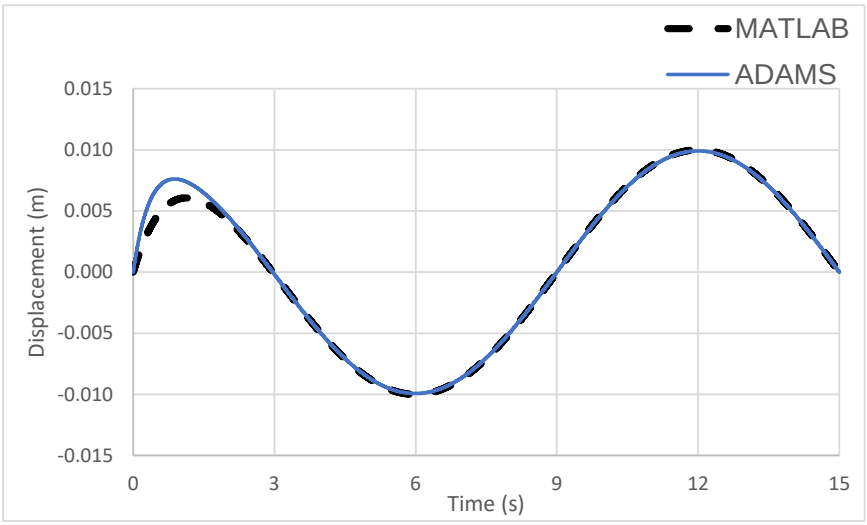

Fig. 49: CTC MATLAB VS ADAMS Y-Trajectory

12 Comparison

As it can be seen, Computed Torque Control provided the most accurate results compared to generic control methods like PD and PID.It can be seen that the X trajectory follows the desired value much more accurate than the actual Y trajectory, this is due to the fact that the initial value of the y is 1 since cos(0) = 1 at t = 0. The position error for each controller is shown in figures Fig.50, Fig.52 and Fig.54. The velocity error for each controller is shown in figures Fig.51, Fig.53 and Fig.55. The figures prove that CTC is superior compared to PD and PID. The

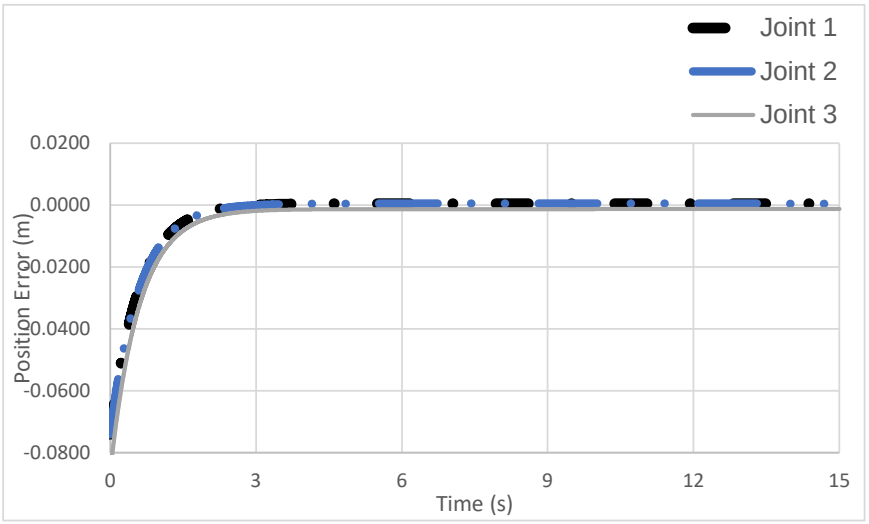

Fig. 50: PD Position Error

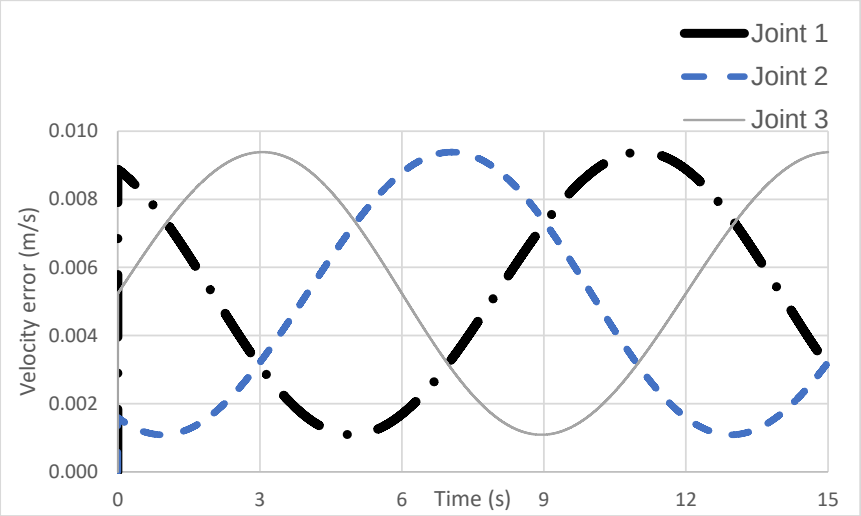

Fig. 51: PD Velocity Error

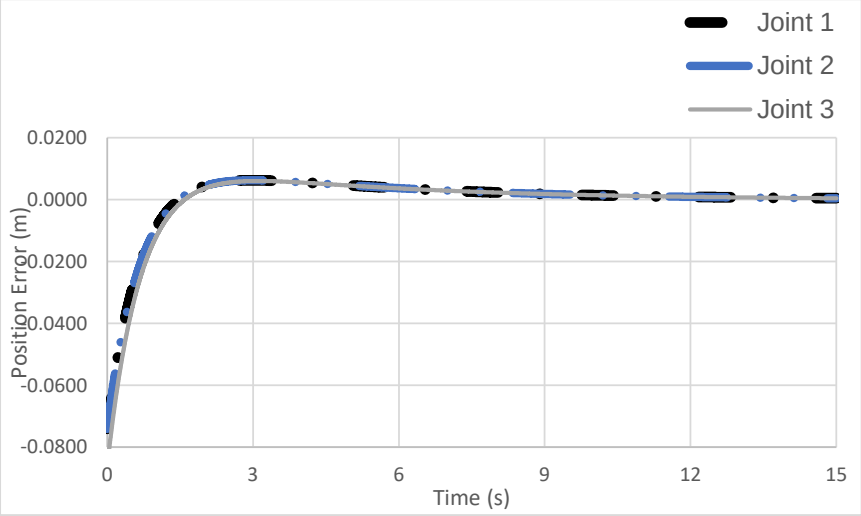

Fig. 52: PID Position Error

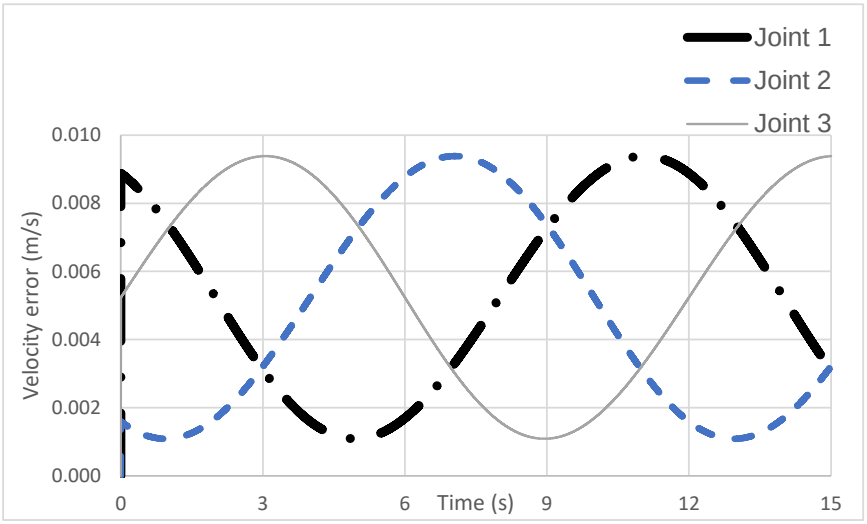

Fig. 53: PID Velocity Error

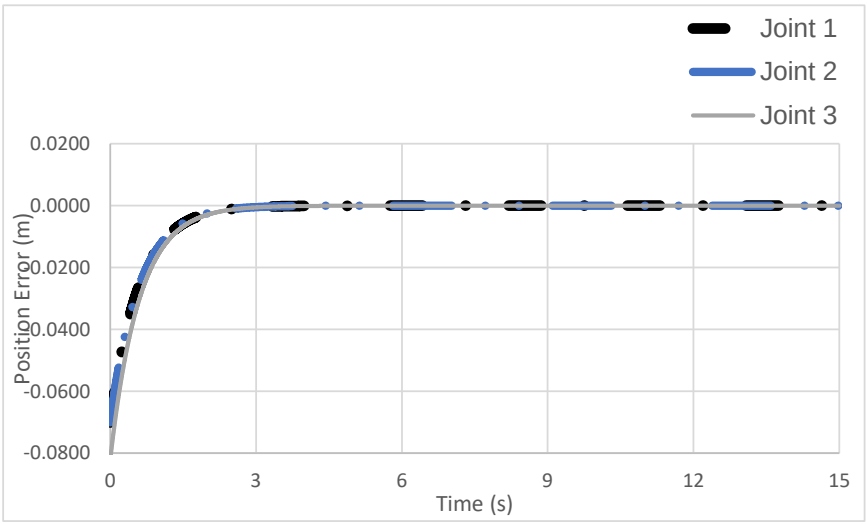

Fig. 54: CTC Position Error

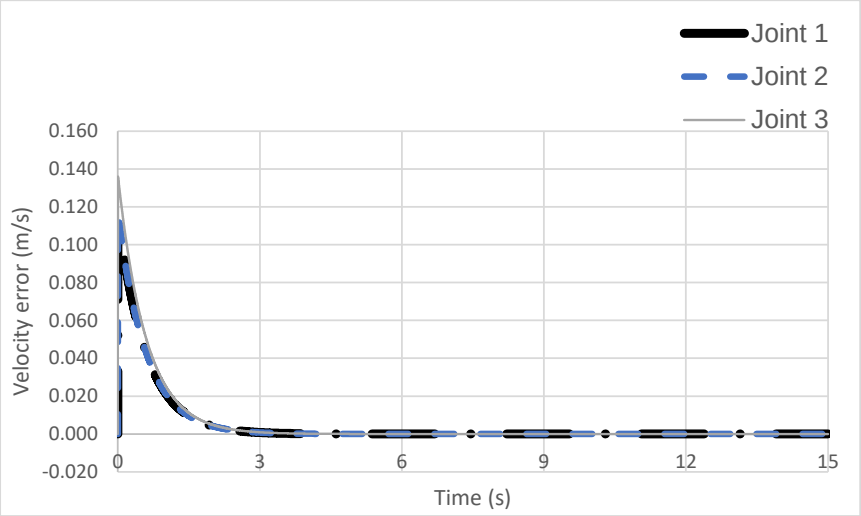

Fig. 55: CTC Velocity Error

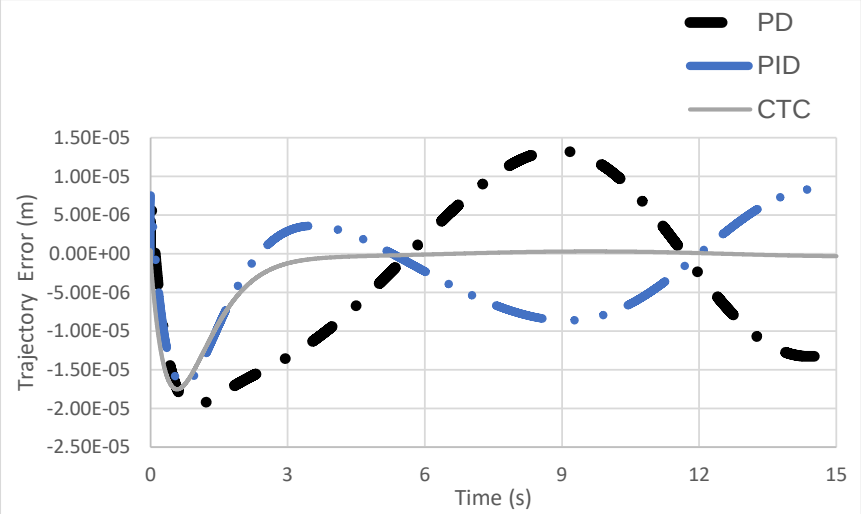

Fig. 56: X Trajectory Error

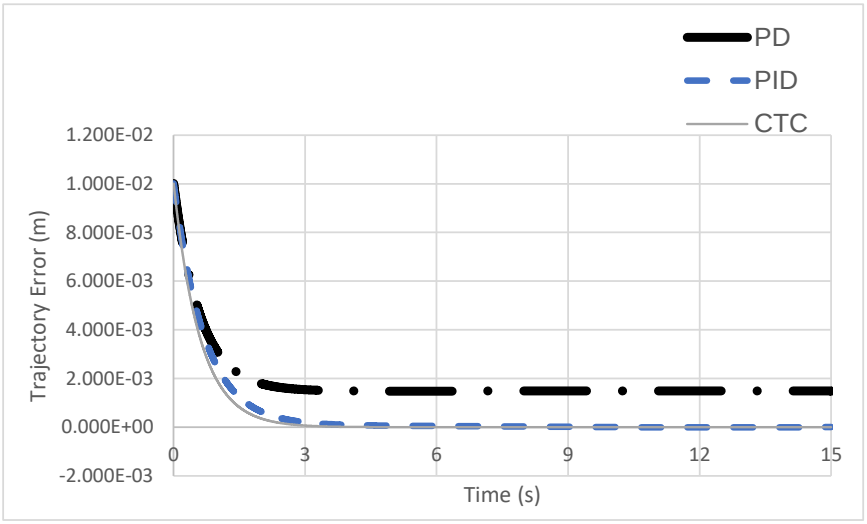

Fig. 57: Y Trajectory Error

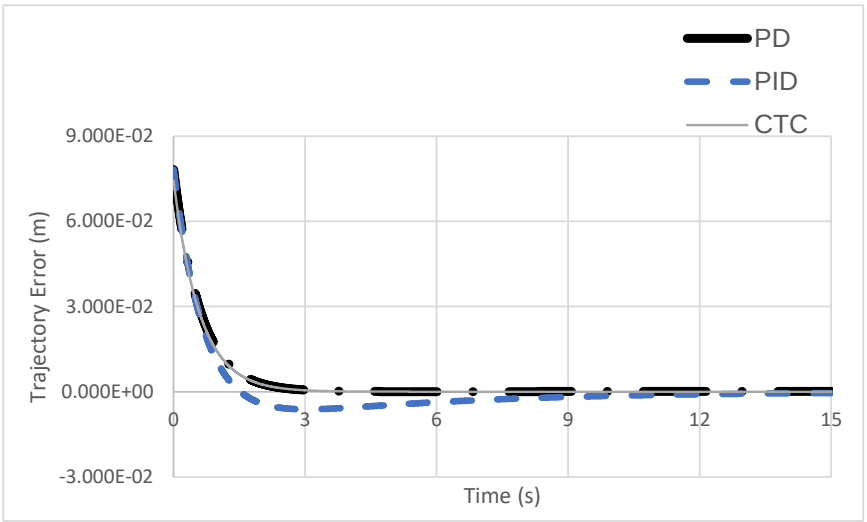

Fig. 58: Z Trajectory Error

The root mean square error (RMSE) is used to compare the outcomes of the PD controller, PID controller, and CTC controller to better evaluate the performance of the proposed CTC controller.where d_d and d denote the desired coordinate and the actual coordinate of the j-th position, respectively. n is the total number of positions, which is automatically determined by MATLAB Simulink. e_x, e_y and e_z represent RMSE of x, y and z directions of the platform, respectively.

Controller	e_x	e_y	e_z
PD	1.1061e-05	0.0022	0.0111
PID	6.6435e-06	0.0015	0.0111
CTC	4.6210e-06	0.0014	0.0105

Table 5: Controller errors in m

$$e = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(d_d - d)^2} \tag{52}$$

As it can be seen, the performance of CTC is superior compared to PD and PID which is kind of expected since CTC is the most used control technique for robot manipulators besides PD+ (PD controller with gravity compensation).

13 Conclusion

In summary, a model of the parallel delta robot was drawn using Solidworks and exported to Adams. The inverse and forward kinematics of the delta robot were derived and introduced to calculate the joint position and to allow for the flexibility of alternating between the task and joint spaces. To allow the robot to function in high-speed and acceleration, the dynamic properties of the robot were derived using principle of virtual work and simulations were carried out to show the effectiveness of the dynamic model. Multiple control methods were used to analyze and determine the best performance between MATLAB and ADAMS by exporting the forces generated from the control unit to ADAMS and acquiring the position and velocity of the joints for a feedback control. Root Mean Square Error (RMSE) was used to determine the controller with best performance and more suitable to be applied to the parallel robot manipulator.

14 Future Works

It is recommended

- To derive the dynamics using the Lagrangian method to take into account the moment of inertia of the links for more accurate results.
- Since CTC is highly dependent on the correctness of the mathematical model, different controllers can be implemented like PD+ controller, MPC controller.
- Implement a disturbance observer if the experimental work is to be added.
- Try different cubic polynoimal equations and to compare their respective results

References

[1] Alain Codourey. Dynamic modeling of parallel robots for computed-torque control implementation. *The International Journal of Robotics Research*, 17(12):1325–1336, 1998.

[2] Philippe Guglielmetti and Roland Longchamp. A closed form inverse dynamics model of the delta parallel robot. *J. Robot. Syst.*, 6, 09 1994.

[3] Liping Wang, Jun Wu, and Jinsong Wang. Dynamic formulation of a planar 3-dof parallel manipulator with actuation redundancy. *Robotics and Computer-Integrated Manufacturing*, 26:67–73, 02 2010.

[4] Michael Stock and Karol Miller. Optimal kinematic design of spatial parallel manipulators: Application to linear delta robot. *Journal of Mechanical Design - J MECH DESIGN*, 125, 06 2003.

[5] Ayman El-Badawy and Khaled Youssef. On modeling and simulation of 6 degrees of freedom stewart platform mechanism using multibody dynamics approach. 07 2013.

[6] Behzad Mehrafrooz, Mohsen Mohammadi, and Mehdi Masouleh. Kinematic sensitivity evaluation of revolute and prismatic 3-dof delta robots. pages 225–231, 10 2017.

[7] Ka-Tjun Oen and Li-Chun Wang. Extreme reaches and maximal reachable workspace for rotary tools mounted on a stewart platform manipulator. *Journal of the Chinese Institute of Engineers*, 29:967–974, 09 2006.

[8] Yongjie Zhao. Dynamic optimum design of a 3u p s- p ru parallel robot. *International Journal of Advanced Robotic Systems*, 13:172988141667617, 12 2016.

[9] Yangmin Li. Dynamic modeling and robust control of a 3-prc translational parallel kinematic machine. *Robotics and Computer-Integrated Manufacturing*, 25:630–640, 06 2009.

[10] Wisama Khalil and Ouarda Ibrahim. General solution for the dynamic modeling of parallel robots. *Journal of Intelligent and Robotic Systems*, 49, 05 2007.

[11] Ping-Lang Yen and Chi-Chung Lai. Dynamic modeling and control of a 3-dof cartesian parallel manipulator. *Mechatronics*, 19:390–398, 04 2009.

[12] Yangmin Li. Dynamic modeling and robust control of a 3-prc translational parallel kinematic machine. *Robotics and Computer-Integrated Manufacturing*, 25:630–640, 06 2009.

[13] Mingkun Wu, Jiangping Mei, Jinlu Ni, and Weizhong Hu. Trajectory tracking control of delta parallel robot based on disturbance observer. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 235:095965182097482, 12 2020.

[14] I-Hsum Li, Hsin-Han Chiang, and Lian-Wang Lee. Development of a linear delta robot with three horizontal-axial pneumatic actuators for 3-dof trajectory tracking. *Applied Sciences*, 10:3526, 05 2020.

[15] Chien Cheah and Xiang Li. *Task-Space Sensory Feedback Control of Robot Manipulators*, volume 73. 01 2015.

[16] Richard Murray, Zexiang Li, and Shankar Sastry. A mathematical introduction to robot manipulation. 29, 12 2010.

[17] Peng Xu, Bing Li, and Chi-Fai Chueng. Dynamic analysis of a linear delta robot in hybrid polishing machine based on the principle of virtual work. pages 379–384, 07 2017.

[18] Yongjie Zhao. Dynamic optimum design of a 3u p s- p ru parallel robot. *International Journal of Advanced Robotic Systems*, 13:172988141667617, 12 2016.