Contents

Chapter 1 Cartesian Robot Project

Written by] Ali Gamal Ali
120200026
Egypt-Japan University of Science and Technology
May 31, 2023

Introduction

The project presents the Design, Modelling and the Simulation of a widely used Mechatronics system, ”3 axis cartesian robot” (HBOT). Throughout this project, the robot’s mechanical system was designed and assembled using SOLIDWORKS, the simulation of robot motion was conducted using Simulink and Simscape Multibody, and the trajectory planning and Control of Robot motion utilizing the obtained robot’s kinematics.

Background

Cartesian Robots are found in the most demanding of applications. It’s a very common and simple robotic technology that manufacturers have used for decades. But what exactly are these robots? For which tasks can they be used? Cartesian robots are excellent choice for applications and poor for others. A major factor is their limited degrees of freedom which compared to 6 axis robots which have higher manipulation abilities. Cartesian Robots are much more rigid in their motion. This may seem like a weakness in their design. However, this feature makes cartesian robots excel all other robot types for certain applications.

- CNC applications
- 3D printing
- Plasma/laser cutting
- Pick and Place

Cartesian Robots can excel in some applications that are commonly performed upon one plane or dimensions like the applications mentioned above. Cartesian robots can achieve the highest levels of precision of all the robot types. Why is this? Put simply, that more rigid and less freedom of movement feature discussed earlier allows these robots to be extremely precise.

Figure 1.1: Lucas Robot

Simulation Results

1.0.1 HBOT Mechanical Design

The Robot's mechanical Design was illustrated from "Lucas Robotics 3 Axis Cartesian Robot" shown in Figure 1.1. It was fully constructed and Assembled using Solidworks to create the 3 axis mechanism of the robot. After finalizing the assembly, the model was exported as .xml file to be converted to simulink file using **smimport** matlab command. Shown in Figure 1.2, the Solidworks model and the model after being imported to simscape. also shown in Figure 1.3, the simulink model after being imported to matlab.

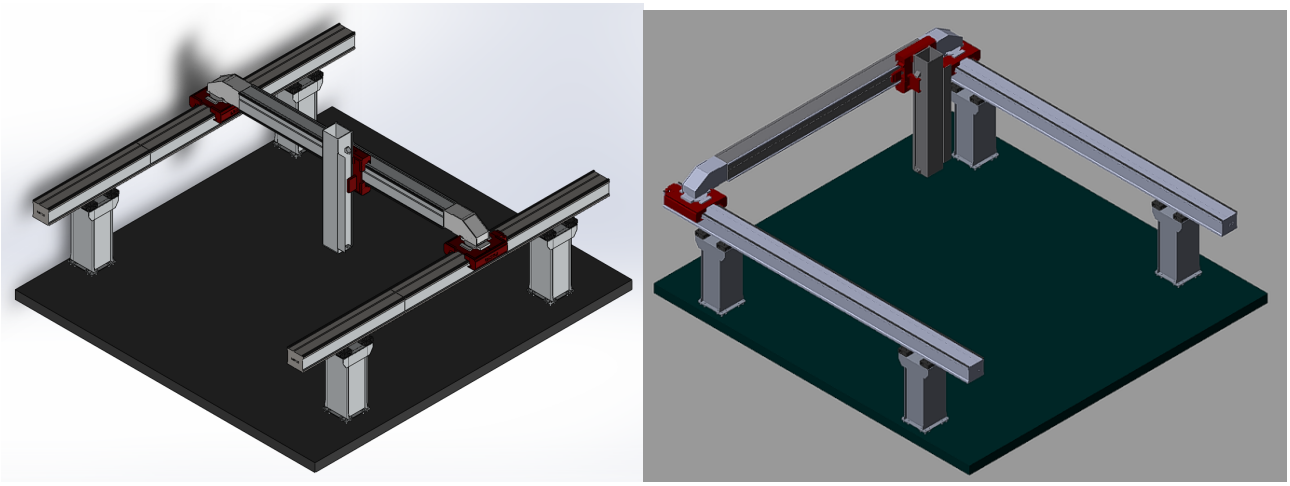

Figure 1.2: Solidowkrs Model/ Imported Simscape model

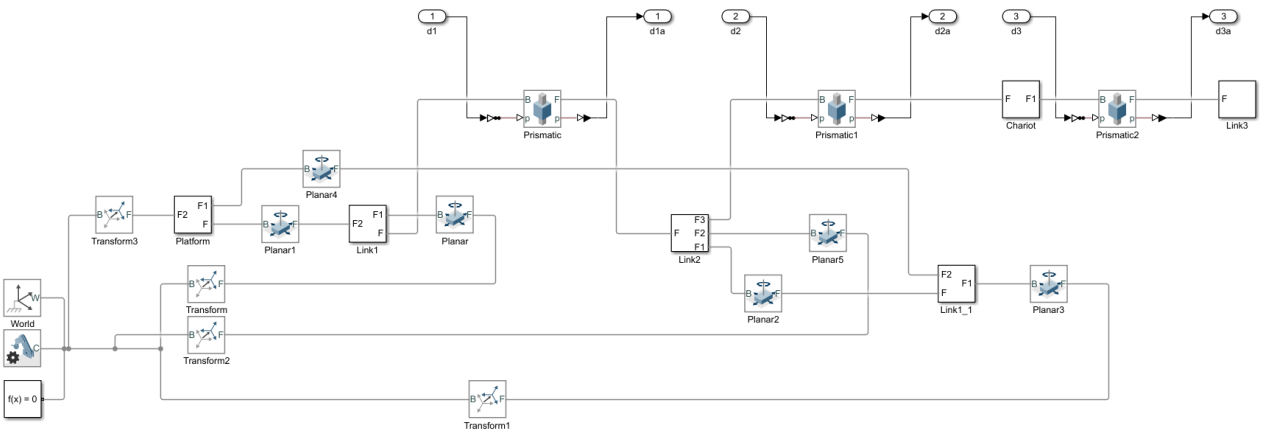

Figure 1.3: Imported Simulink Model

1.0.2 HBOT DH Parameters

DH TABLE

Joint	θ	α	d	a
1	0	$-\frac{\pi}{2}$	d_1^*	0
2	$\frac{\pi}{2}$	$-\frac{\pi}{2}$	d_2^*	0
3	0	0	d_3^*	0

Table 1.1: Denavit-Hartenberg parameters for HBOT.

1.0.2.1 Forward Kinematics

Apply the DH parameters to the A Matrix

$$\begin{bmatrix} c_1c_2 & -c_1s_2 & s_1 & a_1c_1 \\ s_1c_2 & -s_1s_2 & -c_1 & a_1s_1 \\ s_2 & c_2 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

To compute Forward Kinematics, we multiply the resulting A matrices from base to end effector using

$$^0A_1\,{}^1A_2\,{}^2A_3 = {}^0\,T_3$$

$$\left(\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & -1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}\right) \cdot \left(\begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}\right) \cdot \left(\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}\right) = \left(\begin{bmatrix} 0 & 0 & 1 & d3 \\ 0 & 1 & 0 & -d2 \\ -1 & 0 & 0 & d1 \\ 0 & 0 & 0 & 1 \end{bmatrix}\right)$$

We can get the transformation matrix from the translation vector of ${}^0T_3$:

Forward Model

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} d3 \\ -d2 \\ d1 \end{bmatrix}$$

1.0.2.2 Inverse Kinematics

Relative to the base frame, the primsatic joint variables were computed to be:

Inverse Model

$$d_1 = X,\ d_2 = Y,\ d_3 = Z$$

The Derivation of Robot's Forward and Inverse Kinematics is a primary step in develop a trajectory for a robot, and to make independent joint control. So the previous equations were inserted in MATLAB functions as illustrated in Figure 1.4.

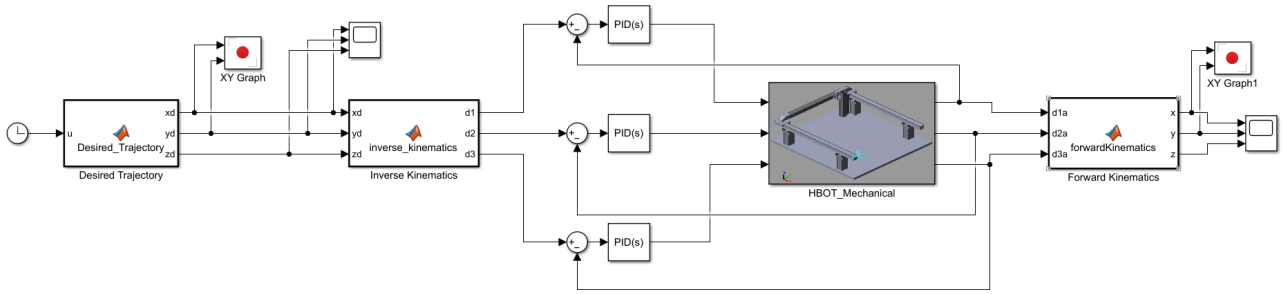

Figure 1.4: Final Simulink Model

Testing and Evaluation Results

1.0.3 Trajectory Planning

In order to test the robot motion, trajectory equations had to be provided for the robot to follow. The following are the three equations provided for the robot:

Trajectory Equations

Linear Trajectory Equation:

$$x_d = x_0 + v_x * t;\ y_d = y_0 + v_y * t;\ z_d = z_0 + v_z * t;$$

Circular Trajectory Equation:

$$x_d = x_0 + r * cos(w * t);\ y_d = y_0 + r * sin(w * t);\ z_d = z_0;$$

Parabolic Trajectory Equation:

$$x_d = x_0 + v_x * u + (1/2) * a * t^2; y_d = y_0 + v_y * u + (1/2) * a * t^2; z_d = z_0;$$

The previous Trajectory Equations were inserted to the MATLAB function whose output is the desired pose of the robot X_d, Y_d, Z_d. the desired pose acts as the input to inverse kinematics function which obtains the prismatic joint variables required to actuate the robot to desired pose. Finally the actual position of the robot X_a, Y_a, Z_a was computed using forward kinematics function, with the actual joint variables as input to the function. the difference between the desired and actual pose was then computed to apply the independent joint control using PID Controller.

1.0.4 Trajectory (Without PID Controller)

Following Analysis were conducted on the circular trajectory, whose plotted output using xy graph was is shown in Figure 1.5.

1.0.5 Trajectory (With PID Controller)

After Tuning the PID Controller, the error in end-effector's positioning significantly decreased as shown in Figure 1.6.

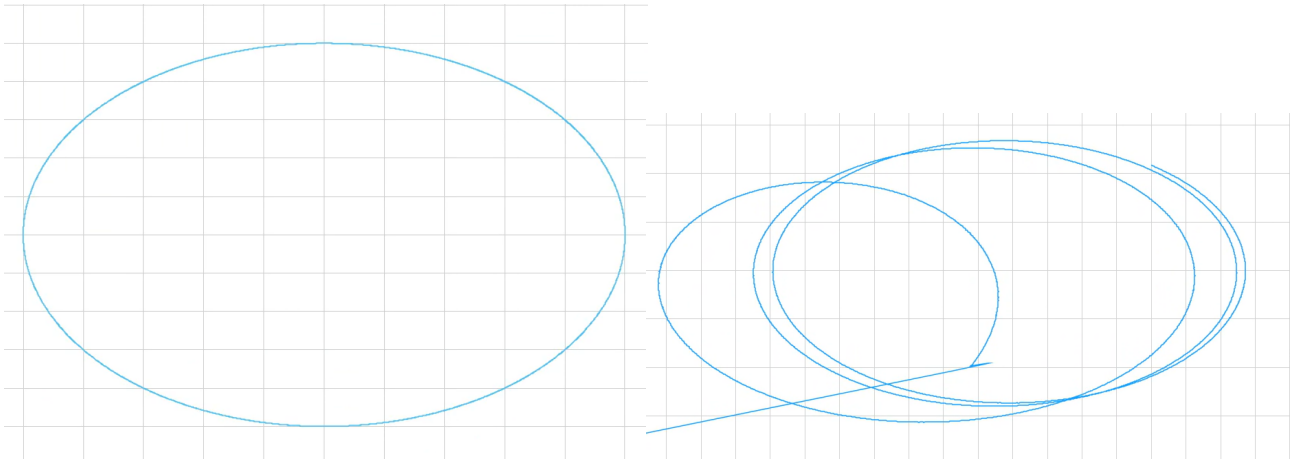

Figure 1.5: Desired/ Actual (PID not Tuned)

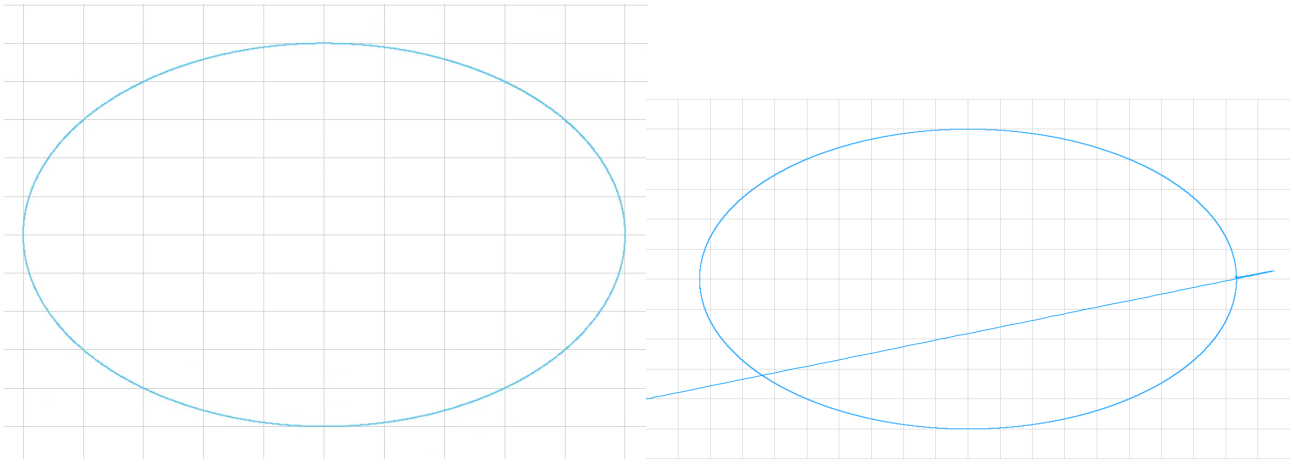

Figure 1.6: Desired/ Actual (PID Tuned)

Chapter 2 Fuzzy Logic Controller

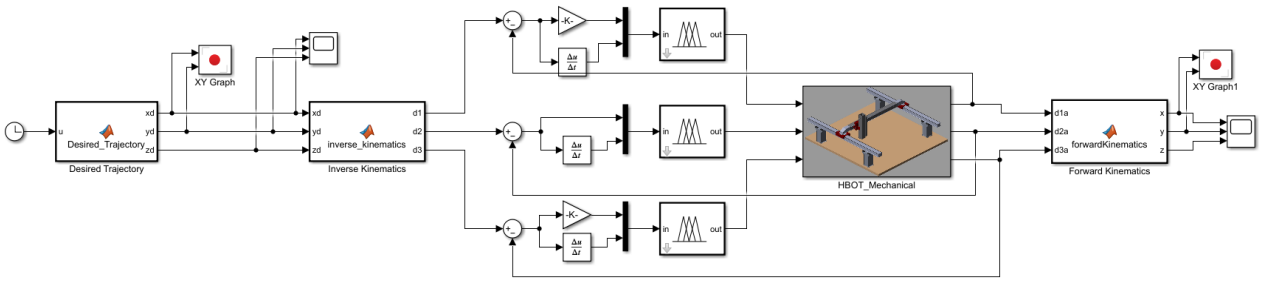

Figure 2.1: fuzzy controller simulink

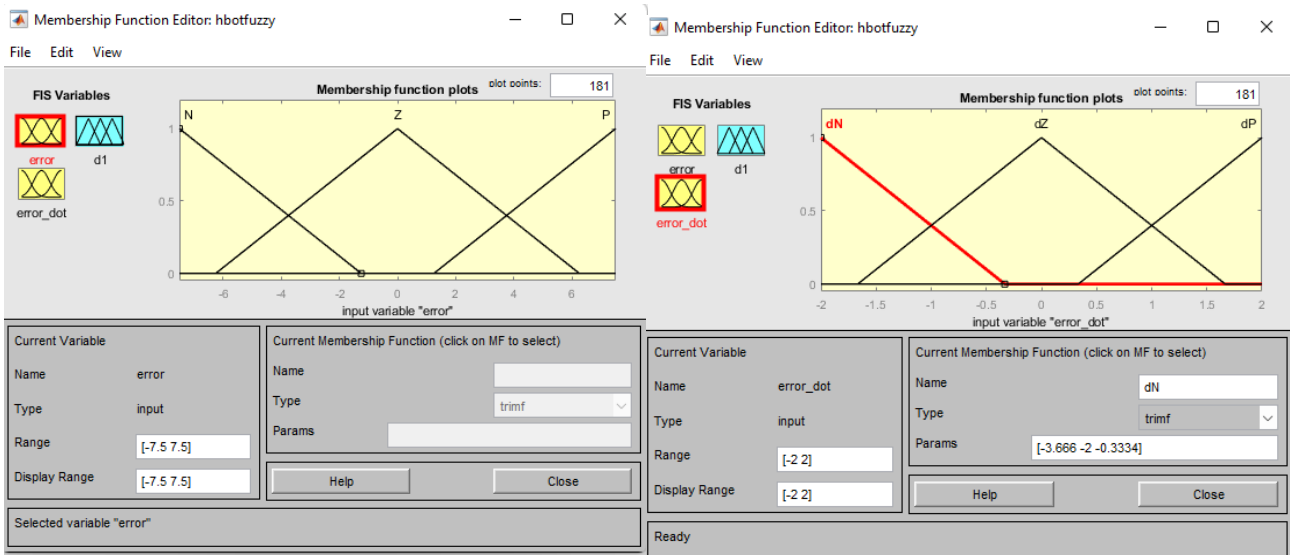

Figure 2.2: error/ errordotMFs

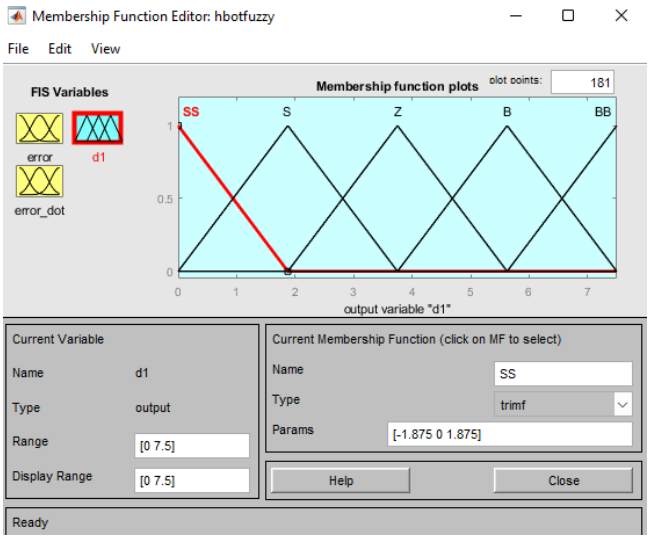

Figure 2.3: output for joint 1

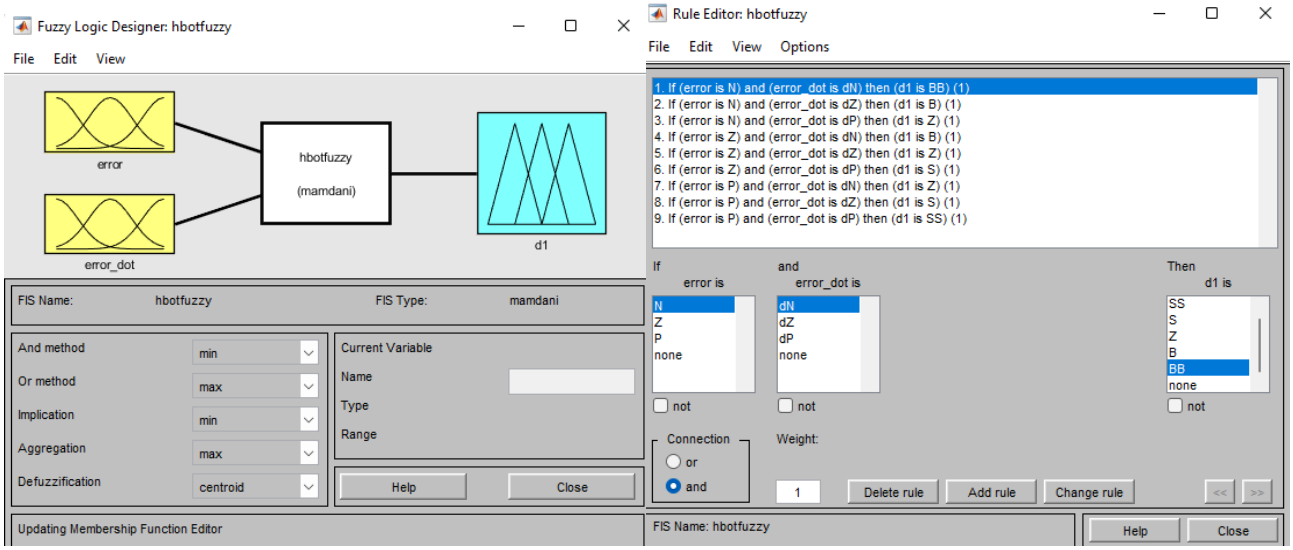

Figure 2.4: fuzzy inference system

Important note

the same procedure was conducted for the other two joints, but when I tried to run the code, there was a problem in matlab solver, I tried to change it and made it fixed step as instructed, yet nothing solved. But this was the furthest step I could reach.

Conclusion

In this project, modelling and simulation of 3 axis industrial cartesian robot was conducted. The model is based on the Denavit-Hartenberg convention and takes into account the effects of gravity, friction, and inertia. The simulation is implemented in MATLAB/Simulink and allowed us to plan a trajectory of the robot based on the robot kinematics. The results of the simulation show that the model is able to accurately predict the motion of the robot. We have also used the simulation to simulate the trajectory of different equations and implemented PID control on the joints in order to get precise trajectory.