Journal of Studies in Science and Engineering

EngiScience Publisher

Research Article

Implementation and Analysis of Motion Control Techniques on an Industrial 7 DOF Robotic Manipulator

Syed Huzaifa Ali [1,*], Ehtisham Ul Hassan[1]

[1] Department of Mechatronics & Control Engineering, Faculty of Mechanical Engineering, University of Engineering and Technology Lahore, Lahore, 54000, Pakistan

*Corresponding Author: Syed Huzaifa Ali, E-mail: shahhuzaifa2119@gmail.com

Article Info

Article History

Received May 02, 2022

Revised Jun 07, 2022

Accepted Jun 11, 2022

Keywords

Robotics

Robot Motion Control

Inverse Kinematics

Trajectory Planning

Path Planning in Matlab

Abstract

With the quest for modernization in robotics motion control of robot manipulators is an area of active research. In motion control, inverse trajectory application is the most adopted solution which is further improved by considering the effect of external forces of the environment. The impact of external force involves dynamic modeling of the system and force control, impedance control, and admittance control techniques are adopted for the solution. This research implements the simulation and more precise visualization of the real-life manipulator under the rules of kinematics and dynamics on a 7 Degree of Freedom manipulator. The objective of this research is to implement motion control techniques: Forward Kinematics, Inverse kinematics, and Trajectory tracking using MATLAB Robotic System Toolbox and analyze the results obtained. These simulations can an effective tools for tuning the motion of robotics by varying the control parameter in simulation and then implementing them in physical hardware.

1. Introduction

Robots have certainly changed the human outlook toward technology, and they have revolutionized the way work is carried out in industries. Due to the presence of robots, the production of different things has increased with the efficiency of almost 100% because they do not get tired like humans, and they are capable of doing the same tasks again and again without any mistakes once they are programmed or trained. Different kinds of robots are used in the industry for doing these tasks depending upon the requirements and the tasks that are performed.

Even after the intensive use of robots, some tasks require human-robot interaction for these kinds of tasks the most important factor which should be considered is the security of humans that work with these robots. So, for these kinds of tasks, collaborative robots are used.

Basic motion planning of robotic manipulators involves Inverses and forward kinematics.IK of a robotic manipulator is based on computing the joint angles of the robotic manipulator when the position and orientation of a manipulator end-effector are given and al kinematic equations are used for conversion from cartesian space to joint space[6]. Whereas in FK kinematic equations are used to compute the position of the end-effector from specified values for the joint parameters [3]. Brief compassion on IK and FK is published in a research article by Lucas Branika [1] according to which in general, inverse kinematics is much more difficult than forward kinematics as it requires much more computation power than forward kinematics. In a task that requires trajectory planning inverse kinematics is mostly used for calculating joint configuration at each waypoint [3].

In the case of a robotic manipulator which has greater than six degrees of freedom, they require orientation of the end effector can be also provided along with the translational vectors [8]. Forward kinematics comes up with a unique solution but in inverse kinematics, there can be multiple joint configurations for a manipulator to achieve a specific joint position and orientation so inverse kinematics may not always provide a unique solution if your arm has more than six joints. In inverse kinematics, a robot manipulator with less than six DOF may have some Cartesian positions and orientations that may not be achievable, and the inverse kinematics at those poses will not be well defined [8].

For this study we have we considered Franka Emika Panda robot motion control implementation in MATLAB.In this research article, we will design the CAD model of Franka Emika Panda, import it on MATLAB [7] then apply different control schemes the behavior of the robot is then studied with the help of graphs and simulations.

2. Literature Review

The reference robotic manipulator which we have selected for this study is Franda Emika Panda is a state of art collaborative robotic arm that has been designed by a company with major research in the field of humanoid robotics known as Franka Emika which is founded by Sami Haddadin [4] [5]. It is 7 degrees of a freedom robot arm and is being utilized in industrial applications where high redundancy and interaction with the external environment are required. This robot provides users direct control and all possibility to program the manipulator for user-defined tasks [6]. In Franka Emika Panda impedance control provides agility and compliance as similar as possible to those of a human arm.

Impedance control is a dynamic approach that enables a robot to interact robot with such an environment where force and position relation are important. It provides an effective way to control the motion and contact force at the same time in impedance control desired performance is specified through a generalized dynamic impedance, namely a complete set of mass-spring-damper equations [11].

Initial research in impedance control was published by Hogan in 1984 [2]when deduced from his observation that the pairs of human muscle in which one muscle contracts at a time and the other relax in correspondence with them and changing the number of forces with the change in contact force from the environment. Hogan stated these muscles are not considered as body generating force, but an impedance adjuster for the human arm to achieve desired motion and elaborate manipulation when contacted with environmental force. Based on the analysis of human manipulations, the concept of impedance control was introduced into robotics, and gradually methods were derived for implementation on robotic manipulator.

In the early 80s, many industrial approaches were developed to focus on controlling the force exerted on the environment by a direct force feedback loop to manipulator state-of-the-art review provided in by D.E. Whitney [14]. Impedance control was initially used for the manipulation of mechanical systems it is the ratio of output effort which is the interaction between the manipulator and the environment to the input force which is the velocity of the robot end effector [4],[14].

Another concept that is involved in impedance control is admittance control which is the opposite of impedance control hence admittance is an effort of the manipulator on the environment as input and velocity of the manipulator as output. In an interaction between the environment and the manipulator if one acts as impedance the other must act as admittance or vice versa. In most the cases, the environment is considered as the environment as it can accept force and the manipulator as impedance.

3. Working Methodology

The work on this project is conducted in the following manner

- Designing CAD Model: A hollow CAD model of the robot is designed in SOLIDWORKS.
- Import On MATLAB: After the design, it is then imported into MATLAB in URDF Format.
- Simulations: When it is imported into MATLAB different schemes like inverse kinematics and trajectory planning.

3.1. Mathematical model of the robot system:

3.1.1. Importing CAD model to MATLAB

After the design of the Cad model in SolidWorks, the next step is to import the CAD model into MATLAB. In MATLAB CAD Files are imported as a rigid body tree which enables MATLAB to apply different control schemes to the Robot and control and manipulate the model according to need. For task, there is an extension available in SolidWorks that exports the CAD model as a URDF File which can be imported into MATLAB as a rigid body tree A URDF file defines the robot joint type joint transforms, and axes and also contains meshes and textures for the robot. URDF files are also compatible with working on Robot Operating systems (ROS). URDF file links and create a tree based on the SW assembly hierarchy and also extract parts of an assembly as single links and allocate them joint properties.

3.1.2. SolidWorks to URDF exporter:

The URDF exporter is an add-in available in SolidWorks that allows exporting SW Parts and Assemblies into a URDF file. In exporting the CAD model of URDF the parent body and the type of joint between the individual inter-connected bodies are specified and then the exporter will create a package that contains a directory for meshes and textures of robots which can be imported into MATLAB to make a rigid body tree model. Before importing the file into MATLAB, the file must be added into the MATLAB directory[7].

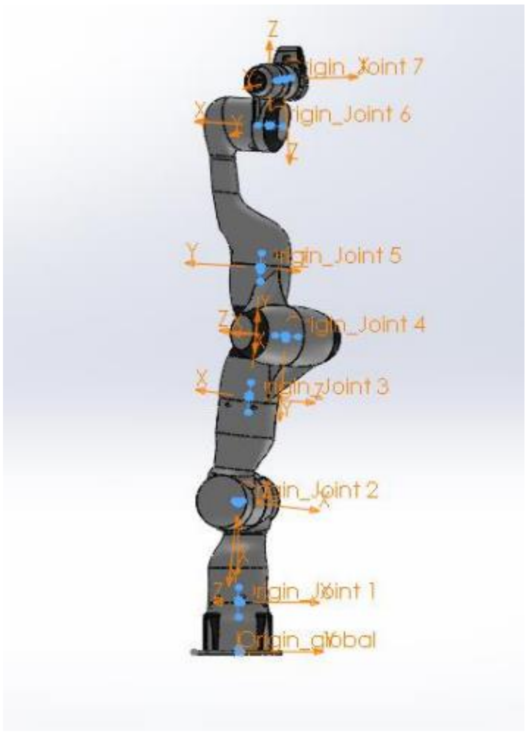
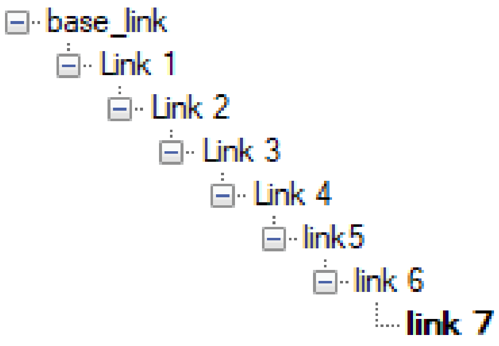


Figure 1. Robot Transformation

The created URDF file is then imported into the MATLAB using command

- In Exporting CAD Model to URDF the First Step is to define the hierarchy which is the child link and the parent links. In the model, this will show how the parts relate to each other concerning the base.-In Figure 1.
- In the second step, the join type between the link is specified and Joint and link names are also specified
- Then the file is exported as a URDF and the exporter automatically assigns frames for each link of the robot for the base frame. The figure below shows the frame assignment for each joint and link.
- The last step is to export the URDF as a rigid body tree in MATLAB[7]



```
base_link
  Link 1
    Link 2
      Link 3
        Link 4
          link5
            link 6
              link 7
```

Figure 2. Joint Linkage

3.2. MATLAB Command:

The URDF file and the meshes are added to the MATLAB path and using “smimport”[7],[12] command. Figure below shows the rigid body tree of the SolidWorks model of Franka Emika Robot imported in to MATLAB from the figure below it shows that the model contains 7 rigid bodies and 7 revolute joints interconnected with respect to base of the robot as a parent link of the model

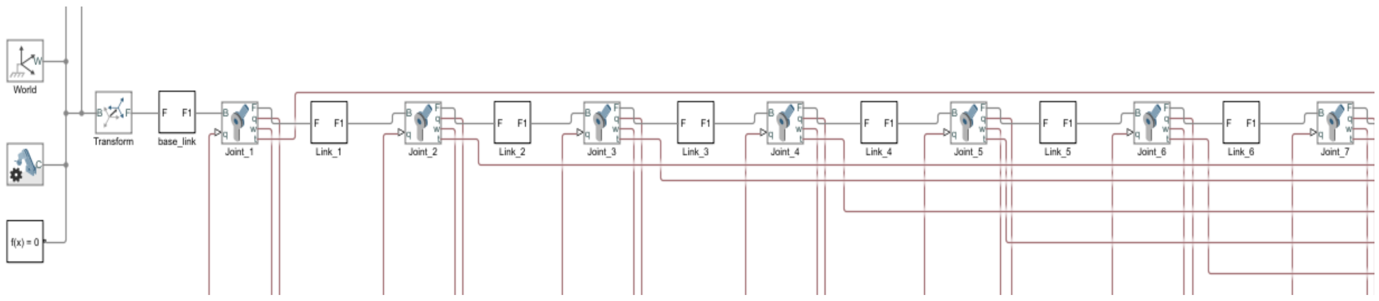


Figure 3. Robot Representation in MATLAB

3.3. Trajectory Planning implementation in MATLAB/Simulink:

3.3.1. Trajectory Generation

In this project, trajectory generation can be done using the following sequence. Start with the Translation vector, x y, and z values as input to the system. Coordinate blocks with sample time equal to 0.1 sec

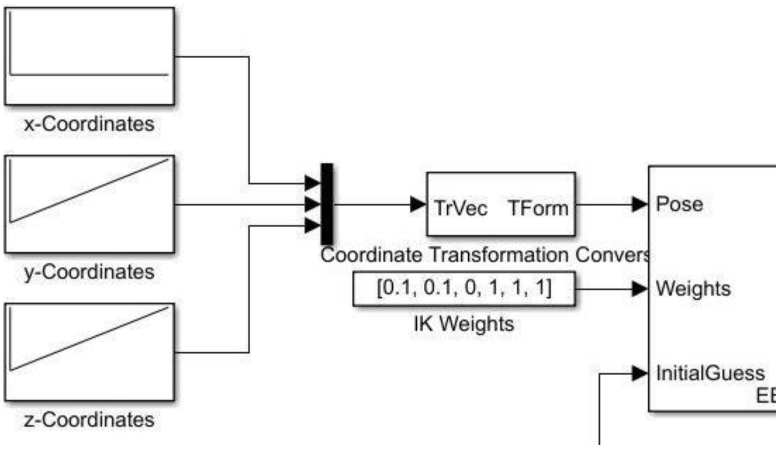
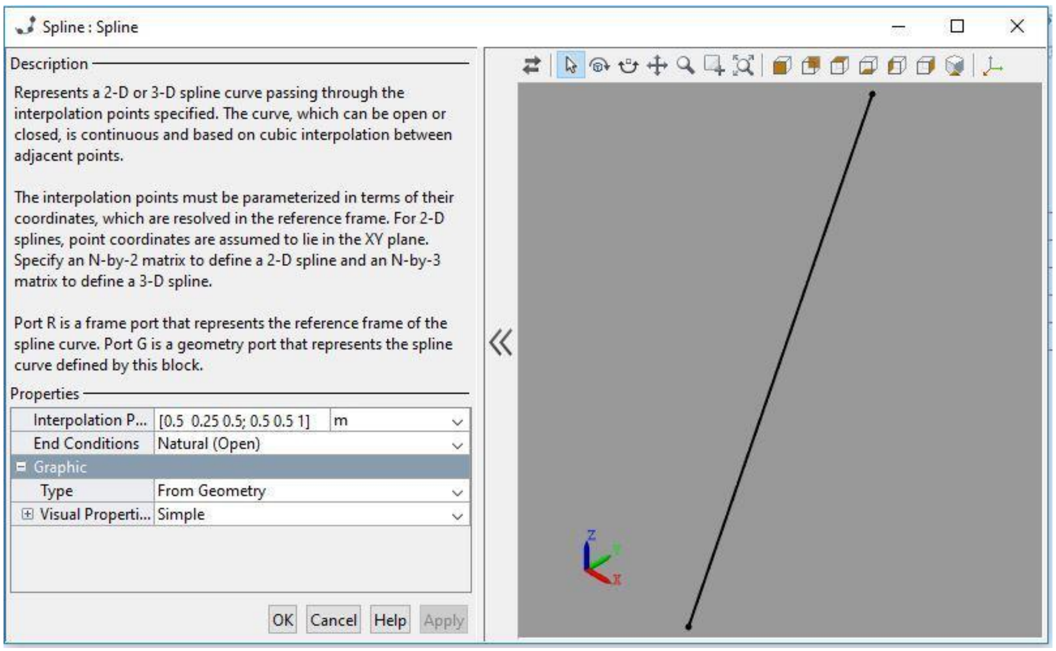


Figure 4. Input Trajectory to robot

3.3.2. Trajectory Plot in Space

To show the trajectory in the world, use the “Spline” block in the Simulink library. Use the same x,y, and z values for the spline. Attach this spline input with the world frame of a subsystem.


Spline : Spline

Description

Represents a 2-D or 3-D spline curve passing through the interpolation points specified. The curve, which can be open or closed, is continuous and based on cubic interpolation between adjacent points.

The interpolation points must be parameterized in terms of their coordinates, which are resolved in the reference frame. For 2-D splines, point coordinates are assumed to lie in the XY plane. Specify an N-by-2 matrix to define a 2-D spline and an N-by-3 matrix to define a 3-D spline.

Port R is a frame port that represents the reference frame of the spline curve. Port G is a geometry port that represents the spline curve defined by this block.

Properties

Interpolation P...	[0.5 0.25 0.5; 0.5 0.5 1]	m
End Conditions	Natural (Open)	
⊟ Graphic		
Type	From Geometry	
⊞ Visual Properti...	Simple	

Figure 5.:Trajectory Plot

- RST(Robotic System Toolbox) block Translation Vector to Homogenous Vector conversion was used to get the Homogenous transform vector.

- Use Config, the output of IK block, for two purposes. First of all, use this Config value as input in the Unit Delay block and then attach the Unit delay output as the Initial Guess of IK block. Secondly, Use the Config as input to Subsystem. Use vector [0.1 0.1 0.1 1 1 1] as Weight values for IK block [1].

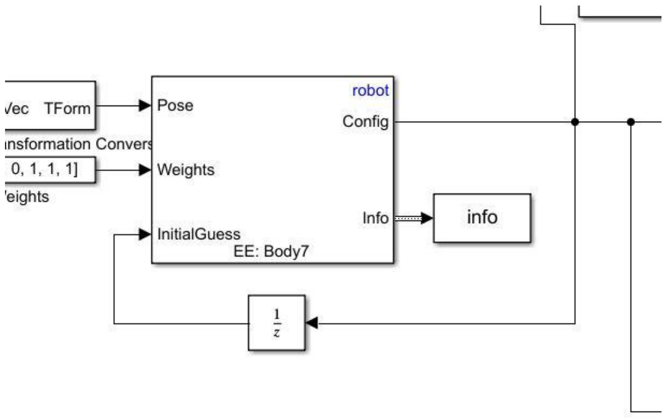


Figure 6. Inverse kinematics Block in MATLAB

3.4. Verification of output results

To make sure whether output values are the same as input values, attach these output values with the input, Config of getting Transform. Then Get Transform will provide the Forward kinematics of the rigid body tree. The output of getting Transform will be a Homogeneous Transformation Matrix. Again, use a Coordinate Transformation Conversion (CTC) with TForm established as input and TrVec as output[13]. Attach HTM with TForm and output of CTC, TrVec with a Demux, having several outputs as 3. Use Display block from Simulink library and attach each display to each output of Demux.

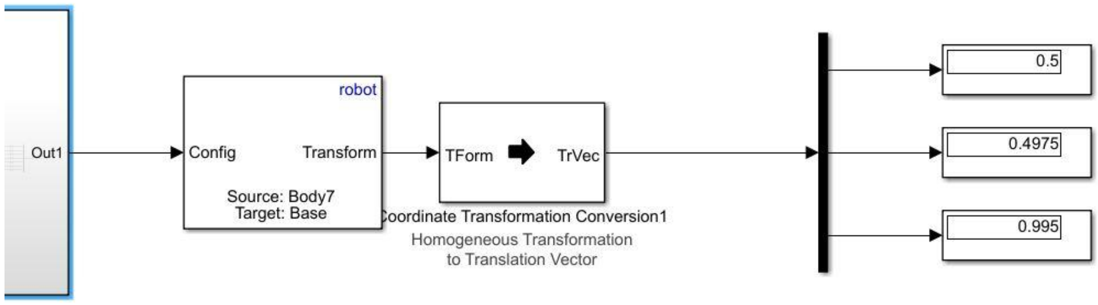


Figure 7. Output Display Block

4. Results for Trajectory

The above method works for almost any URDF manipulator for path following. The Inverse Kinematics block solves for any kind of manipulator of any DOF. The following graphs show two conditions. One shows x, y, and z values of the input trajectory, and the second scope shows x, y, and z values of the output trajectory. It can be observed

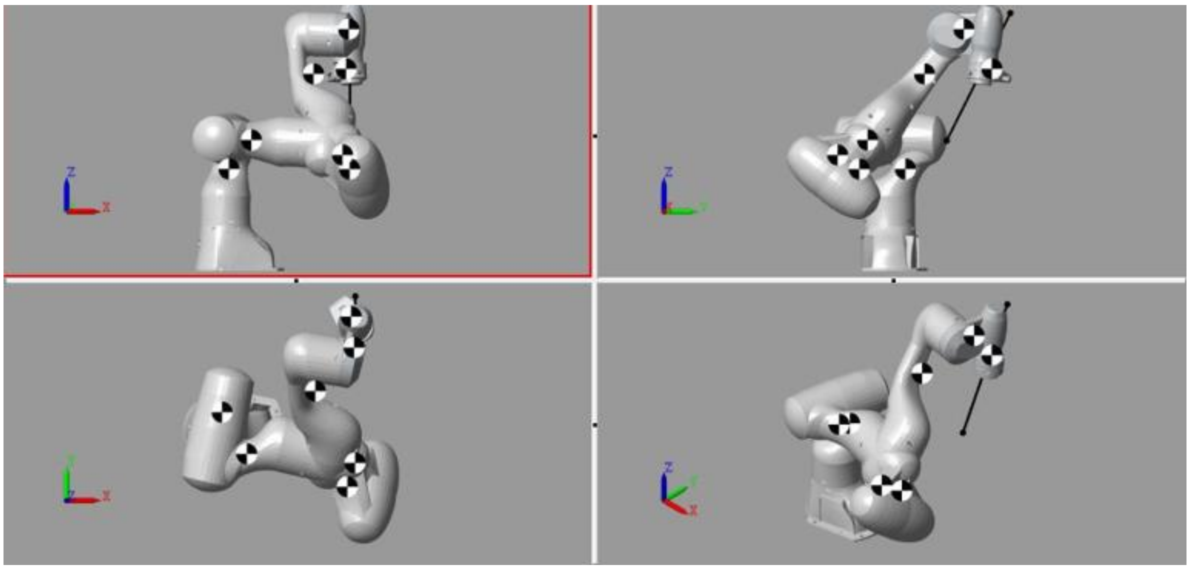


Figure 8. Simulation Output

5. Results and Discussions

The above method works for almost any URDF manipulator for path following. The Inverse Kinematics block solves for any kind of manipulator of any DOF. In the case of trajectory following only numerical values are shown as results that satisfy the output values similarity. Franka Emika Panda Trajectory following in World is shown in the following figure, where the end-effector of the manipulator is following the spline satisfactorily.

Taking a simple example of input trajectory with input values as shown below in table. The input trajectory is given as spline to the manipulator. While the coordinate inputs is given as script file with a sequence of input as

```
% Input trajectory Script file
x1 = 0.5*zeros(1,2)+0.5;
y1 = [0.25, 0.5];
z1 = [0.5, 1];
```

Table 1. Trajectory input values

Coordinates	X	Y	Z
1	0.5	0.25	0.5
2	0.5	0.5	1

The output values at the display are shown in the following figure 41. You can see that the

- Final terminal input point has values [0.5 0.5 1] while
- Final output values are [0.5 0.4975 0.995]

There is only a slight difference between input and output values. This is because of the tolerance values such as Gradient Tolerance- (threshold on a gradient of the cost function), Solution Tolerance- (threshold on pose error), Step Tolerance- (minimum step size), etc. For Inverse Kinematics "More Accurate require more Computational power and time".

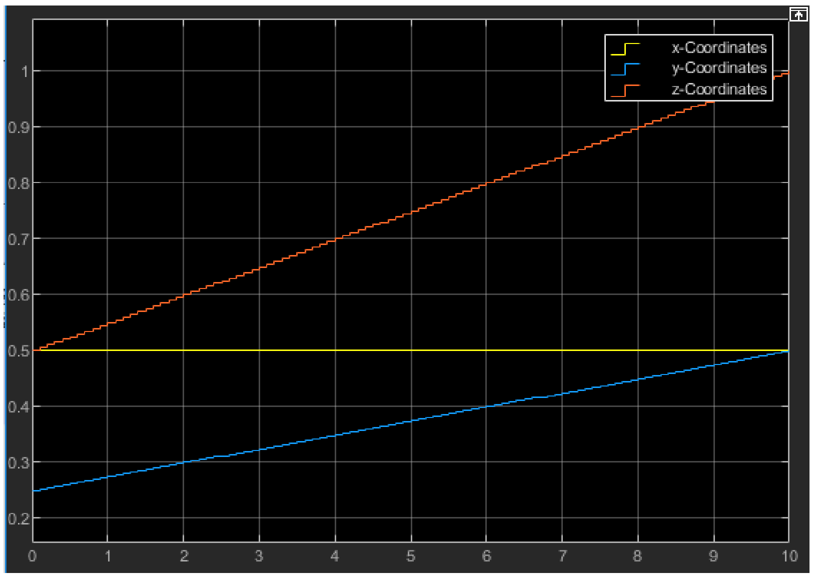


Figure 9. Input Trajectory

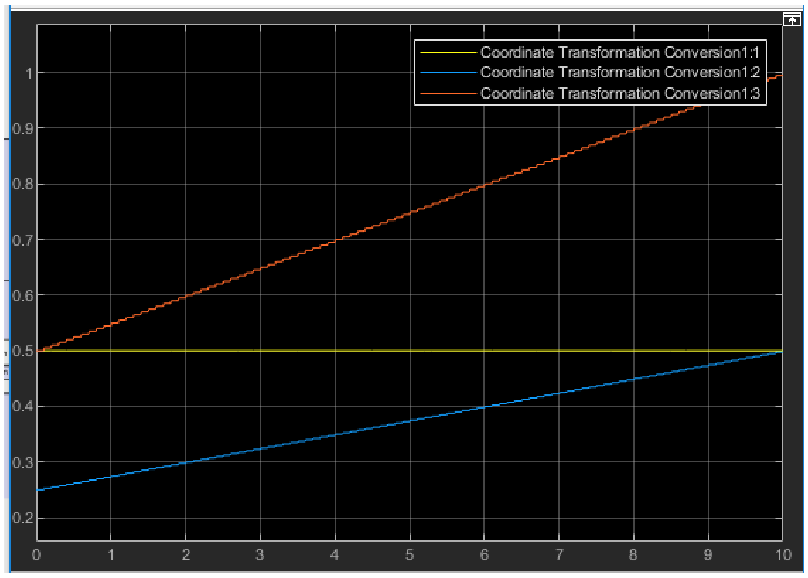


Figure 10. Output Trajectory

5. Conclusions

In this research work, we have applied different control variants like inverse kinematics, trajectory and path planning, to Franka Emika Panda, replicated the cad model and deduce different results. Further, improvements can be made in the domain of obstacle avoidance. So, this model can further be modified to make it capable of avoiding any obstacle coming in its path.[2],[3] which will also allow the manipulator to avoid singularity in its joints. And then impedance Control Behavior can be observed in a simulated environment[9],[10]

6. Recommendations

In this research work, all the implementation of control techniques is simulation-based for the next step the control techniques can be implemented on a physically built robotic manipulator to further analyze the outcomes with the simulated ones in the presence of any external forces. Further, the simulation result can be improved by designing and defining the inertial properties of the robot cad model closer to the physical build robotic system. Also, to better visualize the practical applications of robot manipulators pick and place tasks can be implemented

Declaration of Competing Interest The authors declare that they have no known competing interest

Reference

[1] Barinka, L. and Berka, R., 2002. Inverse kinematics-basic methods. *Czech Technical University*, pp.1-10.

[2] Pertuz, S.A., Llanos, C.H., & Muñoz, D. (2017). Simulation and implementation of impedance control in robotic hand. In *24th ABCM international congress of mechanical engineering* (pp. 1–10). ABCM, Curitiba.

[3] D'Souza, A., Vijayakumar, S. and Schaal, S., 2001, October. Learning inverse kinematics. In *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No. 01CH37180)* , IEEE, 1:298-303.

[4] . Neville, H., 1985. Impedance Control: An Approach to Manipulation: Part I~ III. *Trans. of ASME Journal of Dynamic System, Measurement, and Control*, *107*, p.1.

[5] Hogan, N., 1984, June. Impedance control: An approach to manipulation. In *1984 American control conference* (pp. 304-313). IEEE.

[6] Hernandez-Barragan, J., Lopez-Franco, C., Antonio-Gopar, C., Alanis, A.Y. and Arana-Daniel, N., 2018, November. The inverse kinematics solutions for robot manipulators based on firefly algorithm. In *2018 IEEE Latin American Conference on Computational Intelligence (LA-CCI)* (pp. 1-5). IEEE.

[7] Benotsmane, R., Kovács, G. and Dudás, L., 2019. Economic, social impacts and operation of smart factories in Industry 4.0 focusing on simulation and artificial intelligence of collaborating robots. *Social Sciences*, *8*(5), p.143.

[8] Bertram, D., Kuffner, J., Dillmann, R. and Asfour, T., 2006, May. An integrated approach to inverse kinematics and path planning for redundant manipulators. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006*. (pp. 1874-1879). IEEE.

[9] Song, P., Yu, Y. and Zhang, X., 2017, July. Impedance control of robots: an overview. In *2017 2nd international conference on cybernetics, robotics and control (CRC)* (pp. 51-55). IEEE.

[10] Lu, W.S. and Meng, Q.H., 1991. Impedance control with adaptation for robotic manipulations. *IEEE Transactions on Robotics and Automation*, *7*(3), pp.408-415.

[11] Yoshikawa, T., 2000, April. Force control of robot manipulators. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)* (Vol. 1, pp. 220-226). IEEE.

[12] Faraj, B.M. and Ahmed, F.W., 2019. On the matlab technique by using laplace transform for solving second order ode with initial conditions exactly. *Matrix Science Mathematic*, *3*(2), pp.08-10.

[13] de Gea, J. and Kirchner, F., 2008. Modelling and simulation of robot arm interaction forces using impedance control. *IFAC Proceedings Volumes*, *41*(2), pp.15589-15594.

[14] Whitney, D.E., 1987. Historical perspective and state of the art in robot force control. *The International Journal of Robotics Research*, 6(1), pp.3-14.