Distributed Control of Manipulation: A Real-Time Simulation Framework for EtherCAT Network

Vishwa Nath
Research and Development Estb.(Engineers)
Pune, India
nth.vishwa@gmail.com

Soumya S
Defence Institute of Advance Technology
Pune, India
soumya5.subbu@gmail.com

S E Talole
Research and Development Estb.(Engineers)
Pune, India
setalole@hotmail.com

doi: https://doi.org/10.37285/bsp.SACAD2025.57

***Abstract*—Distributed control architecture has been developed and simulated in real-time to test the performance of controllers developed in MATLAB/Simulink for a highly nonlinear and dynamically coupled 3-link planar rigid robotic manipulator (3-LPRRM). To simulate the performance in real-time, 3 isolated cores are used out of 8 cores from an octa-core Processor. These isolated cores are used to execute the software modules(tasks) like trajectory, controllers, and plant in real-time with cycle rate of $1ms$ and $0.1ms$. The dynamic equations of joint-link pairs are restructured to avoid duplication of calculations. Hence, computational load during the run-time of controllers is reduced. Finally, the performance of PD based controller is evaluated in real-time using TwinCAT3.**

***Index Terms*—Manipulator, EtherCAT, 7 DoF, multi-agent systems (MAS), EtherCAT master, EtherCAT Slave Controller (ESC).**

I. Introduction

A robotic manipulator comprises several links interfaced with joints that allow relatively restricted movement considering the mechanical limits of the connected links. Each joint-link collectively considered as a subsystem that interacts with other like a multi-agent system (MAS). Thus, a manipulator could be considered a distributed system. In the domain of multi-agent systems, system structures can be categorized as centralized, decentralized, or distributed control strategies. In the centralized, a single central controller governs all agents. In decentralized architecture, each agent or subsystem is governed by a distinct controller. In a distributed, each agent (joint-link pair) possesses an individual controller, which communicates with other controllers across a field bus to collaboratively and synchronously accomplish a specified job. This architecture significantly reduces computational load compared to the centralized architecture, as several agent-level controllers distribute workload [1].

The major challenge in distributed control architecture is that each joint need to operate in real-time and at the same time they have to sync with all other joints. EtherCAT is one of the real-time field bus that can operate and sync all the joints/axes in real time. EtherCAT technology get over these limitations. The packets are no longer received, interpreted, and then processed at every device. EtherCAT succeeds in dealing with the nondeterministic nature of TCP/IP of basic Ethernet using a Distributed Clock technique. This assists an EtherCAT network to provide accurate time synchronization between the master and slaves to ensure deterministic operation for coordinated motion application. The DC technique ensure low jitter less than $1\mu s$, between the synchronized master and slaves during operation.

EtherCAT master and EtherCAT based slaves required to implement the EtherCAT communication. EtherCAT master software has its own real-time kernel to perform task execution and synchronization in real-time. An ESC is must to do the ”processing on the fly”. The ESC reads the data address during the frame passing through the joint controllers. In this way frames are delayed only by a few nanoseconds.

In ref [2] three nonlinear dynamic equations have been graphically computed using the phase variable approach in MATLAB. The paper illustrates the graphical solution of the transient response of angular velocity and position for each link during a specified time period. Utilizing simulated torque values, optimal torque has been derived using a GA to appropriately guide the robot.

The paper [3] suggests a distributed cooperative control strategy, in which separate controllers are used to manage each pair of joint links, and these controllers coordinate and share information with one another. The distribution of computational load among joint-level controllers decreases computational lead time relative to a centralized model-based control method, while still accounting for coupling effects. A distributed manipulator control system based on multiple agents is suggested in ref [4], where task execution involves interaction among several software modules referred as agents. The author aims to combine a model and behavior-based methods within a multi-agent control framework. This framework is utilized for manipulator control without employing inverse kinematics. The control system has several functional agents, each endeavoring to enhance the accomplishment of the specified task.

A comparative comparison of three controllers are done for a 3-DOF robotic manipulator using PID, PD and fuzzy logic technique. The efficacy of each control approach was assessed based on the attributes of the transient and steady-state responses [5] [6]. A through study on fractional-order modeling and control methods manipulators is discussed. The primary area of attention includes modeling, fractional-order, control and parameters of controller along with the tuning approaches, objective functions, as well as simulation tools and validation types is shown in detail [7].

In ref [8] modeling and control of a RRR rotary robotic manipulator is investigated. A camera is employed to record the movement of arm and monitor specific angles formed by its segments. A camera model is employed to produce reference angles in simulations. The specified angles are input into the controller. A proportional-derivative (PD) controller is designed and implemented for enhanced trajectory tracking of the manipulator. The outcomes of the suggested control technique exhibit effective trajectory following. A self-regulated fractional-order fuzzy PID controller is developed for the control of a highly nonlinear, complicated, and linked three-link planar rigid robotic manipulator. The Backtracking Search Algorithm is employed to reduce a combination of the IAE signal and the IA change in the control signal as the cost function [9].

A 7-DoF cooperative robot [10], with an open robot control system based on TwinCAT3, serves as the research subject. The robotic control platform was developed with TwinCAT 3 software. An experiment on arc trajectory planning was done utilizing a kinematics algorithm to assess the control system's performance. The robot's actual trajectory precisely matches the predetermined trajectory, with each joint operating effectively, so demonstrating that the open robot control system utilizing TwinCAT 3 fulfills the design specifications. In ref [11] a study of a real-time motion control system is conducted using the EtherCAT protocol and its application to a differential drive robot. A timing analysis is carried out to evaluate the viability of the real-time system, evaluating the periodicity of the cyclic job, jitter, and execution time as assessment metrics between the master and the slaves.

A guidance for selecting the appropriate protocol in automotive networked control systems is presented through a comparative analysis of the performance of EtherNet/IP, EtherCAT, and PROFINET IRT protocols [12]. A centralized motion control system using EtherCAT for real-time robotic control is utilized in high-speed motion networks to provide force-based motion control. The master controller is developed utilizing a real-time kernel and slave network, to achieving hard real-time control capabilities across diverse applications [13].

A unique control system is developed for a 7-degree-of-freedom (DoF) lightweight mechanism, utilizing the EtherCAT bus. The system comprises an arm and a joint slave controller. The arm controller incorporates an intel based PC, providing trajectory planning and movement control in treal-time. The joint controller is made using a FPGA, used for sensor sampling and motor control. The experimental results indicate that control system may attained better control frequency and enhanced scalability [14].

In this paper a real-time simulation for 3-LPRRM is presented by distributing the tasks at different core of the processor using a EtherCAT master stack. Dynamic Equations of the manipulator is written in such a way that computation load is minimized. Finally a real-time simulation is done at different cycle time.

II. Mathematical model of the 3-LPRRM

In this section, dynamic equations of a 3-LPRRM, considered for study. Also, the problem statement and corresponding performance constraints are presented. The manipulator is designed with three links and three revolute joints turning about the z-axis. An ideal 3-LPRRM is shown in Fig. 1. It is termed ideal as all the masses exist at the end point of each link and there is no friction. These masses (kg) are m_1, m_2 and m_3 respectively. The angular positions (radians) and lengths (m) of links are θ_1, θ_2 and θ_3, l_1, l_2 and l_3, respectively. The torques (N-m) needed for the motors to change angular positions of links of manipulator are τ_1, τ_2 and τ_3, respectively. The nominal values of parameters are listed in Table 1. In present study in order to make the control problem less complex lengths of links and mass are taken identical. Using EulerLagrangian techniques, dynamic equation of 3- LPRRM is defined as:

Manipulator.png

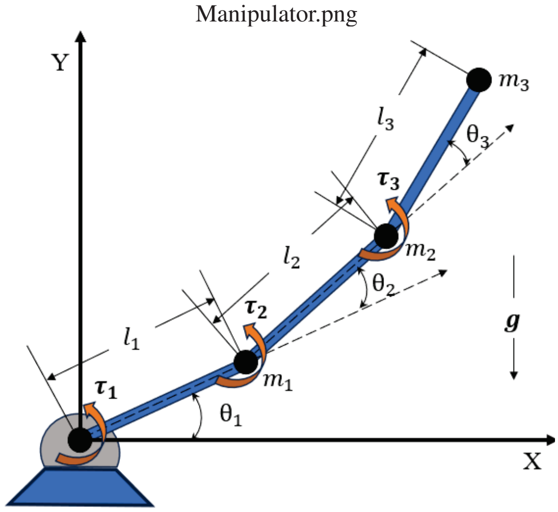

Fig. 1: 3-Link planar rigid robotic manipulator (3-LPRRM)

$$\tau \; = M(\theta)\ddot{\theta} + V(\theta, \dot{\theta}) + G(\theta). \tag{1}$$

Here, τ is a Nx1 joint torque vector, $M(.)$ is the NxN mass matrix, $V(.,.)$ is a Nx1 vector of centripetal/Coriolis accelerations and $G(.)$ is Nx1 vector of gravity terms. Here θ, $\dot{\theta}$ and $\ddot{\theta}$ is angular positions, velocity and acceleration of joints respectively. Where we can refer Nx1 joint torque vector dynamic equations from reference [7]. To reduce the calculation during manipulator operation equations, the above refered torque equations can be written as

$$\tau_1 \; = \tilde{\tau_1} + \tilde{\tau_2} + \tilde{\tau_3}; \tag{2}$$

$$\tau_2 \; = \tilde{\tau_2} + \tilde{\tau_3}; \tag{3}$$

$$\tau_3 \; = \tilde{\tau_3} \tag{4}$$

where

$$\begin{aligned}\tilde{\tau_1} \; =& [(m_1 + m_2 + m_3)l_1^2 + m_3 l_1 l_3 cos(\theta_2 + \theta_3) \\ &+ (m_2 + m_3)l_1 l_2 cos(\theta_2)]\ddot{\theta}_1 + [m_3 l_1 l_3 cos(\theta_2 + \theta_3) \\ &+ (m_2 + m_3)l_1 l_2 cos(\theta_2)]\ddot{\theta}_2 + [m_3 l_1 l_3 cos(\theta_2 + \theta_3)]\ddot{\theta}_3 \\ &- l_1\{m_3 l_3 sin(\theta_2 + \theta_3) + m_2 l_2 sin(\theta_2) \\ &+ m_3 l_2 sin(\theta_2)\}\dot{\theta}_1^2 - l_1\{m_3 l_3 sin(\theta_2 + \theta_3) \\ &+ m_2 l_2 sin(\theta_2) + m_3 l_2 sin(\theta_2)\}\dot{\theta}_2^2 \\ &- m_3 l_1 l_3 sin(\theta_2 + \theta_3)\dot{\theta}_3^2 - 2l_1\{m_3 l_3 sin(\theta_2 + \theta_3) \\ &+ (m_2 + m_3)l_2 sin(\theta_2)\}\dot{\theta}_1\dot{\theta}_2 \\ &- 2m_3 l_1 l_3 sin(\theta_2 + \theta_3)\dot{\theta}_2\dot{\theta}_3 \\ &- 2m_3 l_1 l_3 sin(\theta_2 + \theta_3)\dot{\theta}_1\dot{\theta}_3 \\ &+ (m_1 + m_2 + m_3)g l_1 cos(\theta_1)\end{aligned} \tag{5}$$

$$
\begin{aligned}
\tilde{\tau}_2 = & [m_2 l_2^2 + m_3 l_2^2 + m_2 l_1 l_2 cos(\theta_2) + m_3 l_1 l_2 cos(\theta_2) \\
& + m_3 l_2 l_3 cos(\theta_3)]\ddot{\theta}_1 + [m_2 l_2^2 + m_3 l_2^2 \\
& + m_3 l_2 l_3 cos(\theta_3)]\ddot{\theta}_2 \\
& + m_3 l_2 l_3 cos(\theta_3)\ddot{\theta}_3 + \{m_2 l_1 l_2 sin(\theta_2) \\
& + m_3 l_1 l_2 sin(\theta_2) - m_3 l_2 l_3 sin(\theta_3)\}\dot{\theta}_1^2 \\
& - m_3 l_2 l_3 sin(\theta_3)\dot{\theta}_2^2 - m_3 l_2 l_3 sin(\theta_3)\dot{\theta}_3^2 \\
& - 2m_3 l_2 l_3 sin(\theta_3)\dot{\theta}_1\dot{\theta}_3 - 2m_3 l_2 l_3 sin(\theta_3)\dot{\theta}_2\dot{\theta}_3 \\
& - 2m_3 l_2 l_3 sin(\theta_3)\dot{\theta}_1\dot{\theta}_2 \\
& + (m_2 + m_3) g l_2 cos(\theta_1 + \theta_2)
\end{aligned} \tag{6}
$$

$$
\begin{aligned}
\tilde{\tau}_3 = & [m_3 l_3^2 + m_3 l_1 l_3 cos(\theta_2 + \theta_3) + m_3 l_2 l_3 cos(\theta_3)]\ddot{\theta}_1 \\
& + [m_3 l_3^2 + m_3 l_2 l_3 cos(\theta_3)]\ddot{\theta}_2 + m_3 l_3^2 \ddot{\theta}_3 \\
& + m_3 l_3\{l_1 sin(\theta_2 + \theta_3) + l_2 sin(\theta_3)\}\dot{\theta}_1^2 \\
& + m_3 l_2 l_3 sin(\theta_3)\dot{\theta}_2^2 + 2m_3 l_2 l_3 sin(\theta_3)\dot{\theta}_1\dot{\theta}_2 \\
& + m_3 g l_3 cos(\theta_1 + \theta_2 + \theta_3)
\end{aligned} \tag{7}
$$

III. Problem Statement

A 3-LPRRM, described by Equation 1, is a highly non-linear and coupled MIMO system. Being a highly non-linear plant, it is a very challenging control problem. The control algorithm should handle nonlinearity and generate torque commands within prescribed saturation limits ($\tau_{min} \leq \tau \leq \tau_{max}$) for joint-1, joint -2 and joint-3 respectively, so that the end effector follow respective reference trajectory accurately. A model developed in Simulink for closed-loop control of the manipulator is shown in Fig.2.

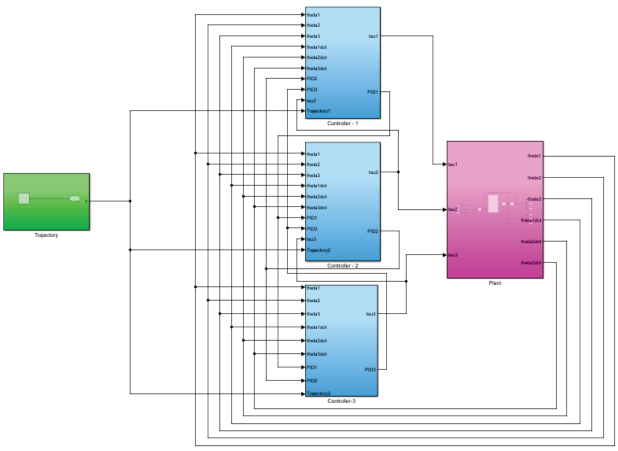

Fig. 2: Simulation model in Simulink/MATLAB

IV. Distribution of model and Control Algorithm

A dedicated controller is used to evaluate the real-time simulation performance. The dynamic model of the manipulator discussed above is distributed in three parts as a trajectory to be followed by the manipulator, controller, and plant as shown in Fig. 3. In implementation individual controllers are not distributed at different cores, as the controllers are tightly coupled and an algebraic loop can be formed during execution.

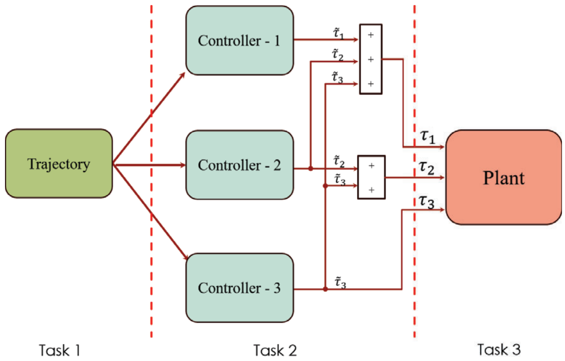

Fig. 3: Distribution of Model as tasks

Beckhoff offers the TwinCAT software, which primarily functions to allocate a specified number of physical cores on user's Windows computer to operate the EtherCAT network. The Windows operating system operates on the remaining cores as determined by the user. Beckhoff also facilitates the execution of Simulink models within the kernel space. A list of necessary software modules are given Below:

- TwinCAT 3.1
- A Beckhoff TE14xx module
- MATLAB/SIMULINK including MATLAB/SIMULINK Coder
- Microsoft Windows Driver Kit 7

To match the implementation with distributed control algorithms, all the modules are distributed at different CPU cores to reduce processing load. Three software modules are used to simulate the model in MATLAB/Simulink. These modules are trajectory generators, controllers for each joint, and plant as shown in Fig.4 .

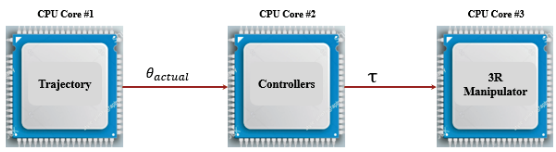

Fig. 4: Model distribution at different cores

As discussed above the simulation model developed in Simulink is distributed as a task over three different cores of CPU using TwinCAT3 and Beckhoff TE14xx module. These tasks are named Trajectory, Controller and Plant 5. These tasks are executed with a cycle time of 1 ms to simulated the model in real-time.

Object	RT-Core	Base Time (ms)	Cycle Time (ms)
Trajectory	Core 5	1 ms	1 ms
Controllers	Core 6	1 ms	1 ms
Plant	Core 7	1 ms	1 ms
I/O Idle Task	Default (5)	1 ms	1 ms

Fig. 5: Task distribution at CPU cores

The overall control inputs to the plant based on the partitioned controller scheme as Equation 8.

$$
\tau = B(q)(\ddot{q}_d + K_D \dot{e}_q + K_P e_q) + V(\theta, \dot{\theta}) + G(\theta) \tag{8}
$$

ISBN: 978-93-47311-06-2

TABLE I: simulation parameters for 3-LPRRM

Parameters	Link-I	Link-II	Link-III
Mass	1 kg	1 kg	1 kg
Link length	0.25 m	0.25 m	0.25 m

Where τ is the $nx1$ vector of joints torques [15].

Where, the closed-loop system is characterized by error equation 9.

$$\ddot{q}_d + K_D\dot{e}_q + K_P e_q = 0 \tag{9}$$

The above error equation is decoupled means the matrices K_P and K_D are diagonal. Taking damping ratio(ζ) = 1 and settling time(ts) = 0.1 s for an over damped system, we can calculate the natural frequency (ω_n) as shown in equation 10.

$$\omega_n \ = 4/(\zeta * t_s) \tag{10}$$

Based on the above parameters we can calculate the PD gain as shown below:

$$K_P = \omega_n * \omega_n; \tag{11}$$
$$K_D = 2 * \zeta * \omega_n;$$

We have used the following parameters for manipulator for simulation in TwinCAT3.

V. Result and Discussion

We now provide the results of simulation tests conducted with TwinCAT3 to illustrate the suggested distributed control method for a 3R planar manipulator. Figures 6, 7, and 8 illustrate the tracking performance of distribted control system in real-timer for applied sinusoidal trajectory with an amplitude of 0.5 and a frequency of 1 rad/s for each joint with cycle-time of 1 ms.

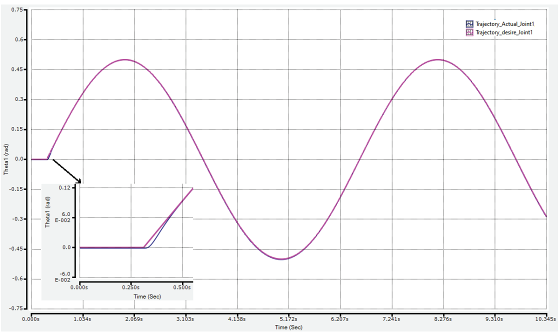

Fig. 6: Trajectory Joint 1

Torque dynamics is illustrated in Fig. 9. The control algorithm generates torque commands within specified saturation limits $(-50\,\mathrm{Nm} \leq \tau \leq 50\,\mathrm{Nm})$ utilize for joint-1, joint-2, and joint-3 to track the specified trajectory.

VI. Conclusion and Future Work

A computationally efficient real-time and synchronized implementation of distributed control algorithms/techniques for 3R robotic manipulators is discussed. The real-time simulation in hardware is a must to demonstrate the effectiveness of the control algorithm. The same algorithm can be tested directly on EtherCAT-based hardware.

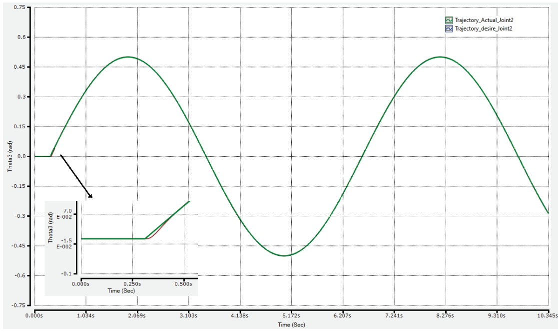

Fig. 7: Trajectory Joint 2

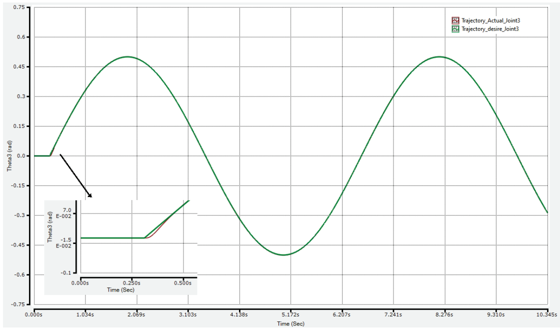

Fig. 8: Trajectory Joint 3

This work needs to be validated on actual hardware to demonstrate the performance in actual field scenarios. The control scheme discussed is PD control, so it does not handle the parameter variation of the plant. It is desired that the control scheme be adaptive and sufficiently robust to track the reference paths in the actual field environment. Now days, the application of redundant manipulators in the field of robotics is growing. Redundant manipulator, with their additional degrees of freedom, provides a unique opportunity to implement and test the various control algorithms and compare their responses. This simulation can be further extended for the redundant manipulator in the future.

TABLE II: Execution time and system latency during real-time simulation $t_s = 1ms$

Parameters	Core-5	Core-6	Core-7
Software Module	Trajectory	Controllers	Plant
Execution Time	2.2 μs	1.6 μs	1.5 μs
Cycle Time	1 ms	1 ms	1 ms
CPU core Load	<1%	<1%	<1%
System Latency	0.5 μs	0.5 μs	0.5 μs

TABLE III: Execution time and system latency during real-time simulation $t_s = 0.1ms$

Parameters	Core-5	Core-6	Core-7
Software Module	Trajectory	Controllers	Plant
Execution Time	1.2 μs	1.5 μs	1.4 μs
Cycle Time	0.1 ms	0.1 ms	0.1 ms
CPU core Load	1%	1%	1%
System Latency	3 μs	3 μs	3 μs

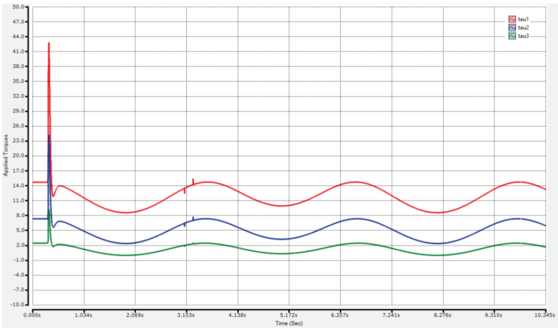

Fig. 9: Applied Torques during tracking

Acknowledgement

To Abhijit D Kamble, for his guidence over EtherCAT and EtherCAT master.

References

[1] I. J. N. R Mittal, *Robotics and Control*. Candor TechSpace, 8th Floor, Tower No 5, Plot No 2, Block B,Sector 62, Noida, Uttar Pradesh - 201309: McGraw Hill Education, 2017.

[2] N. Sahay and S. Chattopadhyay, "Kinematic modelling of three link robot manipulator and joint torque optimization using genetic algorithm in matlab," *IAES International Journal of Robotics and Automation (IJRA)*, vol. 13, p. 160, 06 2024.

[3] S. Soumya and K. R. Guruprasad, "Model-based distributed cooperative control of a robotic manipulator," in *2015 IEEE International WIE Conference on Electrical and Computer Engineering (WIECON-ECE)*, 2015, pp. 313–316.

[4] A. Mori, F. Naya, N. Osato, and T. Kawaoka, "Multiagent-based distributed manipulator control," in *1996 IEEE/SICE/RSJ International Conference on Multisensor Fusion and Integration for Intelligent Systems (Cat. No.96TH8242)*, 1996, pp. 289–296.

[5] U. Kabir, M. F. Hamza, A. Haruna, and G. S. Shehu, "Performance analysis of pid, pd and fuzzy controllers for position control of 3-dof robot manipulator," *ArXiv*, vol. abs/1910.12076, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:204904533

[6] M. S. E. F. G. A. Sherif G. Ahmad, Ahmad S. Elbanna, "Dynamic modelling with a modified pid controller of a three link rigid manipulator," *International Journal of Computer Applications*, vol. 179, no. 34, pp. 37–42, Apr 2018. [Online]. Available: https://ijcaonline.org/archives/volume179/number34/29221-2018916772/

[7] K. Bingi, B. Rajanarayan Prusty, and A. Pal Singh, "A review on fractional-order modelling and control of robotic manipulators," *Fractal and Fractional*, vol. 7, no. 1, 2023. [Online]. Available: https://www.mdpi.com/2504-3110/7/1/77

[8] J. Kim, A. S. Lee, K. Chang, B. Schwarz, S. A. Gadsden, and M. Al-Shabi, "Dynamic modeling and motion control of a three-link robotic manipulator," 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:44143345

[9] V. Kumar, K. Rana, and D. Kler, "Efficient control of a 3-link planar rigid manipulator using self-regulated fractional-order fuzzy pid controller," *Applied Soft Computing*, vol. 82, p. 105531, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1568494619303011

[10] C. Zhang, J. Yuan, H. Zeng, J. Jia, D. Zuo, F. Shen, B. Liu, and S. Niu, "Design of redundant robot control system based on twincat3," in *2022 2nd International Conference on Electrical Engineering and Control Science (IC2ECS)*, 2022, pp. 1239–1243.

[11] R. Delgado, S.-Y. Kim, B.-J. You, and B.-W. Choi, "An ethercat-based real-time motion control system in mobile robot application," in *2016 13th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, 2016, pp. 710–715.

[12] X. Wu and L. Xie, "Performance evaluation of industrial ethernet protocols for networked control application," *Control Engineering Practice*, vol. 84, pp. 208–217, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0967066118303691

[13] I.-K. Jung and S. Lim, "An ethercat based real-time centralized soft robot motion controller," in *2012 International Symposium on Instrumentation & Measurement, Sensor Network and Automation (IMSNA)*, vol. 1, 2012, pp. 117–120.

[14] G. Zhang, F. Ni, Z. Li, and H. Liu, "A control system design for 7-dof light-weight robot based on ethercat bus," in *2018 IEEE International Conference on Mechatronics and Automation (ICMA)*, 2018, pp. 2169–2174.

[15] J. J. Craig, *Introduction to Robotics: Mechanics and Control*. Pearson India Education Services Pvt. Ltd, 5th Floor, Survey No 203/10B, 200 Ft MMRD Road, Zamin, Pallavaram, Chennai - 600044: Pearson Education, 1 April 2022.