2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

Neuro-Adaptive Dynamic Control with Edge-Computing for Collaborative Digital Twin of an Industrial Robotic Manipulator

Sumit Kumar Das, Mohammad Helal Uddin, Dan O. Popa and Sabur Baidya

Abstract— **With the advancement of industrial manufacturing and an increase in introduction of robots in the workspace, the need of safe operation, communication and information sharing is paramount. The work presented here focuses on cyber-physical system integration through Digital Twin (DT) technology. Our novel DT architecture is based on a model-free Neuro-Adaptive controller (NAC), and an edge-computing scheme for scene monitoring. The NAC can account for varying robot dynamics in both real and virtual environments, and allows for the DT system to expand the realm of cyber-physical integration without expensive model tuning. The edge-computing device introduced in our architecture, observes the robot's workspace from a distance with a wider field of view. This wide viewpoint, enhances the detection and mitigation of any obstacles entering the robot's workspace during operation. We experimentally evaluated the performance of our proposed architecture by introducing dynamic obstacles during a pick-and-place task that both the physical robot and its digital twin had to avoid. Results show that the proposed DT architecture successfully integrates the novel controller and edge-computing elements and successfully performs the given navigation task. The results also show that NAC outperforms a PD controller with more than 70% improvement in joint tracking error between the physical and virtual robots. It was observed that the latency experienced while using NAC is about 48% lower than when Proportional-Derivative (PD) controller was operational.**

Index Terms— **Industry 5.0, Digital Twin, Edge-Computing, Neuro-Adaptive Controller (NAC), Smart Manufacturing, Simulation, Robot Operating System (ROS)**

I. INTRODUCTION

Industrial robots are a common element in manufacturing, and their use is on the rise. With advancement in automation and introduction of new sensors, collaborative robots are now capable of sharing the workspace with humans [1] [2]. In order to "work from home", for example during pandemics, end-users should be able to operate robots remotely in a safe manner. There have been many past studies that explored the control architecture to enable tele-robotic operation [3]–[5]. Architectures based on Digital Twins (DT) [6] have been proposed to allow the users to remotely monitor and control such systems for safety-critical operations. Furthermore, the Industry 4.0 road-map highlights the need for such cyber-physical systems in the modern manufacturing environments [7].

Sumit Kumar Das, Mohammad Helal Uddin, Dan O. Popa and Sabur Baidya are with the Louisville Automation and Robotics Research Institute (LARRI), University of Louisville in Kentucky, USA (e-mail: sabur.baidya@louisville.edu)

This work was supported by NSF Grants FOW DRL# 2026584 and EPSCOR OIA#1849213

While DTs have existed for quite a while, with the rise of Industry 4.0 they have seen renewed interest in their use in the context of remote tele-operation [8], [9]. With the progressive strides taken in the field of smart sensors, Internet-of-Things (IoT), communication devices, computing capabilities, the practicality of implementation of near-real time DTs has now become possible. One of the distinguished elements of modern DT systems is the capability to provide a realistic experience for the user to interact with the real-world objects in a virtual world [6], [10].

Recently, DT systems have been proposed for monitoring, optimization, product performance assessment, remote operation of systems and troubleshooting any issues [6], [11]. With a DT system in place, the industry can benefit from the detailed insights on the machine health conditions and needs, thereby fostering competitiveness, productivity and efficiency [12], [13]. One of the challenges to adopt DT systems is to validate and match virtual models with real-world physical implementation [11]. Zhou et. al. argues that a DT system that includes the robot dynamics, can provide a more realistic experience during the remote operation [14]. Most past research in DT systems with robotic manipulators [15]–[25], with various task models ranging from path planning to welding, have concentrated on the kinematic model of the manipulator rather than its dynamic behavior [14].

The objective of our work, is to also include robot dynamics in the virtual environment. In this paper we propose an efficient collaborative DT system for a robot manipulator, which performs pick-and-place tasks that are frequently used in factory floors. During the pick-and-place tasks, the robot can experience a range of dynamic forces, induced by the robot's motion, namely acceleration and deceleration, as well as contact with the environment. A proportional-integral-derivative (PID) controller is a common solution to accomplish trajectory tracking or force control, with the caveat that the PID's parameters need to be tuned before operation. Also, often with changes in dynamic parameters of the manipulator, such as a heavy payload, the PID parameters need to be re-tuned. Additionally, the controller parameters need to be tuned for the physical robot, however, since the same PID parameters may not fit the virtual robot, gains must be separately tuned for both the real and virtual robots.

To avoid the disadvantage of controller tuning in context of DT, our work employs a model-free Neuro-Adaptive Controller (NAC) for consistent, autonomous and synchronous control of both physical and virtual robots. Our previous research [26]–[30] showed that NAC can adapt to the varying dynamics of any robotic system, without the need for pre-

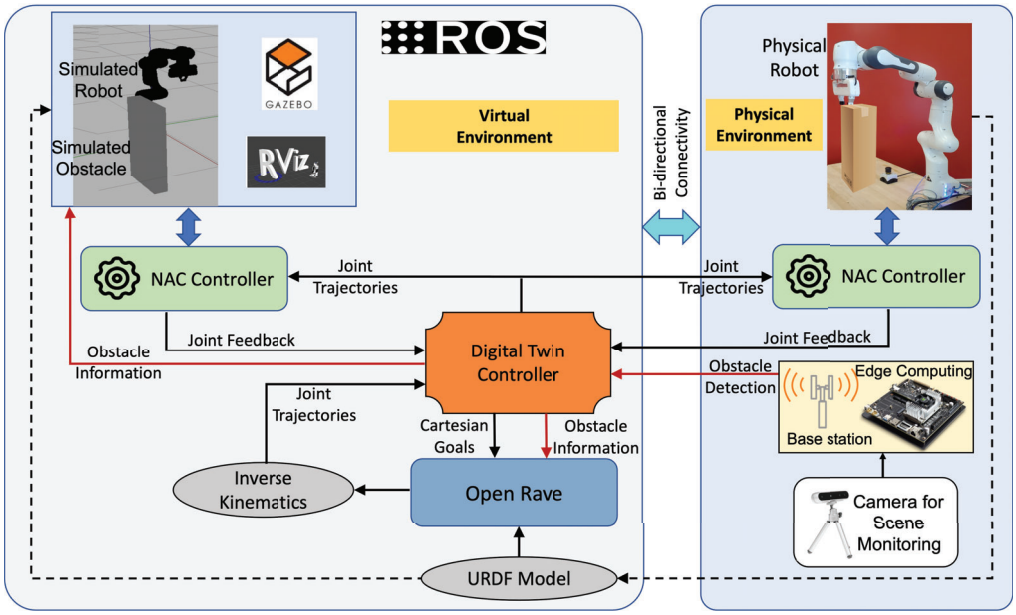


Fig. 1: Architecture Components of Collaborative Digital Twin Framework of a Robotic Manipulator

training, that it works on-line, and that it has robust behavior. Therefore, the use of NAC in a DT system mitigates the need for parameter tuning on both the real robot and the virtual avatar alike.

Another aspect of our proposed work is the use of edge computing [31] to monitor the work area of a robot to provide timely feedback on the target object and dynamic obstacle information. Previous works on edge computing are mainly focused on the optimization and resource allocation for energy efficiency in a mobile platform [32]–[35]. One of the research work presented by Qi et. al., discusses a framework for the integration of edge computing in a smart manufacturing workspace on an individual system level, that includes individual edge configuration to pass on the information to a cloud space for further processing [36], [37]. By contrast, our research focuses on using edge computing at a local level to the robotic site for continuously monitoring the scene of robot operation with low latency.

In this paper we propose a novel DT architecture for industrial robot manipulators. The novel contributions of our research can be enumerated as follows:

- A Model-free neuro-adaptive controller (NAC) in digital twin architecture to account for robot dynamics in both physical and virtual robot, in parallel during their operations.
- Edge-informed obstacle monitoring at the workspace of the robot for uncertainty-aware safe operation.
- Validation and performance evaluation of the DT system with real-world implementation and experiments.
- The performance of NAC was compared with a tuned PD controller in the proposed architecture, which shows a 71.5% improvement in trajectory tracking error, 57% improvement in position tracking error and 48.8% improvement in latency.

This paper is organized as follows: section II describes the proposed DT architecture which includes all the components of physical and virtual counterparts of the DT, Section III presents the sensor-equipped scene monitoring on edge-computing and the dynamic control algorithm with NAC. In section IV the experiment setup and the data collection process is presented. In the end, we conclude with discussions of our results, and future applications of the presented work in section V.

II. Collaborative Digital Twin Framework

The DT architecture is illustrated in Fig. 1, and focuses on safe operation of a robotic manipulator in a manufacturing workspace by real-time collaboration between the physical robot and its digital twin. In an industrial environment, it is important that the robot operates safely while being able to share the workspace with a human user. Additionally, there can be other dynamic uncertainties as well. This is where the edge-computing element comes into action to monitor and provide feedback to the robot controller. This is further discussed in section III-A.

The architecture makes use of Robot Operating System or ROS [38], [39] and Gazebo simulation environment [40] to create the DT. Gazebo utilizes Simulation Description Format (SDF) files to generate simulation models for robots and their surrounding environment. The SDF file contains the dynamics information necessary to simulate the robot's behavior. Previous work on DT have focused on using Unity3D visualizer for the purposes of creating the twin visualization [41], [42]. Some researchers have also presented work using ROS architecture which utilizes Gazebo as simulation platform [43], [44]. In one of the mentioned works, Rassōlkin discusses the use of simulation controllers based on simulation feedback, but have also stated that "*the largest factor in DT controllers is the latency between the DT and the twinned system*" [42]. The proposed architecture

introduces individual controller for the physical and the virtual robot system separately to mitigate any latency as well as account for any deviation in the simulation model's dynamics from the physical robot. Also included is a bi-directional communication channel between the two systems to ensure synchronization. The controller used in the proposed work is further discussed in section III-B.

One of the key aspects of the proposed work is to ensure safe operation of a robotic system through scene monitoring and trajectory re-planning if necessary to avoid any given obstacle. We utilize the OpenRAVE library to plan and generate trajectories for any given task model [45]. It uses a Digital Asset Exchange (DAE) file which is generated from the Unified Robotics Description Format (URDF) file of the robot. The URDF file, like the SDF file contains the kinematic and dynamic information about the robot. Through the DAE file, the OpenRAVE node (as illustrated in Fig. 1) is able to generate the joint angles, through inverse kinematics calculation, that the robot should achieve in order to reach a given point in space. The node outputs these sets of joint angles pertaining to the trajectory path that the robot should follow. These joint angles are then fed to the controller to control the robot in physical and virtual space.

In order to integrate all the components of the DT architecture, a supervisory node called the "digital twin controller" is introduced (Fig. 1). The role of the supervisory node includes creating a bi-directional communication channel between the physical and virtual robot, monitoring any deviation in execution of trajectory in physical and virtual space, properly disseminating the data received from the edge-computing device to initiate any re-planning if necessary, and raise alarm in case any issue is encountered. The supervisory node includes a state-machine based algorithm as well to facilitate the pick-and-place task. The processes handled in each cycle of the node is detailed in Algorithm 1.

III. Edge-assisted Dynamic Adaptive Control

Two major challenges in the framework proposed in the previous sections are - (i) how accurately and timely the dynamics of the robot and also its surrounding environmental uncertainties are processed, and (ii) how accurately, smoothly and synchronously we can control the physical robot and its virtual counterpart. To address the aforementioned challenges, the following subsections discuss the two key components of our proposed work, namely the edge-computing process and the NAC.

A. Edge-Computing based Scene Monitoring

Contrary to cloud computing, where all the data needs to be sent over the communication network, edge computing locally computes sensed information at the robot site, and sends the processed data, which is much smaller in size, to the remote DT. As a result edge computing can help in minimizing communication delays in exchange of sensor information and analytics. Our edge-computing device has a camera which is pointed at the robot's workspace and upon detection of any obstacle which might be in the robot's trajectory path, their location with respect to the robot are passed on through the network for recalculation of arm trajectory. Use of an external camera, further broadens the viewpoint of a robot's workspace, which we have previously explored [46], [47]. Use of such an external system requires an accurate knowledge of the camera's pose with respect to the robot, in order to calculate the pose of any given object/obstacle in the robot's workspace accurately.

Algorithm 1 Algorithm for Supervisory node in the DT Architecture

```
procedure SUPERVISOR_NODE
    Initialize System Components
    User initiates pick-and-place command
    while Not_ShutDown do                ▷ Loop until shutdown
        Get joint states of virtual and real robot q_v, q_r
        if q_v − q_r > ||e_b|| then      ▷ e_b is the error bound
            Raise error signal
            Wait for user intervention.
        end if
        Receive obstacle data from edge-computing device x_obs, y_obs, z_obs
        Receive object data from edge-computing device x_obj, y_obj, z_obj
        Send x_obs, y_obs, z_obs and x_obj, y_obj, z_obj data to OpenRAVE node and Gazebo
        Request OpenRAVE node for Trajectory data q_d
        q_d as input to NAC_simulation and NAC_robot
    end while
end procedure
```

The work presented here dives into the use of such a system where the edge computing device integrated in the proposed algorithm is responsible for continuous monitoring of the workspace. The system consists of an RGB-D camera pointed at the robot's workspace and a computing device connected to the communication network infrastructure. The role of this system is to offload the processing requirements to detect any unforeseen obstacles in the robot's pathway and communicate the same to the supervisory control node.

During the initialization stage, the pose of the robot with respect to the camera, denoted as $\xi_R^C = \begin{pmatrix} p_R^C \\ o_R^C \end{pmatrix}$, is calibrated. We use ξ_R^C to generate the homogeneous transformation matrix, T_{CR}, to be used for calculating the pose of any detected obstacle in the robot's frame of reference. Therefore, when an obstacle is detected, we also get its pose, $\xi_O^C = \begin{pmatrix} p_O^C \\ o_O^C \end{pmatrix}$, with respect to the camera. From the obstacle's pose we can generate the homogeneous transformation matrix as:

$$T_{CO} = \begin{bmatrix} {}^C R_O & p_O^C \\ 0 & 1 \end{bmatrix} \tag{1}$$

where ${}^C R_O \in SO(3)$, represents the rotation matrix derived from the orientation vector o_O^C and is a 3×3 matrix. p_O^C represents the cartesian position of the obstacle w.r.t. the camera. Note that we have T_{CR} from the calibration phase

already. Therefore, the pose calculation of the obstacle with respect to the robot can be calculated as follows:

$$T_{RO} = T_{RC} \cdot T_{CO} \tag{2}$$

where,

$$T_{RC} = T_{CR}^{-1} = \begin{bmatrix} {}^{C}R_{R}^{-1} & -{}^{C}R_{R}^{-1} \cdot p_{R}^{C} \\ 0 & 1 \end{bmatrix}. \tag{3}$$

Fig. 2 illustrates the process of obstacle detection and pose calculations.

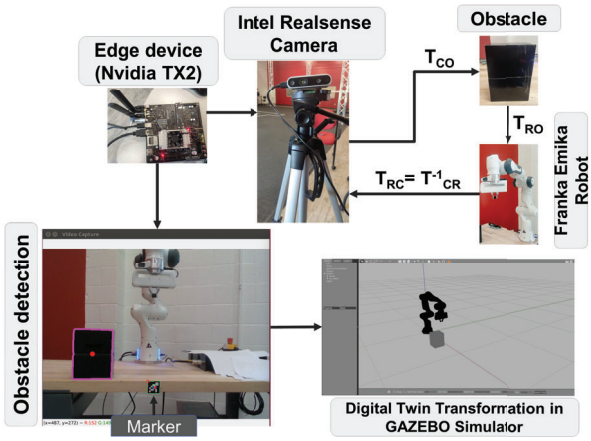


Fig. 2: External Scene Monitoring and Obstacle Detection Process through Edge Computing

In order to accurately detect an obstacle's position w.r.t. the camera, we employ the pin-hole camera model calculations [46], leveraging the fact that we have the depth data from our RGB-D camera. After detection the pixel index, $\{p_{cx}, p_{cy}\}$, referring to the center of the detected obstacle is extracted. Along with it, we also get the depth data, $\{d_{xy}\}$, at that image pixel location. The horizontal and vertical focal length, ρ_h and ρ_v, of the camera, horizontal and vertical field of view, μ_h and μ_v, and the image height and width, i_h and i_w, are constant quantities pertaining to the camera. With this information, we calculate the position of the obstacle as follows:

$$\rho_h = \frac{i_w}{2 \cdot \tan \frac{\mu_h}{2}} \tag{4}$$

$$\rho_v = \frac{i_h}{2 \cdot \tan \frac{\mu_v}{2}} \tag{5}$$

$$x = \frac{p_{cx} \cdot d_{xy}}{\rho_h} \tag{6}$$

$$y = \frac{p_{cy} \cdot d_{xy}}{\rho_v} \tag{7}$$

$$z = d_{xy} \tag{8}$$

where, ρ_h and ρ_v are the horizontal and vertical focal length and $\{x, y, z\}$ are the co-ordinates of the obstacle's position in meters.

B. Neuro-Adaptive Controller (NAC)

The Neuro-Adaptive Controller (NAC) has been explored in our previous works [26]–[28], where its capabilities of controlling a robotic manipulator has been demonstrated. In this section, we go over some of the critical formulations that will be used in the proposed DT system. For detailed proofs and analysis of NAC, we direct the reader to our previous work and the work by Lewis [48].

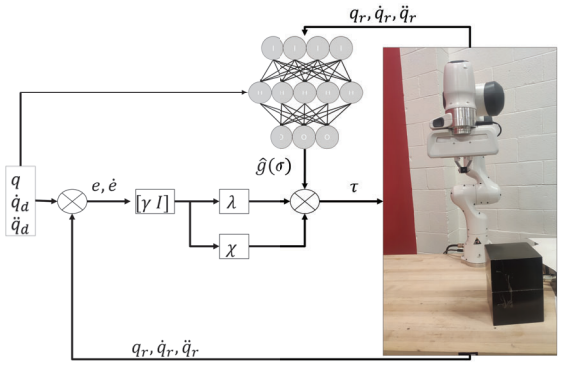


Fig. 3: Neuro-Adaptive Controller Structure for On-Line Dynamic Control of Robot System

The robot dynamic equation of a manipulator can be described as:

$$\Pi(q)\ddot{q} + \Phi(q,\dot{q})\dot{q} + \Upsilon(q) + \tau_d = \tau, \tag{9}$$

where, Π, Φ and Υ represent the mass matrix, Coriolis and gravity term respectively. $q, \dot{q}$ and $\ddot{q}$ are the vectors containing the n-joint angles, velocities and accelerations. τ_d and τ are the disturbance to the system and the control torque respectively. With a given desired joint angle vector q_d and a feedback joint angle vector q_r, the error e becomes $e = q_d - q_r$. Using this, a sliding mode error, s, is defined as $s = \dot{e} + \gamma \cdot e$. Where, γ, is a system parameter which is a symmetric positive definite matrix. Using the error and sliding mode error, the error dynamics of the system from Eq. 9 becomes:

$$\Pi(q)\dot{s} = -\Phi(q,\dot{q})s + g(\sigma) + \tau_d - \tau \tag{10}$$

where the term $g(\sigma)$ is described as:

$$g(\sigma) = \Pi(q)(\ddot{q}_d + \gamma \cdot \dot{e}) + \Phi(q,(\dot{q}))(\dot{q}_d + \gamma \cdot e) + \Upsilon(q) \tag{11}$$

By using Eq. 10 and Eq. 11, a control torque can be defined as below to control the system:

$$\tau = \hat{g}(\sigma) + \lambda \cdot s - \chi \tag{12}$$

where, λ represents the gain matrix and χ is a robustifying term. $\hat{g}(\sigma)$ is an estimation of $g(\sigma)$, which is done by using a neural network. Given an input vector, σ, the output of the neural network can be written as:

$$g(\sigma) = \alpha^T \cdot \Delta(\beta^T \cdot \sigma) + \varepsilon \tag{13}$$

where, α and β are the weights of the neural network and Δ is the activation function of the hidden layer. σ is the input vector which contains $[e^T, \dot{e}^T, q_d, \dot{q}_d, \ddot{q}_d]$. The neural network does not require any pre-training and has unique weight update laws that ensures that the system is stable through a Lyapunov proof. These formulations are detailed in [26], [48], [49]. The NAC structure is illustrated in Fig. 3.

IV. Experiments and Results

To validate the proposed DT architecture proposed in this paper, a testbed with a 7-DOF robotic manipulator was

utilized. The integration of NAC and edge-computing based scene monitoring was tested and validated against a tuned PD controller. This section elaborates the methods employed and metrics used to conduct experiments and analyze results.

A. Experiment Setup

For our experiment setup, we used the Panda robotic manipulator from the Franka Emika®. It is a 7-DOF manipulator arm with 2-fingered gripper capable of lifting a payload of up to 3kg. The SDF file of the robot, available in the public domain, was used to simulate the dynamics of the robot in Gazebo. Robot Operating System (ROS) was used for all programming development purposes. Fig. 4 illustrates the setup for testing purposes.

The Jetson TX2 from the Nvidia®was used as a edge-computing device. It was connected to the DT architecture through wireless communication channel. A Intel®Realsense d435 RGB-D camera that provides the RGB based color images and a depth image as well, was used for the scene monitoring task. The camera was mounted on a tripod and connected to the TX2 computing device to calculate the relative positions of all the obstacles detected in the scene with respect to the robot's frame of reference. OpenCV and ArUco [50] library were used for camera calibration and obstacle pose detection. All the data from the Edge-computing device was forwarded to the DT Control Supervisor node for making dynamic decisions.

The DT Control Supervisor node resides in a remote PC system that controls both the Gazebo twin model as well as the physical robot. It passes on the edge-forwarded obstacle information to the virtual environment for rendering purposes

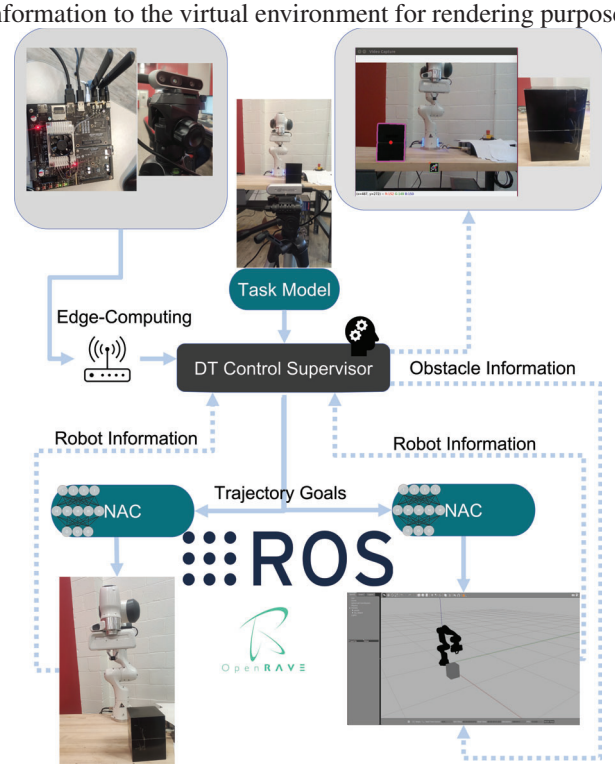


Fig. 4: Experiment setup for DT architecture with 7-DOF robotic manipulator and Edge-computing Device

and keeps track of the robot and its twin's trajectory to ensure no collision occurs. It uses the NAC controller to command both the virtual and real robots to follow a planned trajectory.

The NAC for the DT was designed and deployed as a ROS Control [51] plugin for the robot and its avatar. With this structure, both the physical and the virtual robots have their individual NAC plugin running to control them. A tuned Proportional-Derivative (PD) controller was used as a baseline to compare the NAC's trajectory tracking performance.

TABLE I: MAE for Joint and Position Tracking

Joint Tracking Error (rad)				
	q_0	q_1	q_2	q_3
MAE	0.0035	0.0026	0.0022	0.0026
	q_4	q_5	q_6	
	0.0262	0.0076	0.0097	
Cartesian Tracking Error (m)				
	X	Y	Z	
MAE	0.0028	0.0133	0.0030	

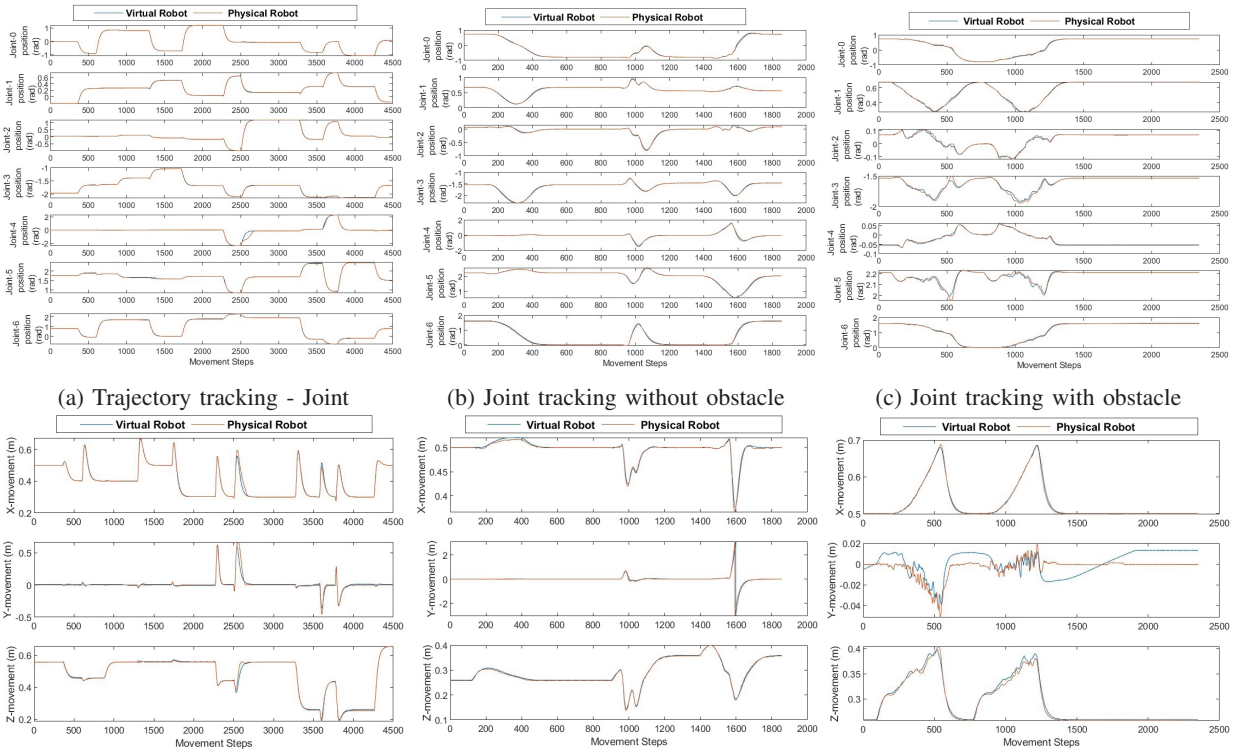
Before conducting the experiments, we calibrated the pose of the camera w.r.t. the robot through the use of ArUco marker and obtained our T_{RC} as described in Eq. 3. After calibration, the DT system was commanded with a trajectory goal and both the physical and virtual robot's joint positions and their end-effector's Cartesian positions were tracked for performance evaluation. The latency between the physical and virtual robot was also evaluated as a performance metric. Obstacles were also introduced in the robot's workspace to evaluate the performance of the DT system.

B. Results

Fig. 5 shows the comprehensive results obtained during the experiments and illustrates the robustness of the proposed architecture. Fig. 5a and 5d show the joint values of the virtual and physical robots during trajectory tracking task. It can be observed that the DT system was able to follow the trajectories while maintaining a high level of synchronization between them.

Fig. 5b, 5e and Fig. 5c, 5f illustrate the difference in performance when no obstacle is encountered vs when an obstacle is introduced to the system respectively. We observe that the motion in Y-axis experienced some disturbances in case of the obstacle. But when we quantify these results we see that the overall Mean Absolute Error (MAE) in tracking between the physical and virtual robot was as high as 0.009 rad for the joints, as well as 0.009 m for Cartesian position. These results are tabulated in Table I. It can be observed that our DT system was able to keep both the virtual and physical robot in sync for a seamless experience. The overall average latency between the virtual and physical system was calculated to be 19.348 ms.

When the DT system was compared with a tuned PD controller, it was observed that there was a combined overall improvement of 71.495% in the joint tracking and 57.06% in position tracking. The results of the experiment are



(a) Trajectory tracking - Joint (b) Joint tracking without obstacle (c) Joint tracking with obstacle

(d) Trajectory tracking - Position (e) Position tracking with obstacle (f) Position tracking without obstacle

Fig. 5: Resulting plots of trajectory and position tracking of the virtual and physical robot under different circumstances

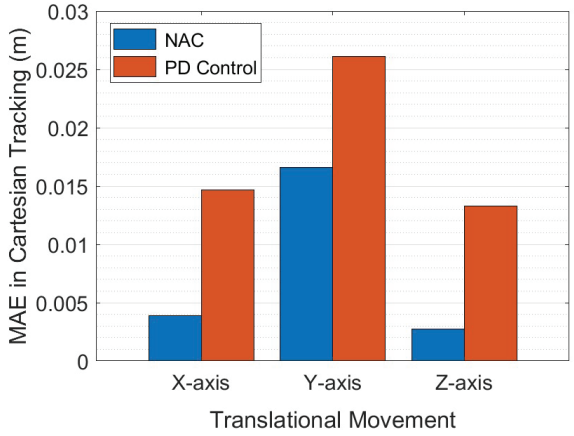


Fig. 6: Position Tracking Error in NAC Vs PD

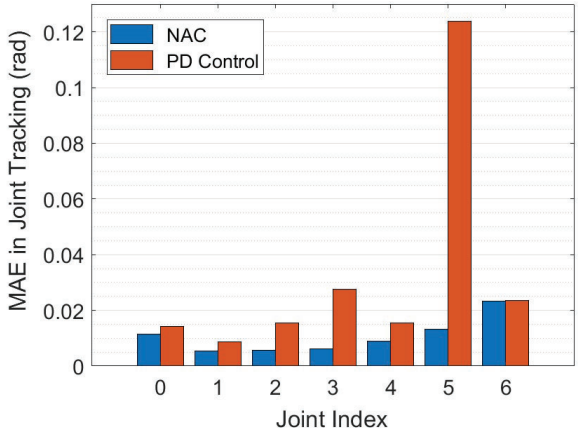


Fig. 7: Joint Tracking Error in NAC Vs PD

illustrated in Fig. 6 and Fig. 7. It was also observed that the average latency between the physical and virtual space was 11.32 ms for NAC and 22.12 ms for PD, which is a 48.82% improvement. It is to be noted that unlike PD controller, NAC required no pre-training or tuning as well.

V. Conclusion and Future Work

In this paper we presented a novel DT architecture which utilizes a Neuro-Adaptive Controller (NAC) and an edge-computing paradigm to ensure proper synchronization between the robot and its virtual model, as well as detect any unplanned obstacles in the path of robot's execution. We illustrate through experiments that the synchronization was effective with minimal error. The experimental results show that NAC outperforms the PD controller in the DT system in both trajectory tracking, in joint space and Cartesian space, and overall data latency.

In the future, the proposed system can be further accentuated with multiple camera system to keep track of the robot's workspace. A multi-robot system and improved trajectory planning algorithms in the future can be used to expand upon the proposed work. Further testing with complex task models could also provide an in-depth view of the performance of the framework presented.

REFERENCES

[1] E. Mariotti, E. Magrini, and A. De Luca, "Admittance control for human-robot interaction using an industrial robot equipped with a f/t sensor," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 6130–6136.

[2] R. Alami, A. Albu-Schäffer, A. Bicchi, R. Bischoff, R. Chatila, A. De Luca, A. De Santis, G. Giralt, J. Guiochet, G. Hirzinger, *et al.*, "Safe and dependable physical human-robot interaction in anthropic domains: State of the art and challenges," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2006, pp. 1–16.

[3] G. Niemeyer and J.-J. Slotine, "Stable adaptive teleoperation," *IEEE Journal of oceanic engineering*, vol. 16, no. 1, pp. 152–162, 1991.

[4] J. Cui, S. Tosunoglu, R. Roberts, C. Moore, and D. W. Repperger, "A review of teleoperation system control," in *Proceedings of the Florida conference on recent advances in robotics*. Florida Atlantic University Boca Raton, FL, 2003, pp. 1–12.

[5] L. Chan, F. Naghdy, and D. Stirling, "Application of adaptive controllers in teleoperation systems: A survey," *IEEE Transactions on Human-Machine Systems*, vol. 44, no. 3, pp. 337–352, 2014.

[6] Q. Qi, F. Tao, T. Hu, N. Anwer, A. Liu, Y. Wei, L. Wang, and A. Nee, "Enabling technologies and tools for digital twin," *Journal of Manufacturing Systems*, vol. 58, pp. 3–21, 2021.

[7] H. Lasi, P. Fettke, H.-G. Kemper, T. Feld, and M. Hoffmann, "Industry 4.0," *Business & information systems engineering*, vol. 6, no. 4, pp. 239–242, 2014.

[8] A. Sharma, E. Kosasih, J. Zhang, A. Brintrup, and A. Calinescu, "Digital twins: State of the art theory and practice, challenges, and open research questions," *Journal of Industrial Information Integration*, p. 100383, 2022.

[9] Z. Jiang, Y. Guo, and Z. Wang, "Digital twin to improve the virtual-real integration of industrial IoT," *Journal of Industrial Information Integration*, p. 100196, 2021.

[10] R. Söderberg, K. Wärmefjord, J. S. Carlson, and L. Lindkvist, "Toward a digital twin for real-time geometry assurance in individualized production," *CIRP annals*, vol. 66, no. 1, pp. 137–140, 2017.

[11] F. Pires, A. Cachada, J. Barbosa, A. P. Moreira, and P. Leitão, "Digital twin in industry 4.0: Technologies, applications and challenges," in *2019 IEEE 17th International Conference on Industrial Informatics (INDIN)*, vol. 1. IEEE, 2019, pp. 721–726.

[12] W. Kritzinger, M. Karner, G. Traar, J. Henjes, and W. Sihn, "Digital twin in manufacturing: A categorical literature review and classification," *IFAC-PapersOnLine*, vol. 51, no. 11, pp. 1016–1022, 2018.

[13] Y. He, J. Guo, and X. Zheng, "From surveillance to digital twin: Challenges and recent advances of signal processing for industrial internet of things," *IEEE Signal Processing Magazine*, vol. 35, no. 5, pp. 120–129, 2018.

[14] Z. Zhou, X. Yang, H. Wang, and X. Zhang, "Digital twin with integrated robot-human/environment interaction dynamics for an industrial mobile manipulator," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 5041–5047.

[15] B. H. Huynh, H. Akhtar, and M. K. Sett, "A universal methodology to create digital twins for serial and parallel manipulators," in *2019 IEEE international conference on systems, man and cybernetics (SMC)*. IEEE, 2019, pp. 3104–3109.

[16] V. Kuts, T. Otto, T. Tähemaa, and Y. Bondarenko, "Digital twin based synchronised control and simulation of the industrial robotic cell using virtual reality," *Journal of Machine Engineering*, vol. 19, 2019.

[17] H. Laaki, Y. Miche, and K. Tammi, "Prototyping a digital twin for real time remote control over mobile networks: Application of remote surgery," *IEEE Access*, vol. 7, pp. 20 325–20 336, 2019.

[18] C. Larsen, "Including a collaborative robot in digital twin manufacturing systems," Master's thesis, Chalmers University Of Technology, Gothenburg, Sweden, 2019.

[19] Y. Cai, Y. Wang, and M. Burnett, "Using augmented reality to build digital twin for reconfigurable additive manufacturing system," *Journal of Manufacturing Systems*, vol. 56, pp. 598–604, 2020.

[20] B. Tipary and G. Erdős, "Generic development methodology for flexible robotic pick-and-place workcells based on digital twin," *Robotics and Computer-Integrated Manufacturing*, vol. 71, p. 102140, 2021.

[21] M. Resman, J. Protner, M. Simic, and N. Herakovic, "A five-step approach to planning data-driven digital twins for discrete manufacturing systems," *Applied Sciences*, vol. 11, no. 8, p. 3639, 2021.

[22] X. Li, B. He, Y. Zhou, and G. Li, "Multisource model-driven digital twin system of robotic assembly," *IEEE Systems Journal*, vol. 15, no. 1, pp. 114–123, 2020.

[23] X. Li, B. He, Z. Wang, Y. Zhou, G. Li, and R. Jiang, "Semantic-enhanced digital twin system for robot–environment interaction monitoring," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–13, 2021.

[24] I. A. Tsokalo, D. Kuss, I. Kharabet, F. H. Fitzek, and M. Reisslein, "Remote robot control with human-in-the-loop over long distances using digital twins," in *2019 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2019, pp. 1–6.

[25] Q. Wang, W. Jiao, P. Wang, and Y. Zhang, "Digital twin for human-robot interactive welding and welder behavior analysis," *IEEE/CAA Journal of Automatica Sinica*, vol. 8, no. 2, pp. 334–343, 2020.

[26] S. Cremer, S. K. Das, I. B. Wijayasinghe, D. O. Popa, and F. L. Lewis, "Model-free online neuroadaptive controller with intent estimation for physical human–robot interaction," *IEEE Transactions on Robotics*, vol. 36, no. 1, pp. 240–253, 2019.

[27] S. K. Das, I. Wijayasighe, M. N. Saadatzi, and D. O. Popa, "Whole body human-robot collision detection using base-sensor neuroadaptive interaction," in *2018 IEEE 14th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2018, pp. 278–283.

[28] S. K. Das, "Adaptive physical human-robot interaction (phri) with a robotic nursing assistant." Ph.D. dissertation, University of Louisville, 2019.

[29] S. K. Das, M. N. Saadatzi, S. Abubakar, and D. O. Popa, "Joint torque estimation using base force-torque sensor to facilitate physical human-robot interaction (phri)," in *2019 IEEE 15th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2019, pp. 1367–1372.

[30] M. N. Saadatzi, S. Abubakar, S. K. Das, M. H. Saadatzi, and D. Popa, "Neuroadaptive controller for physical interaction with an omni-directional mobile nurse assistant robot," in *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, vol. 83990. American Society of Mechanical Engineers, 2020, p. V010T10A055.

[31] T. Qiu, J. Chi, X. Zhou, Z. Ning, M. Atiquzzaman, and D. O. Wu, "Edge computing in industrial internet of things: Architecture, advances and challenges," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 4, pp. 2462–2488, 2020.

[32] R. Dong, C. She, W. Hardjawana, Y. Li, and B. Vucetic, "Deep learning for hybrid 5g services in mobile edge computing systems: Learn from a digital twin," *IEEE Transactions on Wireless Communications*, vol. 18, no. 10, pp. 4692–4707, 2019.

[33] K. Zhang, J. Cao, and Y. Zhang, "Adaptive digital twin and multi-agent deep reinforcement learning for vehicular edge computing and networks," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 2, pp. 1405–1413, 2021.

[34] X. Xu, B. Shen, S. Ding, G. Srivastava, M. Bilal, M. R. Khosravi, V. G. Menon, M. A. Jan, and M. Wang, "Service offloading with deep q-network for digital twinning-empowered internet of vehicles in edge computing," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 2, pp. 1414–1423, 2020.

[35] T. Do-Duy, D. Van Huynh, O. A. Dobre, B. Canberk, and T. Q. Duong, "Digital twin-aided intelligent offloading with edge selection in mobile edge computing," *IEEE Wireless Communications Letters*, vol. 11, no. 4, pp. 806–810, 2022.

[36] Q. Qi, D. Zhao, T. W. Liao, and F. Tao, "Modeling of cyber-physical systems and digital twin based on edge computing, fog computing and cloud computing towards smart manufacturing," in *International Manufacturing Science and Engineering Conference*, vol. 51357. American Society of Mechanical Engineers, 2018, p. V001T05A018.

[37] Q. Qi and F. Tao, "A smart manufacturing service system based on edge computing, fog computing, and cloud computing," *IEEE Access*, vol. 7, pp. 86 769–86 777, 2019.

[38] A. Koubâa *et al.*, *Robot Operating System (ROS)*. Springer, 2017, vol. 1.

[39] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng, *et al.*, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, no. 3.2. Kobe, Japan, 2009, p. 5.

[40] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator."

[41] C.-J. Liang, W. McGee, C. Menassa, and V. Kamat, "Bi-directional communication bridge for state synchronization between digital twin simulations and physical construction robots," in *Proceedings of the International Symposium on Automation and Robotics in Construction (IAARC)*, 2020.

[42] A. Rassõlkin, V. Rjabtšikov, V. Kuts, T. Vaimann, A. Kallaste, B. Asad, and A. Partyshev, "Interface development for digital twin of an electric motor based on empirical performance model," *IEEE Access*, vol. 10, pp. 15 635–15 643, 2022.

[43] N. Kousi, C. Gkournelos, S. Aivaliotis, C. Giannoulis, G. Michalos, and S. Makris, "Digital twin for adaptation of robots' behavior in flexible robotic assembly lines," *Procedia manufacturing*, vol. 28, pp. 121–126, 2019.

[44] X. Wang, C.-J. Liang, C. Menassa, and V. Kamat, "Real-time process-level digital twin for collaborative human-robot construction work," in *Proceedings of the International Symposium on Automation and Robotics in Construction (IAARC)*, 2020.

[45] R. Diankov and J. Kuffner, "Openrave: A planning architecture for autonomous robotics," *Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-08-34*, vol. 79, 2008.

[46] S. K. Das, "Realistic interaction with social robots via facial expressions and neck-eye coordination," Master's thesis, The University of Texas at Arlington, 2015.

[47] A. Habib, S. K. Das, I.-C. Bogdan, D. Hanson, and D. O. Popa, "Learning human-like facial expressions for android phillip k. dick," in *2014 IEEE International Conference on Automation Science and Engineering (CASE)*. IEEE, 2014, pp. 1159–1165.

[48] F. L. Lewis, "Neural network control of robot manipulators," *IEEE Expert*, vol. 11, no. 3, pp. 64–75, 1996.

[49] I. Ranatunga, S. Cremer, F. L. Lewis, and D. O. Popa, "Neuroadaptive control for safe robots in human environments: A case study," in *2015 IEEE International Conference on Automation Science and Engineering (CASE)*. IEEE, 2015, pp. 322–327.

[50] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, 2014.

[51] S. Chitta, E. Marder-Eppstein, W. Meeussen, V. Pradeep, A. R. Tsouroukdissian, J. Bohren, D. Coleman, B. Magyar, G. Raiola, M. Lüdtke, *et al.*, "ros_control: A generic and simple control framework for ros," *The Journal of Open Source Software*, vol. 2, no. 20, pp. 456–456, 2017.