International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 12 Issue: 03 | Mar 2025 www.irjet.net p-ISSN: 2395-0072

A Mobile Based Controller for Robot Arm

Ashlin Joe J[1], Praveen T[2], Anusha D K[3] and J Sivaguru[4]

[1]*Student, Mechatronics Engineering, Kumaraguru College of Technology, Coimbatore, India*
[2] *Student, Mechatronics Engineering, Kumaraguru College of Technology, Coimbatore, India*
[3] *Student, Mechatronics Engineering, Kumaraguru College of Technology, Coimbatore, India*
[4] *Professor, Mechatronics Engineering, Kumaraguru College of Technology, Coimbatore, India*

Abstract - *The role of Industrial robots is vital in Modern Manufacturing. So, operating it efficiently is also necessary But The traditional Teach Pendent used are bulky, costly, inflexible, brand-specific, lack portability, limiting compatibility across different manufacturers and creating inefficiencies in multi-robot setups. It also reduces the efficiency of the robots as well as reduces the production volume. This project proposes a Mobile Based Control solution to replace the traditional teach pendent with the smartphones or Tablets. So, this project aims to create a mobile based application to control an industrial robot arm which enables portability and accessibility (allowing robot operators to control the robot from anywhere within the factory), scalability, cost reduction and improving efficiency in multi-robot industrial setups.*

Keywords: Industrial robots, teach pendant, mobile control, smartphone application, industrial manipulator, secure connectivity, robot automation.

1. INTRODUCTION

Due to the advancement of automation in industries the efficiency, precision, less lead time, high production volume, high accuracy and cost effectiveness had enhanced significantly leading to better manufacturing. So, nowadays, the industrial robot plays an important role in many applications like assembly, welding, material handling, and quality inspection etc. [1]. Even though it has so many benefits most of the industries rely on teach pendent for programming and controlling the robot. While teach pendants have been the industry standard for decades, they have some limitations including bulkiness, high cost, and brand-specific compatibility, which limits it from integration across different robotic systems. These limitations not only increase operational expenses but also reduce overall productivity and flexibility in multi-robot environments [2].

To solve these challenges mobile-based control systems are evolving as a feasible replacement for conventional teach pendants. Using smartphones and tablets, mobile apps offers an easy-to-use interface for programming and controlling industrial robots remotely. This is a cost-effective and efficient choice since the portable and wireless features improve accessibility and remove the need of brand-specific teach pendent [3]. Mobile-based control systems allow operators monitor and modify robotic performance from anywhere instead of physically interacting with the robot as required in use with teach pendants, so improving real-time process and lowering downtime [4].

Apart from cost savings mobile technology integration into industrial robotics provides many benefits. Scalability is one of the main advantages since mobile-based systems are easily customized to manage multiple robots inside a industry improving efficiency and coordination [5]. Wireless communication protocols like Wi-Fi and Bluetooth helps to increase the systems capabilities by allowing quick and dependable data transfer between the mobile user interface and robotic hardware. This reduces latency, increases response times, and makes sure the command execution is done, therefore industrial automation can be done with this system [6].

This study supports the fundamental principles of Industry 4.0, which places a high value on advanced automation of manufacturing processes and digital transformation. In order to improve industrial productivity and efficiency, Industry 4.0 encourages the use of cyber physical systems the Internet of Things (IoT) and real-time data exchange [7]. Industries can increase flexibility, operational efficiency and cost savings by incorporating mobile-based robotic control which makes robotic automation more affordable and flexible for industries of all kinds.

This study focuses on the development and implementation of a mobile-based application for controlling an industrial robotic arm and its effectiveness in real-world manufacturing environments. The goal of this study is to show how mobile technology can enhance efficiency, operational flexibility, and robotic control leading to automation. The suggested solution aims to overcome the drawbacks of traditional teach pendants and also enhances flexibility, user friendly, multi robot control, portability and easy robotic controls which are developed using unity [8]. By demonstrating the advantages of mobile-based robotic control solution to improve automation and workflow efficiency in the industries. The results of this research will contribute to ongoing efforts in smart manufacturing [9].

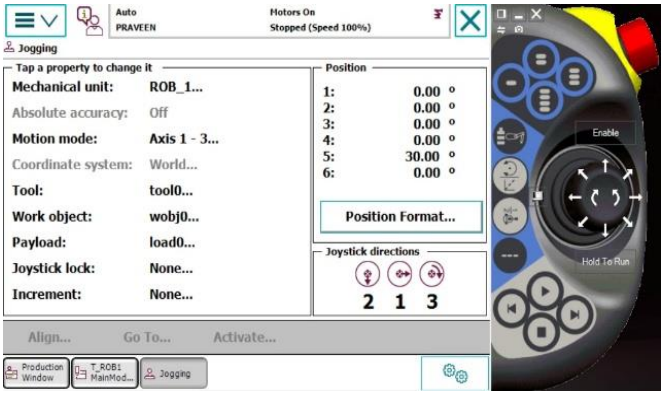

Fig -1: Traditional teach pendent [Source. ABB RobotStudio]

2. INITIAL STUDIES

Through a series of experiments, mobile applications and robotic hardware were integrated in order to develop and validate a mobile-based control system for industrial robots. It began by creating a 3D model in SolidWorks by considering mechanical limitations, degrees of freedom and then servo motor was integrated. Motion analysis and torque analysis were done to improve the design for practical application. with the help of unity we were able to create an user interface with real time rendering capabilities which enables to control the robot through a mobile wit buttons, sliders and real time feedback. After the completion of 3D model, a physical model was 3D printed in a Creality K1 3D printer with infill of 60 percentage and using adaptive cubic infill pattern. Then these parts were assembled by connecting all the links, servo motor and ESP module which will act as a bridge for communication between the User interface and physical model. Then it was programmed to accept movement commands and the connected links change the angle as per the command with the help of servo motors. In order to evaluate system performance, a number of tests were conducted, including testing latency in UI input and motor response, connectivity stability, and motion accuracy to ensure proper execution of commands. From such testing, iterative changes were made to optimize the UI layout, improve firmware for better control of motors, and improve mechanical design for reduced roughness, eventually resulting in an immensely efficient and responsive mobile-based control system for industrial robots.

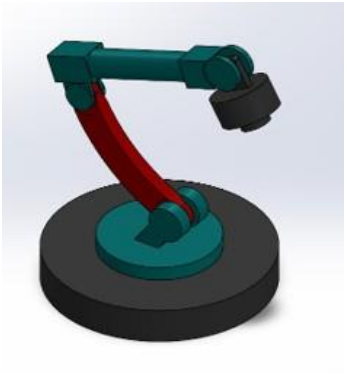

Fig -2: Initial CAD Model

3. INDUSTRIAL ROBOT SIMULATION IN ABB ROBOTSTUDIO

After validating real-time control using servo motors and an ESP module the next phase involved simulating and testing an industrial robot in ABB RobotStudio. This simulation helped to analyse the robot motion, programming and communication between the mobile user interface and robotic software before implementing in the real world robot.

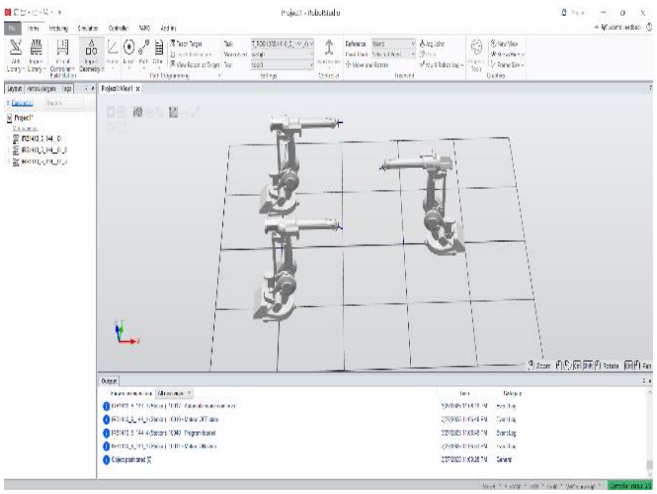

Fig -3: ABB Robot Studio Interface for Multi Robot setup [Source. ABB RobotStudio]

The primary goal of this stage was to replace the traditional teach pendant with a mobile based control system for added flexibility, cost reduction, and convenience. RobotStudio integration with the Unity-based UI enabled real-time execution of commands, facilitating accurate movement control and flawless interaction between the user interface and robot in RobotStudio.

3.1. System Architecture

The system architecture is comprised of several interrelated components:

- Mobile UI (User Interface):

The mobile UI, built using Unity, is an easy-to-use interface by which the user can operate the industrial robot. It comprises interactive joysticks, sliders, and buttons to provide movement commands.

- Processing Unit (PC):

PC is a middleware that accepts commands from the mobile UI, processes information, and gives instructions to ABB RobotStudio. Among others, it smoothens out data transformation, networking, and command synchronization.

- ABB RobotStudio (Virtual Environment):

This platform is used for the verification and virtual simulation of the project. Thus, it simulates the real-time movement of robots before the commands are issued to the industrial robot on site. Workspace setups, virtual robot motion execution, and the definition of tool centre points (TCPs) were carried out in RobotStudio.

- Industrial Robot and Robot Controller:

An industrial robot will be fed with commands processed from ABB RobotStudio through the robot controller. The robot controller executes sequences of movement and sends feedback data to the system, enabling accurate execution of commands from RobotStudio.

3.2. With Installation of Industrial Robot in ABB RobotStudio

3.2.1. Choose and Configure Your Robot:

The appropriate model of industrial robot (ABB IRB 1410 or IRB 2600, for example) was chosen in ABB RobotStudio as per project requirements. The workspace, joint range, reference frames, and TCP parameters were defined so as to mimic real-world operating constraints.

3.2.2. Robot Motion Programming:

To command the robot motions, pre-programmed paths or motion trajectories were defined in RAPID. These trajectories gave users optimum control over robot joint positions and end-effector positions.

3.2.3. Tried Obstacles and Collision Detection:

Collisions were simulated on a virtual scene with amenities to contain the virtual obstacles for simulating detection, accuracy of movement, and workplace limitations. This was to allow the robot to optimize the path and be in a safe manner.

3.3. Robot Control Installation on Unity

3.3.1. Robot Control UI Elements:

An intuitive interface has been designed with Unity; features include the following: Buttons and sliders (For action with precise control of joint movement), Joystick inputs (For freeform movement), Live position feedback display (To monitor the status of the robot in real-time)

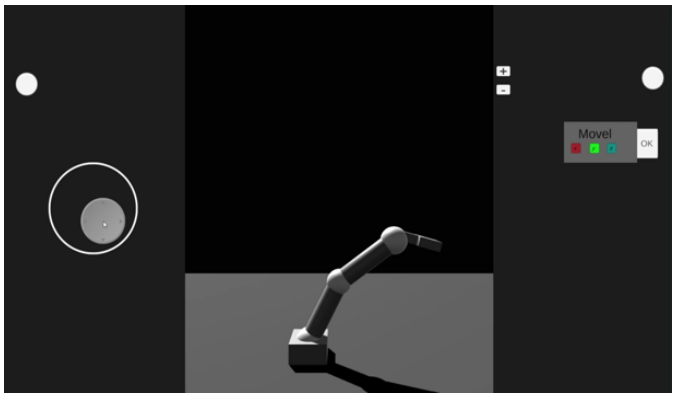

Fig -4: Initial Unity UI

3.3.2. Command Processing with Unity:

Every single piece of user input was received by the Unity application and was processed into the appropriately formatted command packets via a C# script. Command packets contain information like: Type of movement (Linear/Joints), Destination positions (Joint angles / Cartesian coordinates), Speed and acceleration values.

3.3.3. Transmission of Data from Mobile UI to PC:

To provide real-time connectivity, the mobile UI sent control packets to the PC middleware through a wireless communication protocol (Wi-Fi or Bluetooth).

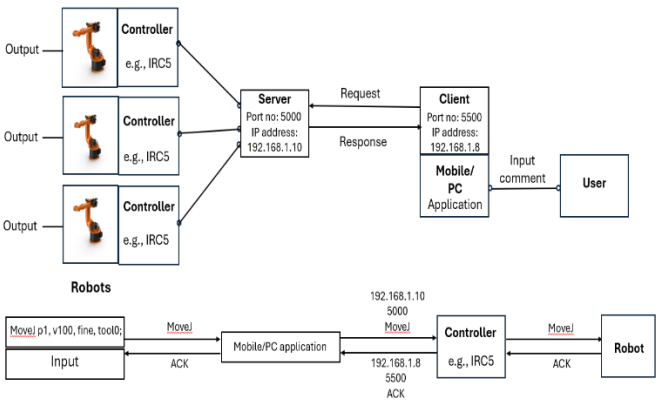

Fig -5: Data transmission Flow chart

3.4. Communication Between PC and Robot Controller

3.4.1. Communication Protocols:

TCP/IP protocol was utilized to provide interruption-free data transmission between the PC and robot controller. This facilitated a low-latency and trustworthy connection for real-time control.

3.4.2. ABB Robot Communication Server Setup:

ABB RobotStudio's Robot control system was configured to accept external control signals. They were decoded using a RAPID script that converted received instructions to associated robot movements.

3.4.3. Supply of Real-Time Processing of Data:

To provide synchrony, latency-reduction techniques were utilized.

3.5. Data Transfer Protocol Implementation

3.5.1. Data Packet Format:

The formatted data packets had: Header data (Command type, Sender ID), Robot joint angles or Cartesian coordinates, Speed, acceleration, and execution time parameters.

3.5.2. Parsing and Running Commands in RobotStudio:

The RAPID script captured and translated control messages from the PC. Depending upon the command mode, the robot performed motions including: MoveL (Linear movement). MoveJ (Joint-directed movement). Emergency stop and pause operations.

3.6. Real-Time Feedback System

3.6.1. Tracing Robot Motion Data:

A feedback loop was added to record robot joint positions, TCP coordinates, and movement status in real time. The information was obtained from ABB RobotStudio and the controller of the industrial robot.

3.6.2. Synchronizing the 3D Model in Unity:

Movement data was returned to the mobile UI and updated a real-time 3D model of the robot in Unity accordingly. Operators would thus be able to visually observe real robot movement and compare it with expected motion.

Fig -6: Mobile Based Controller with Feedback system in 3D Model

3.6.3. Error Detection and Correction:

If any mismatch was felt in commanded vs. actual robot motion, corrective signals were sent automatically to readjust the robot for proper functioning.

3.7. Testing and Validation

3.7.1. Motion Accuracy Testing:

Motion accuracy testing was conducted to verify that movement made was exact in terms of user input.

3.7.2. Latency Analysis:

Network delay was studied in order to ensure the highest rate of data transfer speed and response, with an assurance of a real-time experience for users.

3.7.3. Multi-Robot Control Validation:

To verify scalability, the mobile UI was tested for use to control several robots at once to ensure integration within multi-robot workcells is smooth.

4. CHALLENGES

- The communication between the Unity UI through TCP/IP and ABB RobotStudio suffers from occasional network congestion that results in delayed responsiveness. Command loss during operations also occurs due to unstable occurrence of WiFi connections, hence the need to stabilize the network and buffer the data.

- Execution Delay Commands - Sending the commands themselves was successful. The RAPID scripts and RobotStudio experienced a few microseconds of processing delays that resulted in delays in movement

execution. These delays cause a need for parallel command parsing and acceleration in execution, especially noticeable in complex robot movements.

- Security Threats - Additional cybersecurity threats from unauthorized access or hacking attempts were exposed via wireless control.

- Safe communication must be ensured through encryption, authentication methods, and firewall protection to prevent outside interference.

- Some compatibility problems existed - the system was largely made for ABB robots, but other brands - Fanuc, KUKA, and Yaskawa - utilize their own communication protocols and programming languages. Becoming a universal mobile control interface would involve extra integration frameworks to accommodate various robot models.

- Synchronization Issues – Timing synchronization between Unity UI commands and robot motions was intricate due to processing times, network latency, and command execution time. Smooth synchronized control needed adjusting response mechanisms and incorporating feedback loops.

- The Hardware Constraints - The early tests conducted with the above-stated microcontrollers included those using ESP32s whose supposed capabilities could not yet handle any data sent at rates per second. Although these are sufficient for prototyping under a minor scale, production settings would require high-performing microcontrollers that would be able to achieve much faster response and data processing.

- Scalability & Multi-Robot Control – Scaling the system to operate multiple robots at once posed issues regarding task synchronization, collision evasion, and load balancing. Efficient scheduling mechanisms had to be employed to make multi-robot operation smooth.

- Some failures in data transmission - for example, packets lost or commands misread - would make the robot behave erratically. For example, using error detection, redundant data transmission, and real-time feedback would have given a far higher reliability factor, mechanisms were required to ensure accurate and reliable movement execution.

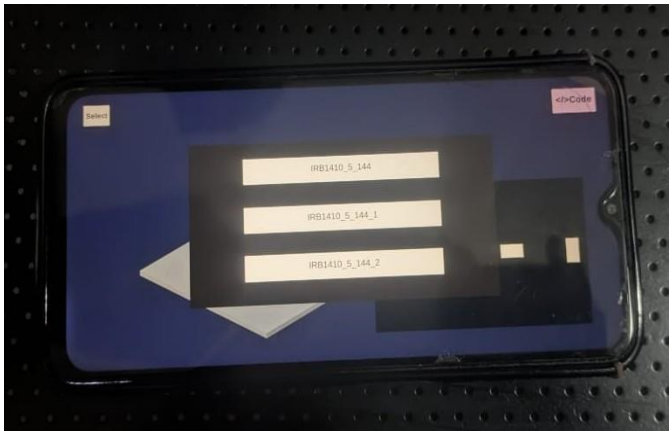

Fig -7: Selection of Robot from Multi Robot Setup

5. CODE BREAKDOWN:

5.1. RAPID Code (Robot Controller Side):

The ABB controller contains RAPID code that holds a number of procedures (reset, moveJoint1, moveJoint2, moveJoint3) that move the robot's joints. They control the movements of the joints based on external devices' command values.

• Main Procedure (main): This procedure waits for commands (received from the mobile app or PC server) and calls the corresponding movement functions. These commands are mapped to different joint movements such as moveJoint1, moveJoint2, moveJoint3, and reset, all of which change the position of the robot. The value variable manages the joint angle changes, enabling precise motion.

• Movement Procedures (moveJoint1, moveJoint2, moveJoint3): These procedures update the joint positions according to the received command. All movement commands would make the joints move either constant amounts or change values with the value and value2 of the robot parameters.

The logic provides real time accurate control to the mechanical arm with flexible adjustment for dealing with the system.

5.2. PC Server Code (Middleware for communication):

This PC server with TCP server in C# used to implement it, is interfaced with ABB Robot Controller and commands from mobile client are processed by RC PC server.

• StartServer(): A TCP server waits for incoming connections on a defined IP address and port. Once this has been done, the server starts reading commands given by the mobile client. Each of these commands changes the value variable, which is in line with its integrity according to a movement procedure of RAPID code (moving joint).

• SetRapidVariableValue(): This function will update the relevant RAPID variable either value.. robot so that it can perform the proper action, for instance turning a joint or initializing arm position by receiving a command.

The server provides real-time control over the robot by decoding the commands received from it and communicating them to the robot controller through the relevant protocol.

5.3. Mobile Client Code (User Interface and Control):

The mobile client developed with Unity is the interface used by any user to interact with the robot.

• UI Elements: Various buttons are established in Unity for sending corresponding commands to the server. Each button sends a corresponding joint movement on the robot.

• SendData(): The respective data is transferred to the PC server through TCP as soon as you click on the mobile UI button. This data is then sent to ABB robot for performing the execution.

• AngleAdjust() and Movement: The AngleAdjust() method adjusts the specified angle of the joint by turning the joint (joint1, joint2, or joint3) according to the received command. The command is from mobile client. It implements smooth interpolation so that it can move delay to desired position.

6. THE COMMUNICATION SYSTEM

Generally, the system is a hybrid of client-server because on one end is a mobile/PC application acting as the client, having been allocated an IP address, its respective port number. The other end is the server with its respective port number, and this one listens in intermittently in order to allow the intermediary that controlled the communication between the application and the various robot controllers. The controllers such as the ABB IRC5 takes commands from the said server and ensures the required movement in the robots attached to it. Thus, this configuration ensures an efficient data transfer so that the user can control the industrial robot and monitor the working condition from a distance without needing a conventional teach pendant.

6.1. Data Transmission process

6.1.1. User Input and Command Generation

The communication process starts when a user interacts with the mobile/PC application to give a motion command. This input command is typically in standard format, such as MoveJ p1, v100, fine, tool0, where MoveJ is a moving instruction. The application then sends a notification to the user, giving real-time feedback on the execution status of the robot. This feedback ensures reliability and permits users to keep track of movement without requiring explicit visual confirmation.

6.2. Communication Protocols Employed

The system is based on a client-server protocol of communication, where the client is played by the mobile/PC application through requests, and the server receives the requests and sends them to the controllers of the robots. Traffic is based on the TCP/IP protocol, and each device has its own unique IP address for communicating with assigned port numbers. The server in the system is on IP address with its respective port number and the client communicates from server IP address with its respective port number. A critical part of this arrangement is the ACK response mechanism that provides reliability through ensuring that all commands executed are acknowledged. Execution of a motion command implies that an ACK signal is sent by the robot controller and routed back through the server to the mobile/PC application. Thus, there is no command loss or incorrect execution represented for a reliable, error-free communication platform for industrial robot control.

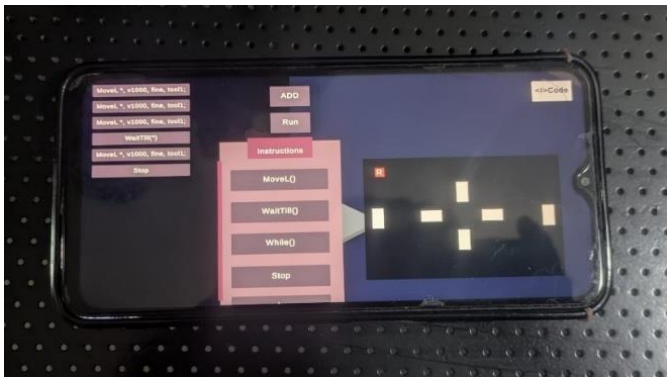

Fig -8: Programming in Mobile Controller

7. FUTURE SCOPE

The development in the future shall consist of the integration of strong encryption techniques, authentication mechanisms, and intrusion detection whereby any unauthorized use and cyber attacks will be avoided [14], Integration of a Digital Twin for the industrial robot will enable real-time simulation and monitoring and optimization of robot performance [18], Enlargement of the system to also cover multiple brands of robots and operating systems will, therefore, further its industrial applicability [16], Incorporation of AI-based predictive analytics that would improve robot control accuracy and efficiency [15] and integrate AI-based self-correction capabilities to detect and correct errors in robot execution [17].

8. CONCLUSION

In this project, the feasibility and effectiveness of a mobile-based control solution for industrial robots replacing the conventional teach pendant have been convincingly demonstrated [10]. The Unity UI, ESP modules, and ABB RobotStudio were integrated to form a complete framework for wireless communication, which could seamlessly transmit and receive data from robotic arms for real-time control [11]. The proposed mobile control system is more portable, cost-effective, and flexible compared to its conventional counterpart, which is bulky, expensive, and proprietary [12]. With this system, the operator can remotely control robots, which liberates them from physical constraints and enables enhanced workflow management in the industrial environment [4].

Multi-robot control is another feature that connects this system to current manufacturing relevance. Instead of requiring multiple teach pendants for different robots, a single mobile interface may efficiently manage two or even three robotic arms so that they can operate in sync, thereby reducing the need for training and operationalization [11]. Using wireless communication protocol, such as Wi-Fi and TCP/IP helps reduce latency and deliver fast and reliable data transfer for accurate manipulation of robots almost without any delay [12]. This validation facilitated in ABB RobotStudio confirms that the mobile solution can replace traditional programming efficiently, thus proving its worth in industrial automation [10].

Besides, the project implementation followed the principles governing Industry 4.0 that emphasize digital transformation, smart manufacturing, and automation toward the optimization of industrial processes [7]. Featuring easy integration of industrial robots with a user-friendly mobile application, the project boosts operational efficiency while simplifying robot programming and monitoring, thus making automation more user-friendly to a wider audience [8]. Despite its success, challenges such as connectivity reliability, cybersecurity risks, and real-time responsiveness still need further investigation [13]. This project represents a significant step toward intelligent, mobile-driven industrial automation, paving the way for more adaptive, cost-effective, and efficient robotic control systems in manufacturing environments [8].

REFERENCES:

[1]. Siciliano, B., & Khatib, O. (2016). Robotics and the handbook. In *Springer Handbook of Robotics* (pp. 1-6). Cham: Springer International Publishing.

[2]. Arents, J., & Greitans, M. (2022). Smart industrial robot control trends, challenges and opportunities within manufacturing. *Applied Sciences*, *12*(2), 937.

[3]. Żółkiewski, S., & Galuszka, K. (2015). Remote control of industry robots using mobile devices. In *New Contributions in Information Systems and Technologies: Volume 2* (pp. 323-332). Springer International Publishing.

[4]. Dahir, M. A., Obaid, A., Ali, A., Mohammed, A., Abou-ElNour, A., & Tarique, M. (2016). Mobile Based Robotic Wireless Path Controller. *Netw. Protoc. Algorithms*, *8*(2), 20-38.

[5]. Khamis, A., Hussein, A., & Elmogy, A. (2015). Multi-robot task allocation: A review of the state-of-the-art. *Cooperative robots and sensor networks 2015*, 31-51.

[6]. Willig, A., Matheus, K., & Wolisz, A. (2005). Wireless technology in industrial networks. *Proceedings of the IEEE*, *93*(6), 1130-1151.

[7]. Lu, Y., Xu, X., & Xu, J. (2014). Development of a hybrid manufacturing cloud. *Journal of manufacturing systems*, *33*(4), 551-566.

[8]. Ni, J., & Balyan, V. (2021). Research on Mobile User Interface for Robot Arm Remote Control in Industrial Application. *Scalable Computing: Practice and Experience*, *22*(2), 237-245.

[9]. Fernando, W. A. M., Jayawardena, C., & Rajapaksha, U. U. S. (2022, September). Developing a user-friendly interface from robotic applications development. In *2022 international research conference on smart computing and systems engineering (SCSE)* (Vol. 5, pp. 196-204). IEEE.

[10]. Alvarado, D., & Asif, S. (2024). A framework for controlling multiple industrial robots using mobile applications. *arXiv preprint arXiv:2403.07639*.

[11]. Kirci, S., Birgul, U., & Atali, G. (2024). A Multi-Functional Web Control Interface for Industrial Autonomous Mobile Robot Fleets. *The European Journal of Research and Development*, *4*(4), 101-109

[12]. Wang, R., Ji, L., Ren, T., He, S., & Shi, Z. (2020, July). A low-latency and interoperable industrial Internet of Things architecture for manufacturing systems. In *2020 IEEE 18th International Conference on Industrial Informatics (INDIN)* (Vol. 1, pp. 859-864). IEEE.

[13]. Maggi, F., Quarta, D., Pogliani, M., Polino, M., Zanchettin, A. M., & Zanero, S. (2017). Rogue robots: Testing the limits of an industrial robot's security. *Trend Micro, Politecnico di Milano, Tech. Rep*, 1-21.

[14]. Pogliani, M., Quarta, D., Polino, M., Vittone, M., Maggi, F., & Zanero, S. (2019). Security of controlled manufacturing systems in the connected factory: the case of industrial robots. *Journal of Computer Virology and Hacking Techniques*, *15*, 161-175.

[15]. Zhang, Y., Wang, H., & Li, T. (2024). AI-Driven Optimization for Industrial Robot Control: A Deep Learning Approach. *IEEE Robotics and Automation Letters, 9(3),* 1024-1035.

[16]. Wang, L. (2017). An overview of internet-enabled cloud-based cyber manufacturing. *Transactions of the Institute of Measurement and Control*, *39*(4), 388-397.

[17]. McMillan, L. (2023). *Artificial intelligence–enabled self-healing infrastructure systems* (Doctoral dissertation, University of London, University College London (United Kingdom)).

[18]. Liu, X., Gan, H., Luo, Y., Chen, Y., & Gao, L. (2023). Digital-twin-based real-time optimization for a fractional order controller for industrial robots. *Fractal and Fractional*, *7*(2), 167.