Computer vision and machine learning-based control for a 6-degree-of-freedom robotic arm

1st João Pedro Barcelos de Assis
PPGIa-PUCPR
Curitiba, Paraná, Brazil
joao.barcelos@pucpr.edu.br

2nd Gustavo Andreas Günther
PPGIa-PUCPR
Curitiba, Parana, Brazil
gustavo.gunther@pucpr.edu.br

3rd Marcelo Eduardo Pellenz
PPGIa-PUCPR
Curitiba, Parana, Brazil
marcelo.pellenz@ppgia.pucpr.br

4th Marco Simões Teixeira
PPGIa-PUCPR
Curitiba, Paraná, Brazil
simoes.marco@ppgia.pucpr.br

2025 IEEE International Conference on Industrial Technology (ICIT) | 979-8-3315-2195-0/25/$31.00 ©2025 IEEE | DOI: 10.1109/ICIT63637.2025.10965279

Abstract*—This paper presents the development of a control system for a robotic arm with six degrees of freedom (6 DoF) simulated in Gazebo, using machine learning. The system captures the operator's arm and hand movements using a camera and the Mediapipe framework, specifically extracting three landmarks: the shoulder, the index finger, and the thumb. These data points, combined with the corresponding angles of six servomotors, are used to train a sequential neural network model. The neural network consists of four dense layers (128, 64, 32, and 16 neurons) and an output layer with six neurons for the servomotor angles. The best results were achieved using a dataset of 3000 samples, 15% of which were used for validation. The model achieved a test loss of 0.1120, an accuracy of 82.03%, and a mean absolute error (MAE) of 0.2223 radians. A joint-specific analysis revealed that the gripper joint exhibited the highest error (0.3451 radians), suggesting the need for improvements in its control mechanism. The trained model's predictions were tested in a simulated environment, where angles are sent to Gazebo via ROS Actions. These results demonstrate the potential of the proposed approach while highlighting opportunities to enhance the gripper's control by simplifying its operation through distance-based mechanisms or by refining data collection methods.

***Index Terms*—Machine Learning, Robotic Arm, Gazebo, Mediapipe, Neural Networks, ROS, Rviz, TensorFlow**

I. Introduction

Machine learning-powered robotic arms are advancing real-world applications across factories, hospitals, and warehouses, bringing excellent safety, precision, and adaptability to complex tasks. Traditional robotic control systems, often limited by pre-programmed instructions, can need help in dynamic or unstructured environments. Machine learning introduces flexibility, enabling these systems to learn from data, adapt in runtime, and improve.

In manufacturing, robotic arms, such as the KUKA KR16, have become integral to Industry 4.0 applications, performing tasks like pick-and-place, welding, and additive manufacturing with enhanced speed and precision [1]. By leveraging machine learning, these robotic systems can optimize their actions, reducing material waste and enhancing productivity. This adaptability allows them to respond dynamically to changing conditions on the factory floor, reducing operational costs and errors.

In healthcare, robotic arms replicating a surgeon's movements in real-time have opened up exciting possibilities for precision surgeries. These robotic arms can assist in minimally invasive surgeries, leading to faster patient recovery [2]. There is also a promising future for remote surgeries, where a surgeon's movements could be mirrored by a robotic arm in a different location, helping bring specialized care to remote areas [3].

Logistics and warehousing are other areas where this technology could significantly impact. Robotic arms with adaptive control can handle, sort, and package items of different shapes and sizes, which is especially valuable in high-demand e-commerce environments [4]. These systems transform supply chain operations. By automating these tasks, chains are much faster and more accurate while reducing dependency on human labor for monotonous tasks.

However, most of the work that focuses on control for robotic arms does not concern itself with the practicality of use and comfort of the operator. Existing control methods often require specialized training, complex programming, or reliance on pre-defined kinematic models. This limits their accessibility and reduces the ease of use for individuals without technical expertise. Gesture-based robotic arm control systems offer a compelling solution to this challenge. Using computer vision and machine learning, these systems can provide a more natural interface, allowing operators to control robotic arms through intuitive movements. This not only reduces the learning curve but also enhances safety by minimizing operators' mental effort, reducing the likelihood of errors in critical tasks.

This project leverages machine learning to develop a close-to-real-time control system for a 6-DOF robotic arm. It captures human movement through computer vision and translates it into commands via a neural network model, providing

Graduate Program in Informatics (PPGIa), Pontifical Catholic University of Paraná (PUCPR), Rua Imaculada Conceição, 1155, 80215-901 Curitiba, Brazil

Cyber-Physical Systems Research Group (PUCPR/PPGIa) fluid and intuitive control for the operator. One of the key advantages of the proposed approach is its ability to implement control for a 6-degree-of-freedom robotic arm without the need to perform kinematic calculations. Instead, the robotic arm learns by imitating the operator's movement through computer vision and ML. This creates a system that is highly adaptable and responsive, making it suitable for scaling up across many real-world tasks and industries.

II. Related works

Artificial intelligence and neural networks have grown significantly in robotic control applications in recent years, especially in tasks requiring adaptive responses to varying environments. Several studies have explored methodologies integrating machine learning techniques to enhance control systems in robotics.

One approach involves using neural networks for visuomotor coordination, as demonstrated in the control of robotic arms. In such systems, neural networks map visual inputs to control actions. Researchers worked on a pneumatic robot arm that utilized a neural map algorithm, where visual feedback was integrated with neural network learning to enhance control accuracy [5]. Hybrid approaches combining genetic algorithms and multilayer perceptrons have also been explored in improving hand gesture detection [6]. Additionally, vision-based methods utilizing RGB-D cameras and dynamic gestures have demonstrated enhanced real-time performance, allowing for more natural interaction between operators and robots [7], [8].

The concept of digital twins has become prominent in robotics research. Digital twins allow the virtual training of robotic arms using various machine-learning methodologies, which can later be transferred to a physical system. In [9], researchers worked on a robot arm digital twin trained in Unity and demonstrated that virtual training significantly reduces the physical wear on robotic systems while enabling concurrent training of multiple agents for faster results. Further, methods such as reinforcement learning paired with optimization techniques like particle swarm optimization have been used to refine robotic arm control in both virtual and physical environments [10].

Recent studies have also employed gesture-based control systems to improve the teleoperation of robotic arms, integrating pose estimation and machine learning to capture operator intentions [11]–[13]. Researchers have demonstrated how computer vision eliminates the need for specialized hardware attached to the operator. Employing an adaptive pipeline for real-time control, users can operate a mobile robotic arm through body movements captured via an RGB camera [14]. Similar techniques have been explored for human-robot interaction, emphasizing efficient face and gesture recognition approaches that simplify operator input while maintaining robust performance [15].

These studies highlight the versatility and efficiency of neural network-based control in robotics, especially when combined with simulation environments or digital twins for training. They provide valuable insights into developing adaptable systems that can interoperate across virtual and real-world environments while leveraging gesture recognition to enhance usability and operator comfort.

III. Methodology

Figure 1 illustrates the system workflow. Starting at Step 1, Mediapipe extracts landmarks from the operator's arm and hand (shoulder, index finger, and thumb). These landmarks are then preprocessed before being directed either to control or to train a new model.

For controlling the arm, this data is then inserted into the ML-trained model, which predicts the joint angles based on the captured landmarks. These predicted angles are sent to Gazebo through ROS2 Actions (Publish Predicted Angles), where the robotic arm executes the movements (Execute Arm Movements).

If a new model needs to be trained, the workflow branches to Step 2, where the RViz simulation tool is used to capture the ground truth joint angles by manually aligning them with the operator's gestures. This data is stored in the Model Training Database, along with the preprocessed landmarks, to serve as training data. The Build and Train Predictive Model step uses this data to produce a new trained model, which is then used in the control loop.

In this section, we explore each of these steps shown in Figure 1, detailing the processes of capturing the operator's movements, training the machine learning model, and evaluating the robotic arm's control in a simulated environment.

A. Robotic Arm Simulation

The robotic arm used in this project is a simulated model with six degrees of freedom (6 DoF) implemented in Gazebo (Figure 2), based on a model provided by Doosan Robotics. Each degree of freedom corresponds to an axis of movement of the robotic arm. The joints are actuated virtually by mimicking servomotors' behavior, and each joint is capable of rotating within a range of 360 degrees. The simulation provides a safe and flexible environment for training and testing without relying on physical hardware.

B. Computer vision

The proposed methodology uses computer vision to identify the operator's movement. The MediaPipe algorithm [17] was chosen to identify the operator's movements. With a low computational cost, it can identify the human body, such as hands and poses, through an RGB camera. In this work, a Lenovo 300 camera was used with the Mediapipe library to capture the movements of the operator's arms and hands.

The system extracts three key landmarks: the shoulder, the index finger, and the thumb. These points are represented as X, Y, and Z coordinates, providing information about the position and orientation of the arm and hand within the frame. This selection simplifies the data while preserving essential movement information. Specifically, as shown in Figure 3, point 12 corresponds to the shoulder joint, while points 4 and 8 represent finger positions considered for control.

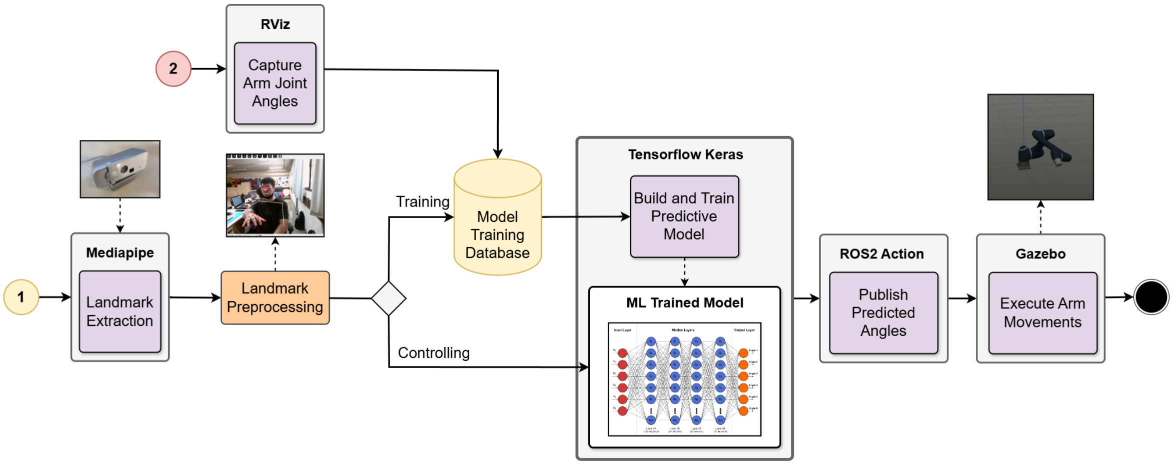


Fig. 1: System workflow: from data capture with Mediapipe to robotic arm control in Gazebo.

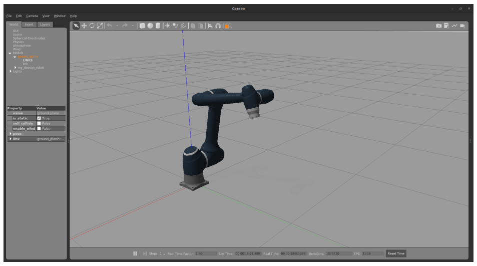
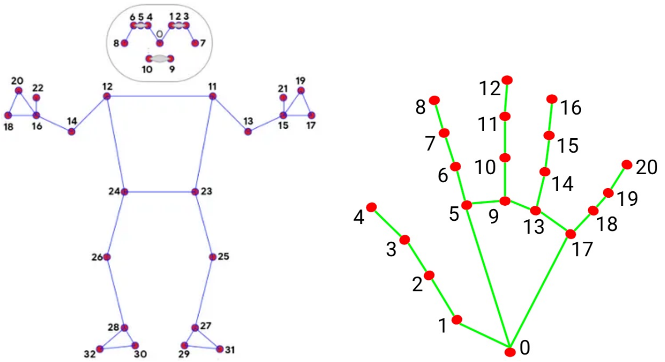


Fig. 2: Simulation of the robotic arm in Gazebo, based on the Doosan Robotics model [16].



(a) Mediapipe body landmarks.

(b) Mediapipe hand landmarks.

Fig. 3: Mediapipe landmarks (Source: MediaPipe Solutions Guide [18]).

C. Creating the dataset

To associate the captured landmarks with the angles of the robotic arm's joints, we also used a simulation in RViz provided by Doosan Robotics [19]. The simulation includes a joint state controller (*joint_state_controller*) with a graphical user interface (GUI), allowing manual adjustment of the joint angles. While the operator performs specific movements with the arm and hand, we can manually adjust the angles of the servomotors in the GUI to match the physical movements. The angle values are then captured online through the `/joint_states` topic in ROS2, forming a pair of time-stamped data that links the captured landmarks with the corresponding arm position.

D. Dataset Preprocessing

The captured landmarks and joint angles are stored in a single CSV file, forming the dataset for training. Each row in the CSV represents a sample used during training, containing:

- **9 input features**: Coordinates of the shoulder and hand landmarks.
- **6 output features**: Corresponding arm motor angles.

The collection process resulted in several datasets, from which we selected the best-performing one in preliminary tests, with a total of 3000 samples. This dataset was considered the most suitable for training. Before training, the data was pre-processed. The landmark coordinates were normalized relative to a single point. We considered each point's X, Y, and Z distance from the shoulder point (Mediapipe point 12). In this way, we aimed to simplify the database, reducing the training input data to only 6 values (normalized coordinates of the finger and thumb landmarks). Finally, the set of 3000 samples was divided into training and validation, using 85% for training and 15% exclusively for validation.

E. Machine Learning Model

We developed a sequential neural network model using TensorFlow and Keras [20]. The architecture was designed to map 6 input attributes (X, Y, Z coordinates of the landmarks) to six output joint angles. For training, we defined the loss function as MSE (Mean Squared Error) using Keras's Adam optimizer (Adaptive Moment Estimation). We also considered 1200 epochs of training in order to avoid significant overfitting. The neural network model used in training, as seen in Figure 4, consists of an input layer with six attributes representing normalized landmark coordinates, followed by four hidden dense layers with ReLU activation (128, 64, 32, and 16 neurons, respectively). The output layer contains six neurons with a linear activation function, predicting the continuous joint angles of the robotic arm.

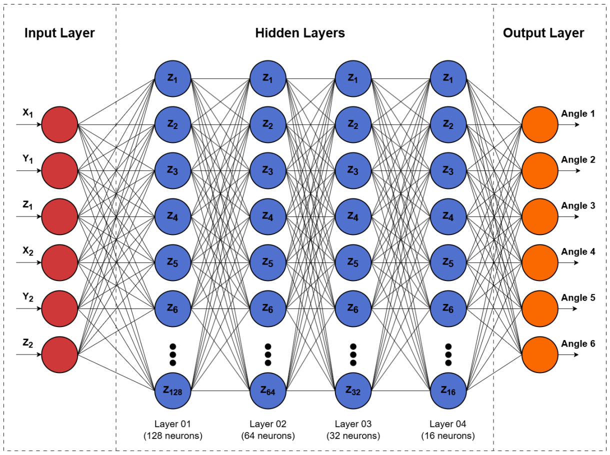
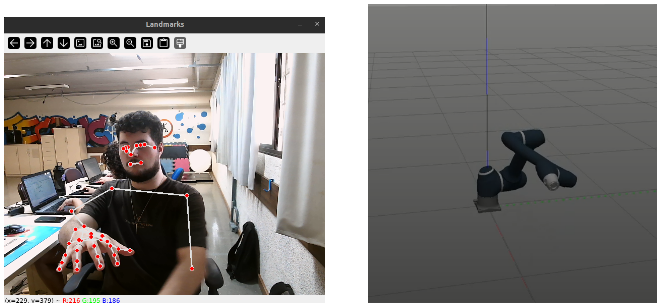


Fig. 4: Neural network model used in training.

F. Simulations and Tests

To test the trained model, we used a simulation in Gazebo, similar to the one used in the data collection stage. The simulation allowed close to real-time prediction of joint angles based on new Mediapipe landmark data captured. Then, predicted angles are sent to Gazebo as ROS actions. The simulation enabled quick adjustments to training parameters by visually evaluating the model's performance for significant discrepancies. Additionally, it allowed us, later on, to collect data from the simulated arm via ROS topics to assess the quality of our training more effectively. This approach provided a safe testing environment, isolating errors related to training and model prediction from potential hardware limitations when testing on a real robotic arm.

Figure 5 illustrates the full process, showing an operator performing a hand gesture with Mediapipe capturing landmarks, alongside the Gazebo simulation's response, where the robotic arm replicates the gesture based on the model's predictions.



(a) Operator performing a hand gesture.

(b) Gazebo's response to the operator's movement.

Fig. 5: Comparison between the operator's hand movement and the robotic arm's response in the Gazebo simulation.

IV. Results

In this section, we present the results obtained from the training and evaluation of the model developed for controlling the robotic arm.

A. Training Data Analysis

The training process was analyzed using two key graphs. The *Training and Validation Loss* (Figure 6) shows how the loss decreased and stabilized over 1200 epochs, indicating convergence. The second graph, *Training and Validation Accuracy* (Figure 7), illustrates the accuracy's progression, demonstrating that the model identified patterns and reached its performance limit with the available data.

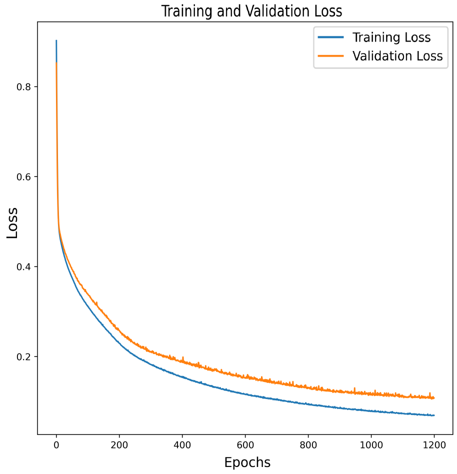


Fig. 6: Training and validation loss over the epochs.

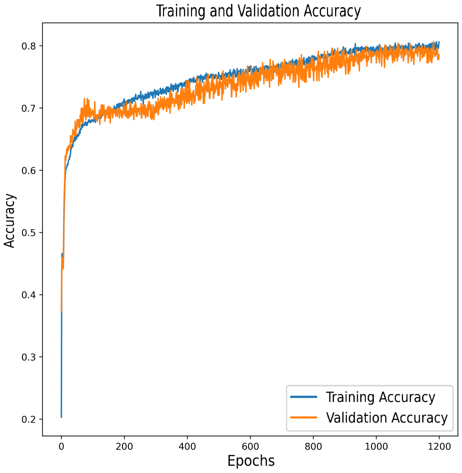


Fig. 7: Training and validation accuracy over the epochs.

B. Model Performance

The model was trained using a dataset of 3000 samples, of which 15% were reserved exclusively for validation. The training process achieved a test loss of 0.1120, a test accuracy of 82.03%, and an overall mean absolute error (MAE) of 0.2223 radians.

The **loss**, calculated as the mean squared error (MSE) between predicted and actual joint angles, indicates the model's overall error during evaluation. A value of 0.1120 suggests that the model effectively minimized large deviations in its predictions.

The **accuracy** reflects the percentage of predictions within an acceptable error margin of 0.15 radians (approximately 8.6 degrees), with 82.03% of the predicted joint angles considered sufficiently close to the true values.

The **MAE** (mean absolute error) quantifies the average deviation between predicted and true joint angles. This metric suggest that, on average, the model's predictions for the joint angles deviate by approximately 0.2223 radians from the ground truth which corresponds to an average error of approximately 12.73 degrees.

A summary of the training results is presented in Table I

Metric	**Value**
Loss	0.1120
Accuracy	82.03%
MAE (overall)	0.2223
Total Samples	3000
Training Samples	2550
Validation Samples	450
Total Trained Parameters	11862

TABLE I: Summary of the training results with MAE presented in radians.

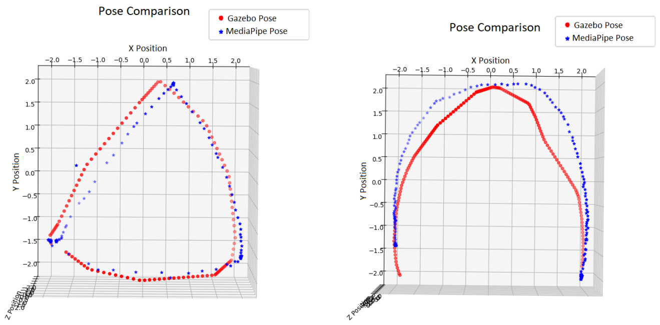
Among the joints, Joint 6 exhibited the highest mean absolute error, at 0.3451 radians. In the robotic arm simulation, Joint 6 is responsible for controlling the opening and closing of the gripper. This higher error can be attributed to the complexity of accurately mimicking the operator's subtle hand gestures for gripper control. The MAE for each joint is summarized in Table II.

Joint	**MAE (radians)**
Joint 1	0.1843
Joint 2	0.1485
Joint 3	0.1791
Joint 4	0.1677
Joint 5	0.3090
Joint 6	0.3451
Overall MAE	**0.2223**

TABLE II: Mean absolute error (MAE) for each joint during testing.

C. Pose Comparison Analysis

To evaluate the correspondence between the operator's movements (captured by Mediapipe) and the robotic arm's response, we performed two distinct tests: a triangular movement test and an arch movement test. The resulting graphs, *Pose Comparisons* (Figure 8), overlay the Mediapipe landmarks (blue) with the simulated gripper's tip positions (red). These analyses demonstrate the system's ability to approximate the operator's intended movements for different motion patterns.



(a) Triangular movement test.

(b) Arch movement test.

Fig. 8: Pose comparison tests: Mediapipe landmarks (blue) versus gripper tip (red).

D. Deviation and Error Analysis

To assess the system's stability during static poses, we measured the deviation of the gripper's position from its expected target over time. Figure 9 shows the deviations along the X, Y, and Z axes and the Euclidean distance error from origin (red, green, yellow, and blue respectively). The analysis reveals a mean euclidean distance error of approximately 0.0217 m (dashed black line).

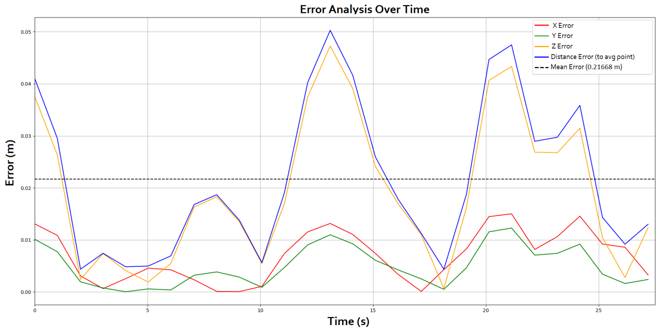


Fig. 9: Deviation and error analysis over time: deviations along X, Y, Z axes and euclidean distance error.

The deviations are primarily attributed to noise in the Mediapipe output, which can mistakenly interpret minor, unintentional movements of the operator's hand, such as shakes, as deliberate actions, especially since no threshold was applied to verify significant motion. Among the axes, the Z-axis shows the largest contribution to the error, likely due to challenges in accurately interpreting depth information from 2D camera inputs in Mediapipe.

To further illustrate this behavior, Figure 10 demonstrates the robotic arm's response to small variations in landmark detection. Although the operator's hand was intended to remain stationary, Mediapipe detected a positional change within a range of approximately 0.01 m over time. The Gazebo simulation replicated these variations, resulting in undesired micro-movements of the robotic arm. This highlights the need for a threshold mechanism to filter insignificant variations in landmark detection and improve system stability.

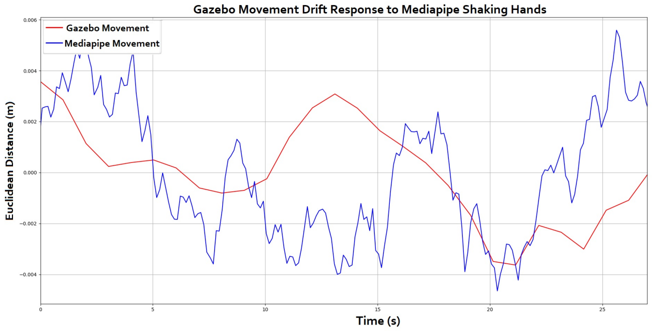


Fig. 10: Gazebo Movement Drift Response to Mediapipe: The robotic arm's response to small variations in the operator's landmarks detection, despite the operator's intention to remain stationary.

V. Conclusion

The results demonstrate significant progress in the system's ability to predict and replicate human hand gestures with the robotic arm. The training process yielded a test loss of 0.1120, an accuracy of 82.03% correct predictions considering an error margin of 0.15 radians, and an overall mean absolute error (MAE) of 0.2223 radians. These metrics indicate that the model performs well in most scenarios, with accurate predictions for the majority of joint angles. However, our analysis reveals specific problems that require closer attention.

The **Deviation and Error Analysis** (Figure 9) demonstrates a lack of system stability during static poses. The analysis reveals a mean Euclidean error of approximately 0.0217 m, with the Z-axis being the primary contributor to the deviations. This behavior can be attributed to noise in the Mediapipe output.

Finally, while the overall MAE is relatively low, the joint-specific analysis in Table I reveals that Joints 5 and 6 exhibit the highest errors. Joint 5, which represents the wrist articulation, has an MAE of 0.3090 radians, and Joint 6, responsible for controlling the gripper's opening and closing, presents an even higher MAE of 0.3451 radians. These joints contribute significantly to the overall error.

A potential solution to improve the gripper's performance (Joint 6) is to implement a distance-based control system for its opening and closing. Calculating the Euclidean distance between the operator's thumb and index finger to control the gripper's opening and closing could simplify the training process by removing one output on our network model, potentially enhancing its overall accuracy. By excluding Joint 6 from the analysis, the recalculated overall MAE for the remaining joints drops to 0.1977 radians, demonstrating the potential improvement in performance if the gripper control mechanism is simplified.

Future work will focus on integrating filtering mechanisms to reduce the impact of noise in Mediapipe's outputs, particularly for the Z-axis, and on a distance-based control for the gripper.

References

[1] W. S. Barbosa, M. M. Gioia, V. G. Natividade, R. F. F. Wanderley, M. R. Chaves, F. C. Gouvea, and F. M. Gonçalves, "Industry 4.0: examples of the use of the robotic arm for digital manufacturing processes," *International Journal on Interactive Design and Manufacturing (IJIDeM)*, vol. 14, p. 1569–1575, Sept. 2020.

[2] W. P. Geis, H. C. Kim, J. Brennan, Edward J., P. C. McAfee, and Y. Wang, "Robotic arm enhancement to accommodate improved efficiency and decreased resource utilization in complex minimally invasive surgical procedures," in *Medicine Meets Virtual Reality*, vol. 29 of *Studies in Health Technology and Informatics*, pp. 471–481, IOS Press, 2023.

[3] B. Challacombe and P. Dasgupta, "Telemedicine- the future of surgery," *The Journal of Surgery*, vol. 1, p. 15–17, Oct. 2003.

[4] N. Sobhan and A. S. Shaikat, "Implementation of pick & place robotic arm for warehouse products management," in *2021 IEEE 7th International Conference on Smart Instrumentation, Measurement and Applications (ICSIMA)*, vol. 9, p. 156–161, IEEE, Aug. 2021.

[5] T. Hesselroth, K. Sarkar, P. P. van der Smagt, and K. Schulten, "Neural network control of a pneumatic robot arm," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 24, no. 1, pp. 28–38, 1994.

[6] S. S. Chaeikar, F. M. Asl, S. Yazdanpanah, and A. Roohi, "Hand gesture detection by genetic algorithm and multilayer perceptron," *Iberian Journal of Applied Sciences and Innovation*, vol. 2, January 2022.

[7] C. Hu, X. Wang, M. K. Mandal, M. Meng, and D. Li, "Efficient face and gesture recognition techniques for robot control," in *CCECE 2003 - Canadian Conference on Electrical and Computer Engineering. Toward a Caring and Humane Technology (Cat. No.03CH37436)*, IEEE, 2004.

[8] J. Qi, L. Ma, Z. Cui, and Y. Yu, "Computer vision-based hand gesture recognition for human-robot interaction: a review," *Complex Intell. Syst.*, vol. 10, pp. 1581–1606, Feb. 2024.

[9] M. Matulis and C. Harvey, "A robot arm digital twin utilising reinforcement learning," *Computers & Graphics*, vol. 95, pp. 106–114, 2021.

[10] M. Mueangprasert, P. Chermprayong, and K. Boonlong, "Robot arm movement control by model-based reinforcement learning using machine learning regression techniques and particle swarm optimization," in *2023 Third International Symposium on Instrumentation, Control, Artificial Intelligence, and Robotics (ICA-SYMP)*, pp. 83–86, 2023.

[11] W. Qi, S. E. Ovur, Z. Li, A. Marzullo, and R. Song, "Multi-sensor guided hand gesture recognition for a teleoperated robot using a recurrent neural network," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 6039–6045, 2021.

[12] S. E. Khan and Z. C. Danziger, "Continuous gesture control of a robot arm: Performance is robust to a variety of hand-to-robot maps," *IEEE Transactions on Biomedical Engineering*, 2023.

[13] Q. Gao, Y. Chen, Z. Ju, and Y. Liang, "Dynamic hand gesture recognition based on 3d hand pose estimation for human–robot interaction," *IEEE Sensors Journal*, vol. 22, no. 18, pp. 17421–17430, 2021.

[14] D. Martinelli, "Sistema adaptativo para teleoperação de base móvel através de reconhecimentos gestuais," Master's thesis, Universidade Tecnológica Federal do Paraná, 2022.

[15] Z. Yu, C. Lu, Y. Zhang, and L. Jing, "Gesture-controlled robotic arm for agricultural harvesting using a data glove with bending sensor and OptiTrack systems," *Micromachines (Basel)*, vol. 15, p. 918, July 2024.

[16] D. Robotics, "doosan-robot: Ros packages for doosan robotics manipulators." https://github.com/doosan-robotics/doosan-robot, 2020. GitHub repository, Accessed on: 1 out. 2024.

[17] C. Lugaresi, J. Tang, H. Nash, C. McClanahan, E. Uboweja, M. Hays, F. Zhang, C.-L. Chang, M. G. Yong, J. Lee, *et al.*, "Mediapipe: A framework for building perception pipelines," *arXiv preprint arXiv:1906.08172*, 2019.

[18] G. Developers, "Mediapipe solutions guide." https://developers.google.com/mediapipe/solutions, 2024. Accessed on: 1 out. 2024.

[19] D. Robotics, "Doosan robotics official website." https://www.doosanrobotics.com/en/Index, 2024. Accessed on: 1 out. 2024.

[20] TensorFlow Developers, *Keras Guide*. Google, 2023. Accessed on: 1 out. 2024.