Trajectory Control Based on On/Off, Fuzzy Logic and Convolutional Neural Networks for an Industrial Robot Arm: An Experimental Comparison

José Raúl Castro, David Paúl Rosales, and Carlos Calderon-Cordova, *Senior Member, IEEE*

***Abstract*— The objective of the present study is to compare three control approaches: ON/OFF control, fuzzy logic, and convolutional neural networks (CNN) implemented in Python for controlling the real-time trajectory tracking of a six-axis industrial robotic arm. This analysis has significant applications in fields that require a high level of precision, such as automated welding and surgical interventions in the medical domain. To evaluate the performance and adaptability of the control models, we will analyze the results using metrics such as Mean Squared Error (MSE), Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), as well as metrics including Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index (SSIM), Jaccard Index, and Pearson's correlation coefficient. The results obtained reveal valuable information about the advantages and limitations of each control approach, highlighting the effectiveness of CNNs in visual perception and trajectory tracking. The ability of CNNs to interpret visual complexities is presented as a key factor for their success in industrial robotics and automation applications, suggesting a promising future for these technologies in dynamic environments.**

Link to graphical and video abstracts, and to code: https://latamt.ieeer9.org/index.php/transactions/article/view/8702

***Index Terms*—industrial robotic arm, line following, trajectory control, fuzzy logic, convolutional neural network.**

I. Introduction

Uncertain systems are those that experience disturbances; that is, in which events occur due to elements that are both an intrinsic part and part of the working environment of a machine or manipulator, whose occurrence directly or indirectly influences its operation, causing generally unforeseen inaccuracies in the tasks performed. Disturbances represent a very important factor in the automation analysis of industrial robots, since they can affect critical parameters such as power, speed, kinematic performance and precision. There are some problems in the robust control formulations that exist in the literature. In particular, certain characteristics of the uncertainty need to be known and the measurement of the position and velocity of the joints is required for most of the controllers proposed for robotic manipulators.

This project was funded by UTPL through Grant PROY_INV_CE_2022_3609.

J. R. Castro, David P. Rosales, and C. Calderon-Cordova are with Universidad Técnica Particular de Loja, 110150 Loja, Ecuador (e-mails: jrcastro@utpl.edu.ec, dprosales2@utpl.edu.ec and cacalderon@utpl.edu.ec).

Over the world, robust controllers have been developed based on the estimation of uncertainty and disturbances, which do not require accurate information on uncertainties or accurate velocity measurement. For these cases, a robust observer and an uncertainty estimation technique are used, in addition to presenting numerical simulations and experimental results [1].

It is feasible to implement control in uncertain systems by estimating and compensating for the impact of uncertainties and disturbances through modifications in the nominal system's controller. This is achieved using techniques such as disturbance observers and unknown input observers. A common technique is time-delayed control (TDC), which estimates the effect of uncertainties and disturbances using past data, designing a controller that counteracts unknown dynamics and disturbances. Additionally, in robotics, techniques like Uncertainty and Disturbance Estimation (UDE), has been applied, addressing limitations of TDC. The UDE is based on the control of uncertainty and disturbances, reinforcing linearized control laws and overcoming problems of uncertainty limits and fluctuations in Sliding Mode Control (SMC). Robust control designs based on UDE have also been proposed for systems with and without linearity and state delays [2], [3].

Controlling a robot involves designing the torque of the joints or the actuator voltage so that they closely follow a desired trajectory. Classic servomechanism methods are applied individually to each joint using linear controllers such as the Proportional Integral Derivative (PID) controller [4], [5]. However, these controllers do not consider nonlinearities in the process or interactions between the joints, which becomes crucial for large displacements, high speeds, and accelerations [6]. To achieve high precision in trajectory, nonlinearity compensations and decoupling of the joints are required, which implies the use of fixed controllers. These controllers, common in robots, demand an exact knowledge of the dynamics and parameters of the system, which can lead to performance degradation and instability due to uncertainties in the design [7]. An effective solution is adaptive control schemes, where controllers automatically adjust to compensate for uncertainties [8]. These approaches are divided into model-reference adaptive control (MRAC) and self-tuning adaptive control (STAC) [9], [10]. The objectives of robot controllers include insensitivity to parameter uncertainties, unknown load variations, low computational demand, and decoupled joint response [11].

The literature proposes a variety of strategies to counteract

disturbances, including passive, active, and hybrid control methods, in addition to the application of machine learning techniques. This study provides crucial insights into the strengths and weaknesses of three different control methods, presenting effective solutions to evaluate precision and efficiency in the field of industrial robotics.

II. Techniques to Counteract Disturbances in Industrial Robots

A. Vibration

In the literature, various vibration control techniques for industrial robots have been discussed, with passive, active, and hybrid control being the primary methods. Passive control, employing devices like friction and hydraulic dampers, springs, and viscoelastic materials, effectively dampens low-frequency vibrations but falls short at higher frequencies. Conversely, active control utilizes sensors and actuators electrical, hydraulic, or pneumatic to detect and counteract vibration, proving more efficient for high-frequency vibrations. Hybrid control merges these two, leveraging their combined strengths to address a broader frequency range [12].

Moreover, control algorithms like PID control, a straightforward yet effective feedback method, and Model Predictive Control (MPC), a more complex but potentially more effective approach in nonlinear or high-disturbance scenarios, have been developed. Zhu et al. proposed a sensor-based vibration control method using a finite element model to identify a robot's natural frequencies and mode shapes [13]. This controller adapts in real-time to reduce vibration, showing significant improvements over non-adaptive controllers.

B. Load Variations

Load variations are common in industrial robots and affect their precision, speed, and stability. This refers to changes in the weight the robot must move, which can cause imbalances and errors in its movement [14]. To address this issue, force feedback sensors are used in the robot's joints, allowing for real-time correction and improving precision and efficiency, resulting in higher quality and reduced production time [15]. In addition to classical control approaches, adaptive control methods have been proposed, such as the Model-Based Adaptive Control (MBAC) for 6-axis robots. In this method, the system model is estimated in real-time using system identification techniques, and an adaptive controller is designed to adjust parameters according to load variations. This approach has proven effective in compensating for load variations and outperforms traditional control methods [15].

C. Sensor Noise

Sensor noise, a prevalent disturbance in industrial robots, undermines the accuracy and reliability of decision-making data. This noise arises from various factors like electromagnetic interference, temperature, humidity, and sensor material quality. Research efforts are increasingly focused on minimizing or controlling this noise. Key strategies include digital filtering, noise subtraction, frequency domain transformation, and employing machine learning for noise reduction.

Digital filters, notably low pass, high pass, band pass, and band reject, are instrumental in smoothing signals by filtering out irrelevant frequencies [16]. Recent advancements include a deep learning-based noise reduction model, particularly for acceleration sensors, utilizing a convolutional neural network trained on industrial robot vibration data. This model has demonstrated significant noise reduction, enhancing data accuracy. Another study highlighted the impact of electrical noise on force sensors, revealing substantial accuracy and reliability reductions in force measurements, with signal filtering systems shown to markedly improve force measurement quality amidst electrical noise [10].

D. Electric Current Variation

Current and frequency variations are common in industrial robots, affecting the robot's speed and position, which can decrease product quality and increase downtime. To address this issue, speed and position controllers are used to maintain a constant electric current and regulate frequency, thus minimizing variations [17].

Regarding frequency variations, frequency regulators are used to maintain a constant frequency in the electrical current supplied to the robot's motors, reducing speed and position variations. Some regulators also adjust the current to compensate for fluctuations in the electrical power supply.

Studies have demonstrated the effectiveness of these techniques. For example, one study evaluated a control system based on a position controller and an energy filter in a 6-axis industrial robot, achieving significant reductions in electrical current variations and improving the robot's precision and stability [18].

E. Friction in Motion

Friction in motion is a common disturbance that affects the performance and precision of industrial robots, increasing the necessary energy, wearing down components, and generating harmful vibrations. To address this problem, techniques such as lubrication have been developed, which reduce friction and wear, as well as decrease noise and vibrations in the system [19]. Studies have evaluated lubrication in SCARA robot drive systems, demonstrating that it significantly reduces friction, improving precision and stability in high-speed operations. Another effective approach to reducing friction is the application of special coatings on moving parts [20].

Table I summarizes the strategies and techniques recommended by specialized literature for mitigating disturbances, with the goal of optimizing precision and efficiency in robotic applications.

TABLE I
SUMMARY OF DISTURBANCES AND TECHNIQUES TO COUNTERACT THEM

Disturbance	Causes	Effects	Techniques Used
Vibration	Imbalance	Loss of accuracy	Vibration isolation.
	Imperfections in the parts	Material fatigue	Vibration absorption.
	External Forces	Deterioration of product quality	Use of shock-absorbing materials.
Load Variations	Changes in demand	Loss of accuracy	Charge Controllers.
	Voltage fluctuations	Deterioration of product quality	Voltage Regulators.
	Load imbalances	Damage to components	Adjusting Controller Parameters.
Sensor Noise	Electromagnetic interference	Inaccurate readings	Shielding the sensors.
	Voltage fluctuations	Loss of accuracy	Signal Filters.
	Interference from other signals	Control errors	Interference isolation.
Electrical Current Variation	Voltage fluctuations	Loss of accuracy	Voltage regulators.
	Overload on the power grid	Damage to components	Current controllers.
	Changes in energy demand	Deterioration of product quality	Energy Filters.
Friction in the movement	Wear on parts	Loss of accuracy	Use of lubricants.
	Material incompatibility	Premature wear of parts	Adjusting the Parts.
	Lack of lubrication	Increased energy consumption	Design of parts with compatible materials.
External agents	Unexpected changes in control application variables.	Crashes or accidents	Application of real-time control mechanisms.
		Unwanted application results	Adapting the trajectory to correct the error.
		Positional inaccuracy	

III. CASE STUDY

In order to compare control algorithms for trajectory tracking in high-precision applications, such as medical surgery or welding, a study will be conducted using a disturbance case that modifies the original direction and position of a planned trajectory. This will require correcting the robot's trajectory to adapt to the change. A camera will be used as a sensor to detect this change.

A. Six Degrees of Freedom Industrial Robot

For the case study, a six-axis industrial robot is considered, specifically the EPSON VT6L-A901S robot. It has six degrees of freedom and is capable of lifting up to 6 kilograms of weight from its head. The main characteristics are shown in Table II.

The robot moves using points stored in variables, which can be specified in Cartesian coordinates or in relative or absolute pulses (degrees) for each axis. Although it is possible to modify the robot's position individually for each axis, two main types of coordinated movements are primarily used: point-to-point and straight line.

Point-to-point movements focus on efficiency by preserving the orientations of the axes, which can lead to curved trajectories instead of straight lines, always seeking to reach the destination without superfluous movements.

TABLE II
MAIN CHARACTERISTICS OF THE VT6 ROBOT ARM

Feature	VT6 – A901S
Number of axes	6
Load	3 kg (nominal), 6 kg (maximum)
Mobility	170° on axis 1, -160° a 65° on axis 2, -51° a 190° on axis 3, 200° in axis 4, 125° in axis 5, 360° in axis 6
Moment	12 N*m for axes 4 y 5, 7 N*m for axis 6

In contrast, straight-line movements seek the mathematically shortest route between the start and end points, without worrying about adjusting the movement of one or several axes of the robot, resulting in a straight line in all cases.

B. Camera

A webcam connected to the control computer via USB will be used as a sensor. The XB201-XM-2 camera features a video resolution of 1980x1080 and the flame rate of 30 FPS. This camera is mounted on the robot's head (axis 5), with its orientation parallel to this axis, allowing its angle to vary synchronously with the axis.

C. Tracking Path

The research aims to evaluate the performance of different control algorithms in accurate trajectory tracking. For this purpose, a tracking curve has been designed that includes simulated variations affecting the robot's trajectory, particularly changes in the position of the line to be followed. Fig. 1 shows the trajectory that will be used to evaluate the performance of the algorithms.

Fig. 1. Curve for trajectory tracking. Note: Drawn on an A4 standard sheet of paper, 297 mm height and 210 mm width.

D. Proposed Scenario

The evaluation scenario for the algorithms will be maintained under uniform conditions for all, with the following characteristics:

- The camera will be mounted on axis 5 of the robot.
- The initial capture position will be set with axis 5 at a constant height of 18 cm above the worktable, without variations in this height, as it will only affect the X and Y axes of the robot. This height will allow the placement of a marker to trace the trajectory.
- A white paper sheet will be placed on the worktable, visible at the height of the camera, with a black-colored curve that will serve as a case study and application of the system. The same original curve is shown in Fig. 1.
- A black-colored marker will be installed on the robot's head, which will touch the blank white sheet at the beginning, validating the tracking of the original curve.
- Hardware connections will be made without positively or negatively affecting the process.
- Adequate lighting will be provided in the workspace to avoid generating false lines for the camera. Tests will be conducted during daylight hours to avoid shadows.

IV. Design of Control Algorithms and Math

In this work, we use Python to analyze the performance of three active control techniques for real-time trajectory tracking of a six-axis industrial robotic arm.

A. ON/OFF Based Active Controller

The active ON/OFF control, known for its simplicity and ease of implementation, is used in factories and household appliances, such as Heating, Ventilation, and Air Conditioning (HVAC). This method supplies full power until the set point is reached. Sensor-based feedback provides the value of the control variable in real-time discretely. When this value exceeds the set point, the power is turned off until the control variable value is updated. If the value falls below the set point within the desired ranges, it is turned on again to correct it. This process is continuously repeated, maintaining the control variable within the desired ranges over time [21], [22].

In this context, the control variable refers to the difference between the percentages of black pixels in the binarized image of the trajectory line, and the main actuator involves the robot's movement along the X-axis to correct its X position and advance in Y to follow the trajectory. Fig. 2 shows the diagram of the active control system for line tracking (external agent disturbance). The essential characteristics of this active control are detailed in Table III.

TABLE III
Features of On/Off Based Active Controller

Control variable	Difference between percentages of black pixels on the left and right side of the captured and binarized image.
Input	Webcam 1080p
Actuators	VT6L robot axes
Output	Robot X position

First, we have the control variable Δ_P, which represents the difference in the percentages of black pixels on each side of the image, calculated with the expression (1).

$$\Delta_P = P_l - P_r \tag{1}$$

Where P_l is the percentage of black pixels in the left half of the image, and P_r is the percentage of black pixels in the right half of the image. Then, we have the mathematical function that describes the basic operation of the controller is as (2).

$$f(\Delta_P) = \begin{cases} (1,0), & \Delta_P < -5 \\ (0,1), & -5 \leq \Delta_P \leq 5 \\ (-1,0), & \Delta_P > 5 \end{cases} \tag{2}$$

This function returns a two-position vector. The first position corresponds to the movement in X in millimeters that the robot should perform, and the second position corresponds to the movement in Y in millimeters that the robot should perform. It is also noted that a 5% maximum error percentage is allowed. This is necessary to prevent infinite loops, as there are cases where reaching an absolute 0% difference with the robot's actuators is impossible. Therefore, the robot must remain within the established ranges of ±5%.

The operation of the proposed control system is visualized as a flowchart in Fig 2.

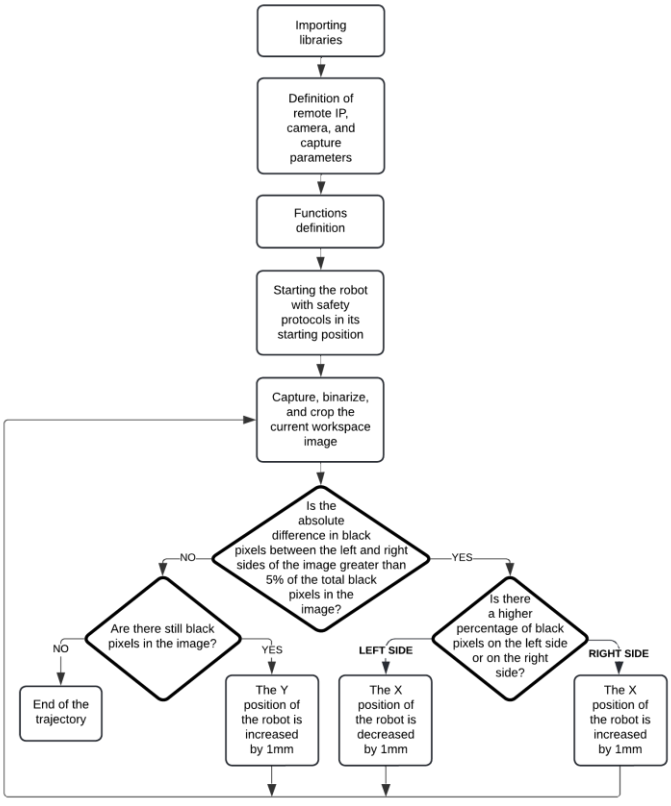


Fig. 2. ON/OFF based active controller flowchart.

B. Fuzzy Logic-Based Active Controller

Fuzzy logic is an extension of classical logic that addresses the imprecision and ambiguity present in natural languages. In recent years, the number and diversity of applications using

fuzzy logic have experienced significant growth. Fuzzy logic represents an approach to synthesizing diverse, even contradictory, control rules to arrive at a coherent decision. This method is distinguished by its ability to compute using words instead of numbers, leveraging tolerance for imprecision to reduce solution costs. Fuzzy logic allows the true values of variables to range between 0 and 1, facilitating the management of vagueness and ambiguity through the theory of fuzzy sets, which addresses uncertainties directly [23], [24]. This logic uses fuzzy sets where elements have degrees of membership instead of binary membership.

For the case study of this research project, the fuzzy logic controller will act on the control variable, which is the difference in black pixels to the left and right of the binarized image of the trajectory line that it follows. The values at the output of the controller will be the different positions in X and Y that the robot must adopt by moving its axes to make the control variable as close to 0 as possible. This will allow for greater precision compared to the previous percentage-based approach.

Fuzzy logic will be applied to two output signals for two actuators, separating the movement on the X-axis and the movement on the Y-axis of the robot into independent actuators. This will enable the generation of smoothed trajectories without a stepping effect. Fig. 4 shows the general flow diagram of the active control system for line tracking (external agent disturbance), and Table IV describes the essential characteristics of this active control.

TABLE IV
FEATURES OF FUZZY LOGIC-BASED ACTIVE CONTROLLER

Control variable	Difference between percentages of black pixels on the left and right side of the captured and binarized image.
Sensor	Webcam 1080p: 1920x1080
Actuators	VT6L robot axes Model: EPSON VT6-A901S
Output signals	X robot position Y robot position (only positive or zero)

In this proposed control system, the control variable is represented by Δ_N, which is the encountered error, meaning the difference in the number of black pixels on each side of the image, as expressed in (3).

$$\Delta_N = N_l - N_r \tag{3}$$

Where N_l is the number of black pixels on the left side of the image, and N_r is the number of black pixels on the right side of the image.

On the other hand, the mathematical function of a fuzzy control system will depend on the number of memberships used. For the current system, three well-differentiated categories are categorized: left, right, and center. These three categories describe the presented error, that is, on which side of the image the line is and then correct its position, attempting to approach the center. The mathematical function for these memberships is described in (4).

$$f(\Delta_N) = \begin{cases} 0, & \Delta_N < -7000 \\ \frac{\Delta_N + 7000}{-7000}, & -7000 \leq \Delta_N \leq 0 \\ \frac{7000 - \Delta_N}{7000}, & 0 \leq \Delta_N \leq 7000 \\ 0, & \Delta_N > 7000 \end{cases} \tag{4}$$

Fig. 3 shows that the mathematical function used in the control system is based on fuzzy logic, specifically a triangular membership function. The values of -7000 and +7000 represent the error range of the control variable, referring to the minimum and maximum differences between the black pixels of the image. The value of 7000 corresponds to the maximum number of black pixels that have been captured in the images obtained by the camera throughout the trajectory tracking of the case study. In other words, in each of the captured and processed images, the maximum number of black pixels found is 7000. In addition to this, it must be considered that the captured and processed images have a dimension of 770 pixels width and 100 pixels height. In this way, 7000 is the maximum value of pixels that the curve can occupy in the image, and therefore the maximum error that can occur in the control variable, which is desired to be as close to 0 as possible. Fuzzy logic regulates actuator movement within specified intervals, adjusting output at extremes for error correction, and enabling progression through subsequent adjustments based on remeasured control variables.

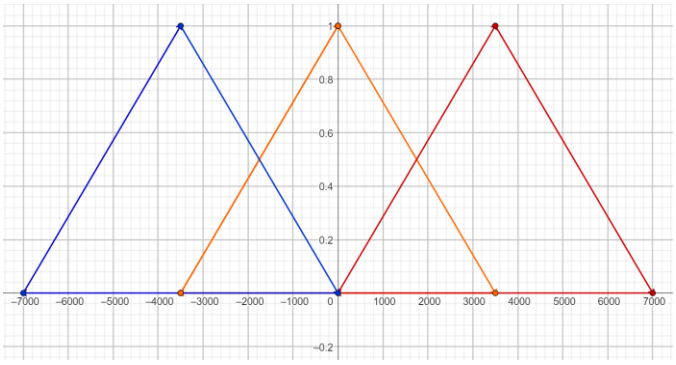


Fig. 3. Graphical representation of the membership functions of the fuzzy logic-based active controller.

The change in the robot's position will be proportional to the error and will depend on the opinions of all elements mentioned in the rules. The output signal will not be a signal with a constant value but with several possible values according to the decision made by the system for the two actuators in play. The rules are described below.

- If the error is negative, then the X-axis actuator moves to the right, and the Y-axis actuator stops.
- If the error is zero, then the X-axis actuator remains in the center, and the Y-axis actuator advances.
- If the error is positive, then the X-axis actuator moves to the left, and the Y-axis actuator stops.

It is important to highlight that these conditions do not generate classic logic responses, but rather combine the opinions of the three elements to produce fuzzy output signals with intermediate values. The results are predictable and

consistent, indicating the maintenance of fundamental principles across situations and applications.

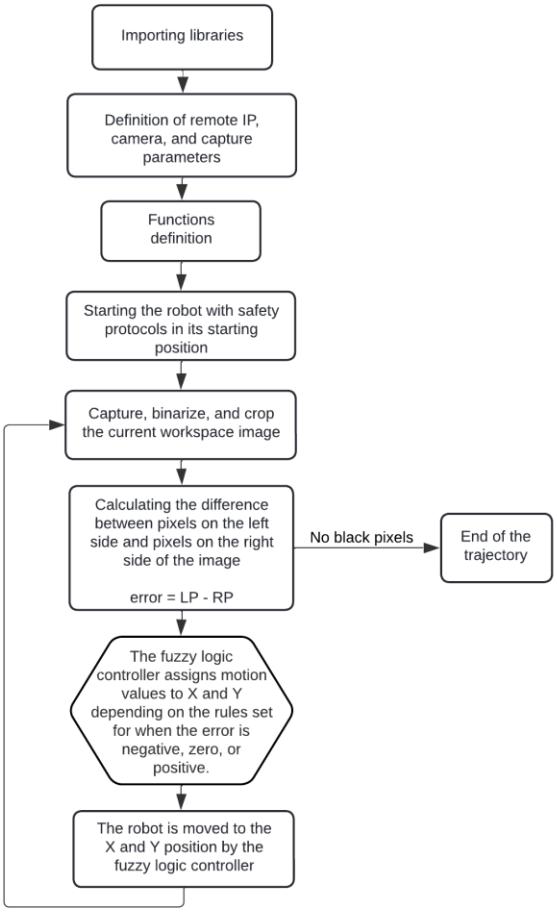


Fig. 4. Fuzzy logic-based active controller flowchart.

C. Convolutional Neural Network Based Active Controller

A Convolutional Neural Network (CNN or ConvNet) is a type of artificial neural network designed to process spatial data, being particularly effective in tasks such as object recognition, image classification, and anomaly detection. CNNs consist of several layers, each with a specific function. These typical layers include [25]:

- Convolutional Layer: Applies filters to the input data to detect specific patterns. The filters are matrices that move across the input data, calculating scalar products between the values of the filter and the input data.
- Pooling Layer: This layer reduces the dimensionality of the output data of the convolutional layer. This is done to reduce the number of parameters the network needs to learn, which can help improve training performance and efficiency.
- Fully Connected Layer: Similar to a traditional layer of artificial neurons, it connects the patterns detected by the convolutional layers and performs the final classification of the data.

CNNs are highly efficient for image processing because they share parameters across different layers. This means that the network only needs to learn a set of parameters for each type of pattern it detects [26].

To train a convolutional neural network for implementation as a control system, a dataset tailored to the network's needs is necessary. This dataset will consist of a series of images captured with the camera mounted on the robotic arm, corresponding to images of lines with different characteristics in shape and size. These images will serve as inputs to the system, representing what the system's sensor can detect. As output data, there will be the movements in X and Y that the robot should ideally make in response to the image captured by the camera. For that reason, a value of X and a value of Y will be needed as a response to each of the images used for the dataset. Instead of arbitrarily assigning these decisions to the robot, which could introduce bias, the dataset will be generated using the fuzzy logic controller discussed earlier. The dataset generation involves the following steps:

- The robot will follow the trajectory of 10 lines with different arbitrarily designed characteristics to express various possible trajectories using the fuzzy logic controller.
- The target curve of the case study in this report will not be used in the dataset generation process, as the goal is to evaluate the system's ability to predict movements needed to follow the curve without being taught during training.
- Each of the 10 training curves will be used multiple times to generate more data for the dataset. To avoid redundancies, duplicate instances that only insert redundancy into the system will be removed during data preprocessing.

It is important to highlight that the 10 training curves used do not represent only 10 images as data in the training set. Remembering that the images obtained by the robot are 770x100 pixels and considering that the system takes an image immediately after obtaining an X and Y position output until finishing the total trajectory, each of the training curves represents an average of 800 images, which means that in the training set of the neural network there are approximately 8000 preprocessed images of 770x100 pixels. In such small capture regions of the images obtained by the camera, this number of images guarantees the generalization of the system against different trajectories. Furthermore, the evaluation of the system is carried out with the curve of the case study that was not included within the 10 training curves.

Both the decisions made by the fuzzy-based algorithm and the input images that generated these decisions in X and Y are stored. It's important to note that, for images, no additional processing is necessary apart from what was already done in the system previously.

Subsequently, to adapt the input and output data of the neural network for training, the following data preprocessing is carried out:

- Removal of duplicate instances.

- Normalization of images to obtain binary intensities of 0 or 1.
- Separation of data into training and evaluation sets, with 20% of the data for evaluating the network.

With the data already prepared according to the previous analysis, it is necessary to create a convolutional neural network model to be trained with all these data to predict the X and Y values that the robot should take in response to any input image. For this, the network topology is followed, in the operational order described below.

a) Input Convolutional Layer: This layer performs convolutions on the image input, extracting features from 3x3 pixel windows. The ReLU function is applied to the outputs of this layer to introduce nonlinearity. It is configured to receive images with a size of 770 x 100 pixels. The 32 neurons at the input will allow the reception of images to convolutional layers. Padding allows the layer to output images with the same dimensionality as they had at the input, filling with zeros if necessary, during the system's training period. Furthermore, it is important to emphasize that the images are binary, so only one channel is needed, saving computational costs, training time, and improving the performance of the training process. The ReLU activation function will in turn deal with images given its specialty with pattern detection in specific regions of the image. Its characteristics are summarized:

- Type: Conv2D
- Neurons: 32
- Filter size: (3, 3)
- Padding: 'same' (filled with zeros to maintain the size of the output as the input if necessary)
- Activation function: ReLU (Rectified Linear Unit)
- Input size: (770, 100, 1)

b) Max Pooling Layer: This layer performs maximum pooling, reducing the spatial resolution of the output from the previous layer by half. It helps reduce the number of parameters and learn more general features. Its characteristics are summarized as follows:

- Type: MaxPooling2D
- Pooling window size: (2, 2)
- Strides: (2, 2)

c) Second Convolutional Layer: This layer performs additional convolutions on the features extracted by the previous layer. It increases the complexity of the learned features. Its characteristics are summarized as follows:

- Type: Conv2D
- Neurons: 64
- Filter size: (3, 3)
- Padding: 'same'
- Activation function: ReLU

d) Second Max Pooling Layer: Reduces the spatial resolution of the output. Its characteristics are summarized as follows:

- Type: MaxPooling2D
- Pooling window size: (2, 2)
- Strides: (2, 2)

e) Flatten Layer: Converts the previous 2D output into a 1D vector, preparing it to connect to dense layers.

f) First Dense Layer: Hidden layer processing the 1D feature vector obtained in the previous layer. Its characteristics are summarized as follows:

- Type: Dense
- Neurons: 128
- Activation function: ReLU

g) Output Dense Layer: The output layer has 2 neurons since this is a regression problem, and the activation function is linear. This means that this layer is expected to produce continuous values, which will be used to predict the X and Y outputs of the system.

- Type: Dense
- Neurons: 2
- Activation function: Linear

The convolutional layers extract features from images, followed by dense layers that perform the final regression. The loss function used is Mean Squared Error (MSE), and the metric is accuracy, indicating how well the model predicts the system's outputs, measuring the percentage of correctness in the prediction. In total, the network has 7 layers, including convolutional layers, pooling layers, dense layers, and an output layer. The number of neurons in each layer is specified above for each type of layer.

Additionally, some hyperparameters of the trained model are briefly described:

- **Epochs:** Training is performed for 10 epochs. An epoch is a complete pass through the entire training set. During each epoch, the model adjusts its weights based on prediction errors and seeks to minimize the loss function.
- **Batch Size:** Batches of data with a size of 128 are used for training. This means that in each epoch, the training set is divided into batches of 128 examples each. The model calculates weight updates after processing each batch, rather than after each individual example. This is beneficial for speeding up training and can help the model converge faster.

After training, the historical evolution of MSE and Accuracy throughout the training is obtained. The history is visualized in Fig. 5, within which the X-axis indicates the number of epochs and the Y-axis shows the value of the metrics obtained.

The history shows how the mean squared error of the model reduced in the second epoch. Similarly, the accuracy increased considerably in the second epoch. It is evident that from the eighth epoch onwards, there is no significant change in either the mean squared error or the accuracy of the model; hence, 8 epochs are considered more than sufficient to achieve similar results. The training lasted approximately 18 minutes, requiring moderately high computational resources for its execution. Using the `evaluate` function of Keras, a mean squared error (MSE) of 0.0008395 and an accuracy of 99.747% are obtained

when evaluating the model with the dataset reserved for this purpose (20%). This implies a comprehensive learning of patterns in the images and the XY outputs to the robot.

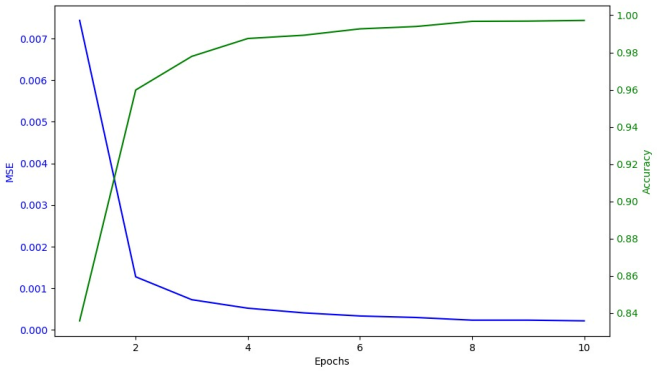


Fig. 5. History of the MSE and the accuracy of model training through the Epochs.

V. RESULTS

In Fig. 6, the curves generated by each of the control algorithms implemented in the six-axis industrial robot can be visualized, in addition to the original tracking curve.

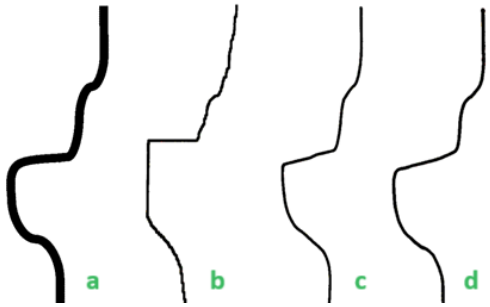


Fig. 6. Tracking curves. a) Original. b) ON/OFF control. c) Fuzzy control. d) Control by CNN. Note: Drawn on A4 standard sheets of paper, 297 mm height and 210 mm width.

In Fig. 7, 8, and 9, the results of the curve comparison are presented, accompanied by overlaid images for visualization.

In Table V, the advantage of the convolutional neural network over the fuzzy controller and the ON/OFF controller can be observed. The convolutional neural network demonstrates a remarkable ability to react more effectively to variations in the original trajectory. In Fig. 10, all the overlaid curves are visualized, with the curve generated by the convolutional neural network being highlighted.

In Table V, as the values of MSE (Mean Squared Error), RMSE (Root Mean Squared Error), and MAE (Mean Absolute Error) approach zero, the similarity between the test curve and the original curve increases significantly. The values of PSNR, SSIM, Jaccard Index, and the Pearson correlation coefficient are metrics that indicate how similar the curves are.

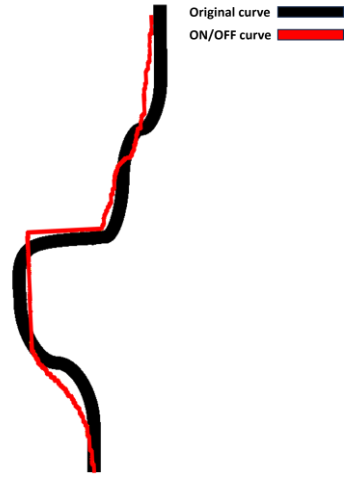


Fig. 7. Comparison between the curve generated by ON/OFF and the original curve.

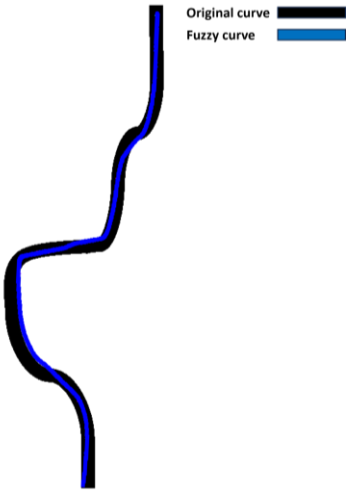


Fig. 8. Comparison between the curve generated by fuzzy and the original curve.

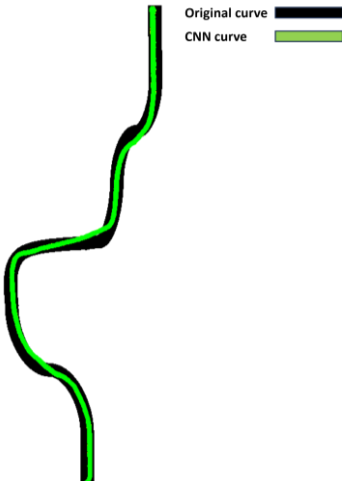


Fig. 9. Comparison between the curve generated by CNN and the original curve.

TABLE V
COMPARATIVE ANALYSIS OF CNN, FUZZY CONTROLLER, AND ON/OFF CONTROLLER

	On/OFF	Fuzzy Logic	CNN
MSE	0,0914	0,0861	0,0838
RMSE	0,3023	0,2934	0,2895
MAE	4,2899	3,5238	3,4817
PSNR	10,3909	10,6494	10,7665
SSIM	0,8895	0,8945	0,8962
Jaccard Index	0,9076	0,9128	0,9148
Pearson Coefficient	0,1835	0,2381	0,2811

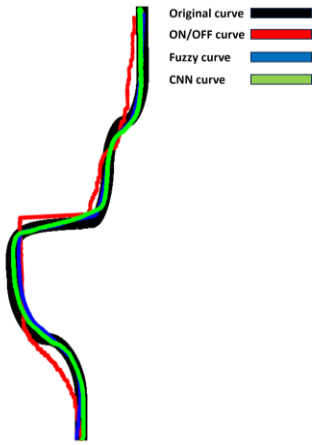


Fig. 10. Comparison between the curves generated by all the control algorithms and the original curve.

In Fig. 11 the robot can be seen following the trajectory from the beginning to the end of the proposed curve.

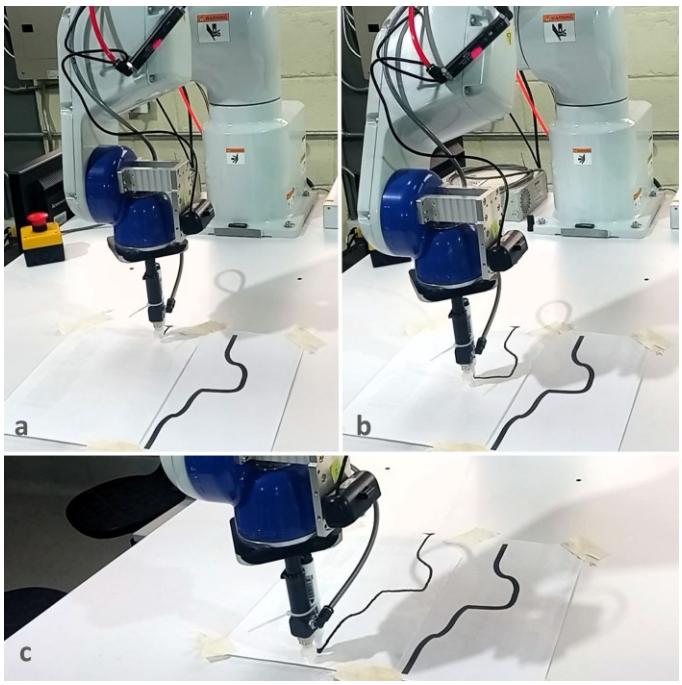


Fig. 11. Trajectory tracking process at a) The beginning b) In the middle of c) The end of the trajectory.

V. CONCLUSIONS

This study presents the comparison of three control methods for trajectory control in an industrial robot. The comparative analysis demonstrates that the implementation of Convolutional Neural Networks (CNN) is essential to improve trajectory tracking in industrial robots. CNNs allow effective processing of visual information, key for high-precision tasks and adaptation to dynamic environments.

The results show that the implementation of Convolutional Neural Networks (CNN) as a control in industrial robots offers significant advantages, such as an improvement in visual perception and trajectory tracking. CNNs demonstrate an exceptional ability to process visual complexities, which is crucial for tasks that require high precision and adaptability to changes in the environment.

Despite the evident benefits, the implementation of convolutional neural networks in industrial robots also presents challenges. The need for large and representative datasets, as well as the complexity in training and tuning these models, poses technical challenges. Additionally, ongoing maintenance and updating of the models to adapt to changes in the industrial environment are critical aspects to consider in long-term implementation.

REFERENCES

[1] B. Singh and J. Sharma, "A review on distributed generation planning," *Renew. Sustain. Energy Rev.*, vol. 76, pp. 529–544, Sep. 2017, doi: 10.1016/J.RSER.2017.03.034.

[2] N. Sadati and R. Ghadami, "Adaptive multi-model sliding mode control of robotic manipulators using soft computing," *Neurocomputing*, vol. 71, no. 13–15, pp. 2702–2710, Aug. 2008, doi: 10.1016/j.neucom.2007.06.019.

[3] H. Wang, "Adaptive Control of Robot Manipulators With Uncertain Kinematics and Dynamics," *IEEE Trans. Automat. Contr.*, vol. 62, no. 2, pp. 948–954, Feb. 2017, doi: 10.1109/TAC.2016.2575827.

[4] A. Ashagrie, A. O. Salau, and T. Weldcherkos, "Modeling and control of a 3-DOF articulated robotic manipulator using self-tuning fuzzy sliding mode controller," *Cogent Eng.*, vol. 8, no. 1, Jan. 2021, doi: 10.1080/23311916.2021.1950105.

[5] R. Kumar, S. Srivastava, and J. R. . Gupta, "Modeling and control of one-link robotic manipulator using neural network based PID controller," in *2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, Sep. 2016, pp. 243–249, doi: 10.1109/ICACCI.2016.7732054.

[6] Z. Cheng and Z. Zhuo, "Adaptive Iterative Learning Trajectory Tracking Control of SCARA Robot," in *2021 IEEE 4th Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*, Jun. 2021, pp. 910–914, doi: 10.1109/IMCEC51613.2021.9482360.

[7] J. Liang, "Research on Industrial Robot Trajectory Tracking Control System based on PID Feedforward Algorithm," in *2022 IEEE 2nd International Conference on Electronic Technology, Communication and Information (ICETCI)*, May 2022, pp. 338–342, doi: 10.1109/ICETCI55101.2022.9832315.

[8] T. Power and D. Berenson, "Learning a Generalizable Trajectory Sampling Distribution for Model Predictive Control," *IEEE Trans. Robot.*, pp. 1–18, 2024, doi: 10.1109/TRO.2024.3370026.

[9] D. Zhang and B. Wei, "A review on model reference adaptive

control of robotic manipulators," *Annu. Rev. Control*, vol. 43, pp. 188–198, 2017, doi: 10.1016/j.arcontrol.2017.02.002.

[10] L. Sciavicco and B. Siciliano, *Modelling and Control of Robot Manipulators*. London: Springer London, 2000.

[11] Z. Wang and P. Keogh, "Active Vibration Control for Robotic Machining," Nov. 2017, doi: 10.1115/IMECE2017-71670.

[12] D. K. Thomsen, R. Søe-Knudsen, O. Balling, and X. Zhang, "Vibration control of industrial robot arms by multi-mode time-varying input shaping," *Mech. Mach. Theory*, vol. 155, p. 104072, Jan. 2021, doi: 10.1016/j.mechmachtheory.2020.104072.

[13] V. Nguyen, J. Johnson, and S. Melkote, "Active vibration suppression in robotic milling using optimal control," *Int. J. Mach. Tools Manuf.*, vol. 152, p. 103541, May 2020, doi: 10.1016/j.ijmachtools.2020.103541.

[14] H. Pan and M. Xin, "Nonlinear robust and optimal control of robot manipulators," *Nonlinear Dyn.*, vol. 76, no. 1, pp. 237–254, Apr. 2014, doi: 10.1007/s11071-013-1123-1.

[15] S. Islam and X. P. Liu, "Robust Sliding Mode Control for Robot Manipulators," *IEEE Trans. Ind. Electron.*, vol. 58, no. 6, pp. 2444–2453, Jun. 2011, doi: 10.1109/TIE.2010.2062472.

[16] E. Olsson, P. Funk, and N. Xiong, "Fault diagnosis in industry using sensor readings and case-based reasoning," *J. Intell. Fuzzy Syst.*, vol. 15, pp. 41–46, 2004.

[17] J. Hollerbach, W. Khalil, and M. Gautier, "Model Identification," 2016, pp. 113–138.

[18] J. Yang, D. Wang, B. Fan, D. Dong, and W. Zhou, "Online absolute pose compensation and steering control of industrial robot based on six degrees of freedom laser measurement," *Opt. Eng.*, vol. 56, no. 3, p. 034111, Mar. 2017, doi: 10.1117/1.OE.56.3.034111.

[19] M. Ruderman, *Analysis and Compensation of Kinetic Friction in Robotic and Mechatronic Control Systems*. Boca Raton: CRC Press, 2023.

[20] S. Zhen, Z. Zhao, X. Liu, F. Chen, H. Zhao, and Y.-H. Chen, "A Novel Practical Robust Control Inheriting PID for SCARA Robot," *IEEE Access*, vol. 8, pp. 227409–227419, 2020, doi: 10.1109/ACCESS.2020.3045789.

[21] P. Santana and L. Correia, "Swarm cognition on off-road autonomous robots," *Swarm Intell.*, vol. 5, no. 1, pp. 45–72, 2011, doi: 10.1007/s11721-010-0051-7.

[22] J. N. Potter, S. A. Neild, and D. J. Wagg, "Generalisation and optimisation of semi-active, on–off switching controllers for single degree-of-freedom systems," *J. Sound Vib.*, vol. 329, no. 13, pp. 2450–2462, Jun. 2010, doi: 10.1016/j.jsv.2009.12.011.

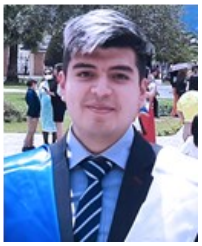
[23] T. Dewi, Y. Wijanarko, P. Risma, and Y. Oktarina, "Fuzzy Logic Controller Design for Leader-Follower Robot Navigation," in *2018 5th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*, Oct. 2018, pp. 298–303, doi: 10.1109/EECSI.2018.8752696.

[24] A. Pandey, R. K. Sonkar, K. K. Pandey, and D. R. Parhi, "Path planning navigation of mobile robot with obstacles avoidance using fuzzy logic controller," in *2014 IEEE 8th International Conference on Intelligent Systems and Control (ISCO)*, Jan. 2014, pp. 39–41, doi: 10.1109/ISCO.2014.7103914.

[25] L. Ran, Y. Zhang, Q. Zhang, and T. Yang, "Convolutional Neural Network-Based Robot Navigation Using Uncalibrated Spherical Images," *Sensors*, vol. 17, no. 6, p. 1341, Jun. 2017, doi: 10.3390/s17061341.

[26] S. Kulik and A. Shtanko, "Using convolutional neural networks for recognition of objects varied in appearance in computer vision for intellectual robots," *Procedia Comput. Sci.*, vol. 169, pp. 164–167, 2020, doi: 10.1016/j.procs.2020.02.129.

José R. Castro Doctor in Electrical Engineering (Applied Engineering) at the Ecole de Technologie Superieure in Montreal, Canada, in 2016. Currently, he is a Research Professor at the Universidad Técnica Particular de Loja, and his field of research is related to Energy and Robotics.



David P. Rosales was born in Loja, Ecuador on June 9, 2000. He holds a degree in Electronics and Telecommunications from the Universidad Técnica Particular de Loja in 2017, corresponding to the bachelor's degree of studies. Currently, he is a thesis student pursuing a master's degree in Artificial Intelligence at the Universidad International of the Rioja in Spain.

Carlos Calderon Cordova (Senior Member, IEEE) is an Electronics and Telecommunications Engineer (UTPL-Ecuador) and a Master in Electromechanics (UNL-Ecuador). His areas of expertise are Digital Transformation of Industry, Industrial Robotics, Automatic Control, and Industrial IoT. He is the Director of the CONSYS-UTPL Research Group and the Coordinator of the LERAP-UTPL Prototyping and Innovation Laboratory. He was Chair of the IEEE Robotics and Automation Society (Ecuador, 2023). He was Chair of IEEE SIGHT (Ecuador, 2020-2021), and the Co-founder and the Executive President of the technology-based company KRADAC (2010-2021). He is the author of 37 Scientific Publications Indexed in Scopus / Web of Science. He has also generated 9 International Patent Applications.