






Using Soft Computing and Computer Vision to Create and Control an Integrated Autonomous Robotic Manipulator Process

João Antônio Toledo Rodrigues¹ ^a, Samuel Dos Anjos¹ ^b, Mateus Coelho Silva^{1,2} ^c,
Ricardo C. Câmara de M. Santos¹ ^d and Ricardo Augusto Rabelo Oliveira¹ ^e

¹Departamento de Computação - DECOM, Universidade Federal de Ouro Preto - UFOP, Ouro Preto, Brazil

²Instituto Tecnológico Vale, Universidade Federal de Ouro Preto - UFOP, Ouro Preto, Brazil

Keywords: Soft Computing, Robotic Manipulator, Object Detection, Evolutionary Algorithm, Inverse Kinematics.

Abstract: The development and control of an integrated autonomous robotic manipulation process requires a focus on the convergence of technologies such as soft computing, object detection, robotic arm engineering, and direct and inverse kinematics. For instance, the inverse kinematics issue can be targeted using soft computing instead of challenging mathematical applications. This paper explores using soft computing systems, an algorithm that produces approximate solutions to complex problems and phenomena. Thus, the use of soft computing proved valid, given the accuracy and speed of the claw. The soft computing technology is based on an evolutionary algorithm that allows us to create several points on a cartesian plane and mix them to implement inverse kinematics. Our results showed that using soft computing, which is different from the traditional way, leads to solid and functional results. The implementation involves integrating Arduino, Raspberry Pi 4.0, a PWM model PCA9685, a camera, and six servo motors to create a robotic arm. The system employs video streaming to transmit data to a local network, where the Raspberry Pi processes RGB to HSV images for object identification. The present work in this paper has experiments for the accuracy and speed of the claw to take a determinate object from the center and the side of the checkered board.


1 INTRODUCTION


Automation and robotics have been playing an increasing role in our society, simplifying complex tasks and performing essential functions in various industries, as Royakkers et al. say in (Royakkers and van Est, 2015). In this context, the value of autonomous robot systems increases daily, and the need to integrate them into our daily lives becomes more apparent. From the manufacturing industry and medicine facilities to the indoors of our houses, autonomous robots are reshaping how we live and work.


In other roles, automation technology dominates daily tasks, with self-driving cars, trucks, drones, machines for modern medical operations, and robot facilities in the industry. In this way, the prospect of a future with technology for cleaning our house, mak-


ing up our beds, storing clothes and utensils is getting closer. In this way, those new kinds of facilities will become crucial for the sake of an easy life. According to Raja et al. (Raja and Nagasubramani, 2018), technology significantly influences how students learn and teachers teach. The introduction of technology into the educational environment has made the teaching-learning process more interactive, efficient, and globalized.


The article also emphasizes using soft computing techniques that are valuable for their ability to handle uncertainty, imprecision and partial truth. In this way, soft computing can achieve and improve challenges and issues in modern agriculture, such as resource management, labor, technology advancements in other industries, product quality, modernization, organic farming and sustainable practices. The article (Yardimci, 2007) by Yardimci et al. reviews the applications of fuzzy logic-neural networks methodology utilized in the clinical science of medicine. Furthermore, the article showed an interest in applying soft computing methodologies in genetics, physiology, radiology, cardiology, and neurology.

^a  <https://orcid.org/0009-0005-3905-3163>

^b  <https://orcid.org/0009-0001-0347-5461>

^c  <https://orcid.org/0000-0003-3717-1906>

^d  <https://orcid.org/0000-0002-2058-6163>

^e  <https://orcid.org/0000-0001-5167-1523>

With that in mind, our work describes the implementation and use of a modern system of a robot arm that performs the catch of an object inside the field of vision of the camera. The image stream comes from a Raspberry Pi working with a Pi Camera. The arm of the robot is made of six servos motors, an Arduino, a Raspberry Pi 4.0, a camera, and a PWM controller, as we can see in Figure 1.

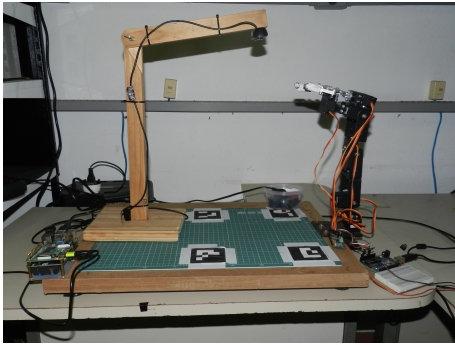


Figure 1: Experimental apparatus used in this work.

In this way, it is possible to create an arm that can perform tasks that are interesting to practitioners of technology or even industrial uses. Thus, the main goal of this work is:

- The employment of soft computing and computer vision techniques to create an autonomous robotic manipulator process.

In the remainder of this work, Section 2 presents the theoretical baseline of this work. Section 3 discusses how this work relates to the literature. Section 4 presents the methodology employed to create the solution, and Section 5 displays the results of the validation experiments. Finally, Section 6 displays the conclusions obtained from this work.

2 THEORETICAL REFERENCES

This section introduces some background that sustains the work. We explored the concepts of soft computing, robotic manipulators, the integration of autonomous robots, and kinematics.

2.1 Soft Computing

Soft computing is system that offers a versatile and adaptable approach to problem-solving, rendering it a valuable asset in tackling intricate, ambiguous, and uncertain challenges across diverse fields such as:

- Decision-Making.
- Optimization in Manufacturing.

- Predictive Maintenance.
- Supply Chain Management.
- Energy Management.
- Quality Control.
- Human-Machine Interaction.
- Smart Infrastructure and Cities.
- Healthcare Diagnostics.
- Financial Forecasting.

The article (Aguado and Cantanhede, 2010) by Aguado et al. talks about fuzzy logic. The logical system introduced by fuzzy logic extends beyond Boolean reasoning, moving beyond the binary notion of merely present or not. Rather than settling for a binary perspective, the aim is to discern the extent to which an element is present or absent. For this, sideways glance fuzzy logic is a mathematical approach that handles uncertainty by allowing for degrees of truth, practical in situations where information is imprecise or ambiguous. In our work, the evolutionary algorithm decides by itself. The evolutionary algorithms are another pillar that sustains the area, as Bartz-Beielstein et al. says in (Bartz-Beielstein et al., 2014). These algorithms refer to a category of population-based stochastic direct search techniques that, in a certain way, imitate the processes of natural evolution. Another cornerstone is the neural networks that Müller et al. say in (Müller et al., 1995). The human brain's architecture inspires these techniques. The computational models are designed for machine learning and artificial intelligence. The artificial neurons, organized in layers, can learn complex patterns and relationships from data.

2.2 Manipulator Process

The manipulation process is deliberately manipulating objects or substances using a robotic manipulator. This specialized mechanical device is intricately designed to execute an array of tasks, encompassing but not limited to handling, lifting, positioning, and moving objects with meticulous precision and control.

Within industrial operations, the manipulator process extends its functionality to tasks such as assembly, packaging, sorting, inspection, and welding. These activities necessitate a high degree of accuracy in controlling the movements of the robotic manipulator. In essence, as Raibert et al. discuss in (Raibert, 1978), the manipulator process signifies the strategic application of a robotic manipulator within specific production or operational contexts, where its dexterity and automated capabilities play a pivotal role in enhancing efficiency and precision.

In this way, the Manipulator process stands as a cornerstone in modern industrial operations, representing a sophisticated approach to the controlled manipulation of materials through robotic manipulators. Mass production shows the highest level of automation in the industry is paramount, as displayed by Krasilnikyants et al. in (Krasilnikyants et al., 2013). In this context, the manipulator process is critical in achieving and maintaining this crucial element. Robotic manipulators, equipped with advanced sensors, actuators, and control systems, contribute to a level of precision unattainable through manual labor alone. However, using computational elements like Raspberry Pi makes it possible to create a whole new world of possibilities.

3 RELATED WORKS

The implementation of the robotic gripper is directly related to an embedded system. In this sense, it is necessary to understand how the elements interact with each other.

3.1 Soft Computing in Robotics

The Use of Evolutionary Algorithms article mentions the use of evolutionary algorithms, specifically for the robotic claw's movement based on the four spots of the checkered board. This soft computing system, named evolutionary algorithms, contributes to the dynamic movement of the robotic arm.

The article (Lopez-Franco et al., 2018) by Lopez et al. shows that inverse kinematics is a crucial factor in allowing a robotic arm to achieve its objective, defining the angle settings applied to each actuator to make this possible. With that here, the following article addresses a possible solution to the inverse kinematics problem using soft computing methods. The article proposes using a CMA-ES algorithm, which presents superior performance among other algorithms evaluated through statistical tests. The results defend an approach used as a promising technique for solving inverse kinematics problems in the movement of robotic arms. The use of the algorithm called CMA-ES makes non-linear and non-convex optimization.

The article (Kumar et al., 2017) by Kumar et al. uses a mathematical way with soft computing to solve inverse kinematics. In this sense, they explored mathematical equations that relate the spatial coordinates of the desired endpoint with the angles of the robot's joints. These equations are derived based on the geometry and kinematic structure of the robot.

Then, they apply soft computing techniques to optimize fuzzy logic controllers. In developing this work, the way to apply inverse kinematics with soft computing did not use complex math manipulation and fuzzy logic.

The article (Chin et al., 2020) by Chin et al. discusses the combination of parameterized analytical models, the Gaussian models, and neural networks (NN). In this way, they apply this technique to improve the control and sensing of soft robotic systems. The innovation is also present in applying soft computing to deal with soft systems' stochastic and non-linear dynamics. Besides that, they used soft computing techniques about control and sensing, using a model that can learn and understand the terrain that is to, but it needs help in modeling and training the system. This way, our work has a different aspect when using the HSV system of vision and soft computing. The facilities to implement and make the code valid in different situations.

3.2 Robotic Manipulator Control

The article (Hock and Sedo, 2018) by Hock et al. presents two main techniques for solving the inverse kinematics problem: analytical and numerical methods. The joint variables are solved analytically in the analytical method according to the given configuration data. In the numerical method, the joint variables are obtained based on numerical techniques. The paper focuses on the analytical solution of the manipulators rather than the numerical solution. On the other hand, our article uses direct kinematics to lead us to a universal solution for the coordinates. As such, as the experiment presents, there is no real difference between the places that the arm is leading to, taking us to a faster response to the arm and keeping the precision.

The article (Chandana et al., 2015) by Chandana et al. proposes a smart surveillance system for home and office security. The system uses Raspberry Pi, a gyroscopic sensor, and a Raspberry Pi camera to detect movement, capture images, and send email alerts to the user. In our article, the data transmission within the system is facilitated through video streaming to a local network, and the Raspberry Pi processes RGB to HSV images. This innovative approach allows for real-time data transmission and processing, contributing to the efficiency of the integrated autonomous robotic manipulator process.

The article (Niloy et al., 2021) by Niloy et al. reviews the main challenges in the design and control of autonomous robots for application in indoor and outdoor environments. The choice of robot components

and models is considered to alleviate some problems related to the movement and perception of robots concerning the environment in which they are located. It also relates to mapping and location, which are essential for the equipment to function correctly. In addition, neural network training techniques used to ensure improvements in robot control and autonomy are mentioned. In this idea, our work uses the system of HSV to perform arm manipulation better and create a model to work in different places.

The article (Deepak et al., 2012) by Deepak et al. uses Denavit Hartenberg parameters to create a mobile arm manipulator. This way, the article uses direct kinematics for the manipulator process based on mathematical formulation. In this way, our article proposes a different approach with inverse kinematics to solve the problem. In this way, we reach the angles of the servo's motor by the final point and not the final point by the angles of the servo's motor. This difference is because this process only requires a little mathematical formulation, as it uses an evolutionary algorithm.

The article by Xu et al. (Xu et al., 2005) used a mathematical formulation similar to the article above but to solve inverse kinematics. In this way, they use a matrix model with Denavit Hartenberg parameters to obtain orientation and the end position of angles. With the matrix to solve, they get the position of joint angles. Our article came with a different approach. It does not require a direct solution for the joint angles and, with the evolutionary algorithm, creates an easier way to get the desired angles for the robotic arm.

4 METHODOLOGY

This work presents soft computing techniques to create an integrated autonomous arm manipulator. The implementation of the project used an Arduino, a Raspberry Pi 4.0, a camera, six servo motors, a checkered plane, and software to receive the information and process it. The idea is to create a robotic arm to recognize targetable objects according to the hue, saturation, and value (HSV) color system. The data is transmitted through video streaming to a local network by installing the Raspberry Pi Cam Web Interface software on the Raspberry Pi. This software is responsible for enabling the transmission and access of camera data over the Wi-Fi network.

From there, the data collected by the camera will be available for access through an IP address provided by the software. The data will then be received on the primary device, which captures the RGB images provided by the camera and makes a transformation to

HSV. These images are then processed by software responsible for converting the data into a checkered plane, which the object identification system will use. After the identification system has read the data, the necessary coordinates for the robotic arm to reach its objective are sent to the Raspberry Pi 4.0 using a USB cable, which will transmit the data to the Arduino. In sequence, the Arduino will act, ensuring the movement of the arms towards the given coordinates.

The arm movement is performed by applying the concepts of direct and inverse kinematics, from which the necessary information for the arm movement is provided, such as the current position of the arm and the possible combinations of angles required at each actuator for the system to reach the desired point. The locomotion of the part is made out by applying the concepts of direct kinematics, which calculates the current position of the arm based on the recorded positions in its joints, and inverse kinematics, which calculates the possible combinations of angles required at each actuator for the system to reach the desired point.

Based on the calculations performed by these two methods, it is possible to precisely control the arm movement, from its departure from the initial point to contact with the target and return to the original position. In this way, direct kinematics deals with computing the position and orientation of the tool reference system in relation to the base system. On the contrary, inverse kinematics involves unraveling the enigma: starting with the desired position and orientation of the end travel, it's about discovering the myriad ways joint angles can join forces to bring that vision to life.

The conversion from RGB to HSV occurs mathematically, and some functions help in the process. The RGB parameters from 0 to 255 are sent to a function that saturates the colors so that the difference between the minimum and maximum values is highlighted. Thus, another function calculates the homography transformation, which leads to finding the angles of the desired figure and highlighting it. In this way, the process ends with highlighting points in the figure to correct and parameterize the object's perspectives.

We start a serial communication with the Raspberry Pi and receive the image from the cameras that are received in the code. Then, the system uses functions to calculate the distance between points and order them. Afterward, with the ArUco tags, the four points are detected. Once the points are located, a perspective transformation is performed on an image based on the tags' reference points.

After defining the points in the gridded space, the

image is segmented to locate objects of interest using RGB and then, for better operation, transformed to HSV. In this sense, the contours of the objects of interest are obtained, and their centers are calculated, which is reread by the system in an infinite loop. Afterward, the actual position of the objectives is calculated based on the ArUco tags.

After pressing the command key, the code sends the pulse to the Arduino, which receives it and passes it to the PWM controller that controls the servo motors so that they mimic an arm picking up an object, which, shortly after, deposits the objective in a predefined space in the code.

4.1 Direct Kinematics

The inverse kinematics problem is built on the problem of how you could calculate angles of $(\theta_0, \theta_1, \theta_2, \theta_3)$ to reach a certain (x, y, z) spot and make the system efficient. In this sense, it is possible to solve the problem with a vector sum and Euclidean distance approach. That is, the vector sum provides, together with the Euclidean distance that the robotic arm system can go from a three-dimensional system to a two-dimensional system. As is evident in the figures Figure 2 and Figure 3.



Figure 2: Two-Dimensional Vector Representation of Arm Configuration.

Direct kinematics is the relation of which angles the servo motors need to be at to the control to have precision. The formulation that describes this modeling is presented in Equations 2 and 1.

$$\begin{aligned}
 y &= v_0 \cdot \sin(\theta_1) \\
 &+ v_1 \cdot \sin(\theta_1 + \theta_2) \\
 &+ v_2 \cdot \sin(\theta_1 + \theta_2 + \theta_3) \\
 &+ v_3 \cdot \sin(\theta_1 + \theta_2 + \theta_3 + \theta_4)
 \end{aligned}
 \tag{1}$$

$$\begin{aligned}
 x &= v_0 \cdot \cos(\theta_1) \\
 &+ v_1 \cdot \cos(\theta_1 + \theta_2) \\
 &+ v_2 \cdot \cos(\theta_1 + \theta_2 + \theta_3) \\
 &+ v_3 \cdot \cos(\theta_1 + \theta_2 + \theta_3 + \theta_4)
 \end{aligned}
 \tag{2}$$

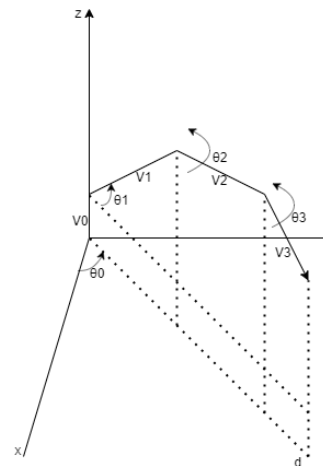


Figure 3: Three-Dimensional Vector Representation of Arm Configuration.

To determine the best functioning of the claw, the following parameters were used to write this article:

- $x = 0 + 104 \times \cos((\pi/2 + t1) + 75 \times \cos((\pi/2) + t1 + t2) + 35 \times \cos((\pi/2) + t1 + t2 + t3) + 165 \times \cos(t1 + t2 + t3)$
- $y = 65 + 104 \times \sin((\pi/2 + t1) + 75 \times \sin((\pi/2) + t1 + t2) + 35 \times \sin((\pi/2) + t1 + t2 + t3) + 165 \times \sin(t1 + t2 + t3)$

4.2 Inverse Kinematics

Inverse kinematics combines the concept of plan dimensions and soft computing to have a method that uses the target coordinates to process and find an answer to position the manipulator tool in this mark. This way, we explored the concepts of evolutionary algorithms to complete the proposal, considering the following steps:

- Population Initialization.
- Aptitude Assessment.
- Selection.
- Crossing over.
- Mutation.
- Replacement.
- Stopping Criterion and Optimal Solution.

To explain further we can see the Figure 4, the next few's paragraphs relate to the upper items.

We find the standard deviation of the range of aggregated data in the repeated expression or field in the chart dimensions. When the standard deviation of fitness is less than 0.01, the population has converged. Another stoppage criterion happens if the fitness standard deviation is less than 0.1, but the system

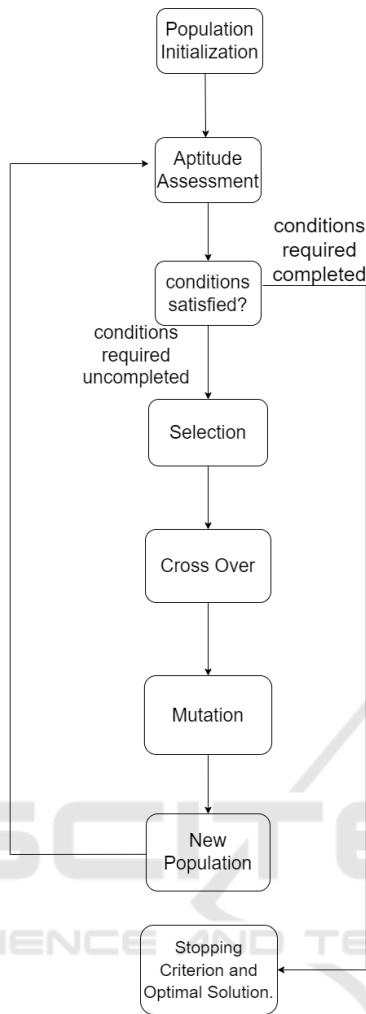


Figure 4: Flowchart of evolutionary algorithm.

has found a solution whose result is sufficiently satisfactory. If this solution is found and the desired point is less than 0.1cm, the solution is good enough. The stages that compose the evolutionary algorithm are:

- **Population Initialization:** Initial generation of random sets of gripper angles. The evolutionary algorithm starts with a population of individuals, each representing a potential solution to the problem. The individual is a set of angles of the robotic arm.
- **Aptitude Assessment:** Inverse kinematics is used to calculate the position of the end-effect with the current joint angles, and the fitness function evaluates how close it is to the desired position. It evaluates how good a solution is in relation to the objective of the problem. The fitness function can measure how close the angles are to the desired position of the robotic arm.
- **Selection:** Individuals are selected proportion-

ally to their fitness (proximity). Individuals with greater fitness are more likely to be selected for reproduction. In the algorithm, this selection mimics the evolutionary process in nature. In this case, the fittest individuals contain the values that best adjust the arm's position relative to the target point.

- **Crossover and Mutation:** Pairs of individuals are combined (crossover), and some individuals mutate. The selected individuals are combined to create offspring. This step involves combining the joint angles of three parents to generate new sets of angles.
- **Replacement:** The descendants and part of the individuals from the previous generation form the next population. That is, the new descendants replace part of the previous population. This stage is a mimicry to select and maintain the best individuals from the previous generation.
- **Stopping Criterion:** The process is repeated until a stopping criterion is reached, be it 0.1 cm or population conversion.

4.3 Validation Experiments

The experiment carried out in this article involved the use of computer vision and soft computing. To this end, two parameters were used for the experiment. One involves the number of hits of the claw to grab the object, and the other is the time required by inference for the claw.

The experiments were carried out with 150 arm tests using soft computing to solve the inverse kinematics at more than one point on the checkerboard. From this perspective, we could observe, note, and detail the algorithm's response time and accuracy. For this matter, the observed variables are:

- Capture success ratio in the central position considering 100 measurements;
- Capture success ratio in the lateral position considering 50 measurements;
- Inference time in central position versus lateral position;

5 RESULTS

This section presents the experiments made in the laboratory to test if the inverse kinematics have different times of responses according to the local that the piece is positioned on the checkered board Figure 5.

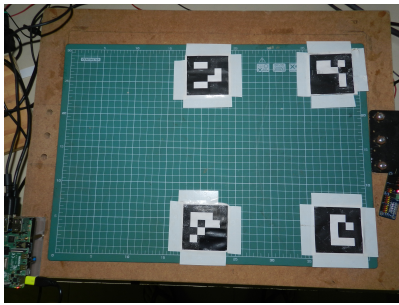


Figure 5: Checkered board.

We separated a red cube to set a condition for the claw to have a better chance to hold it, and we can evaluate the results based on the speed and accuracy of the claw. Initially, it is important to learn that the way the claw moves is based on a soft computing system named evolutionary algorithms that, based on the four spots of the checkered board, makes operations to achieve that coordinate.

With the direct kinematics doing the servo motor angle adjustments, what is needed is the image to use inverse kinematics for the claw to go to the desired point. In this way, the dynamic way that the algorithm transforms the image to the read of RGB is shown in Figure 6. Moreover, the continuous read is shown in Figure 7.

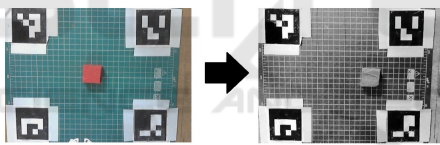


Figure 6: Location of the ArUco tags to separate the working area on the checkerboard.



Figure 7: Continuous camera reading to find the target object.

With that all settled, the red cube is positioned in two ways. In the first and left one, it is positioned right in the center, and in the second and right one, it is on the spot next to the center but distant enough to make a difference.

For the experiments, we took 100 rounds of the claw, taking the red cube in the center, leading us to the result of 92% of catches and 100 measurements of how fast the claw went to take right in the center of the checkered board. For the side positioning, we took 50 rounds of experiments, and we came to the result of 80% of catches and 50 measurements of how fast the

claw went to catch the red cube in a position next to the center. Figure 8 depicts the boxplots containing the results for both stages.

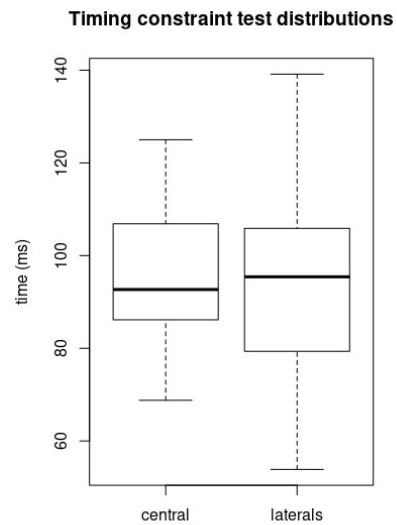


Figure 8: Timing distribution for both tests in a boxplot.

In this way, we evaluate and investigate the hypothesis of the existence of differences between central and lateral positioning concerning the time it takes the algorithm to infer inverse kinematics. We performed statistical tests to compare the two results we had in the experiments.

The variance test evidenced that the two data sets' value of 0.01 indicates no significant difference in variances, suggesting that the variances are equal. After that, we performed the t-test, demonstrating no significant difference between the means of the two groups. The averages are listed as 97ms and 99ms. The conclusion is that the object location makes no significant difference in the average inference time.

Figure 8 depicts the boxplot representing the timing distribution for both tests. Although the image suggests different variances for the situations, the variance tests indicate no significant difference ($p < 0.05$)

6 CONCLUSIONS

The paper describes the development of a robotic arm manipulator solution using soft computing and computer vision. With that approach, the manipulator process became more manageable because it did not require a complex mathematical formulation as the other articles presented in this work. The theory is to create a path in which, even without mathematical modeling, we have a functional and practical inverse

kinematics that is easier to implement.

This work is based on the concept of soft computing. This perspective emphasizes the development of novel solutions without the need for hard mathematical modeling to solve them. Usual tools implemented in soft computing appliances are neural networks, fuzzy logic, and evolutionary computing. In this work, we used the latter to integrate the manipulator application.

The methodology to create and validate the theory that evolutionary algorithms would work for inverse kinematics has reasonable evidence of success, as seen throughout the research. The results come with limitations due to the quality of the claw, which is imprecise. However, as we showed in the experiments, the capture error rate does not give us a real difference, even if the location of the targeted object to be picked up is central or lateral on the checkerboard. The limitation already mentioned above is due to the servo motors having a system that limits them to 180 degrees of rotation and an error, both in the camera view and the precision of the robotic claw, requiring a new investment to change the claw itself.

As shown in related works, the usual way to create inverse kinematics was with mathematical modeling, but in our paper, we used evolutionary algorithms. Future works include using novel elements to integrate a more complex environment. For instance, further steps can integrate a moving treadmill with another camera to detect objects, resulting in the treadmill stopping to remove the object.

ACKNOWLEDGEMENTS

The authors would like to thank FAPEMIG, CAPES, CNPq, Instituto Tecnológico Vale, and the Federal University of Ouro Preto for supporting this work. This work was partially funded by CAPES (Finance Code 001) and CNPq (306572/2019-2).

REFERENCES

- Aguado, A. G. and Cantanhede, M. A. (2010). Lógica fuzzy. *Artigo sem.*
- Bartz-Beielstein, T., Branke, J., Mehnen, J., and Mersmann, O. (2014). Evolutionary algorithms. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 4(3):178–195.
- Chandana, R., Jilani, S., and Hussain, S. J. (2015). Smart surveillance system using thing speak and raspberry pi. *International Journal of Advanced Research in Computer and Communication Engineering*, 4(7):214–218.
- Chin, K., Hellebrekers, T., and Majidi, C. (2020). Machine learning for soft robotic sensing and control. *Advanced Intelligent Systems*, 2(6):1900171.
- Deepak, B., Parhi, D. R., and Amrit, A. (2012). Inverse kinematic models for mobile manipulators. *Caspian Journal of Applied Sciences Research*, 1(13):322–151.
- Hock, O. and Sedo, J. (2018). Inverse kinematics using transposition method for robotic arm. In *2018 ELEKTRO*, pages 1–5. IEEE.
- Krasilnikyants, E. V., Varkov, A. A., and Tyutikov, V. V. (2013). Robot manipulator control system. *Automation & Remote Control*, 74(9).
- Kumar, S., Rani, K., and Banga, V. (2017). Robotic arm movement optimization using soft computing. *IAES International Journal of Robotics and Automation (IJRA)*, 6(1):1–14.
- Lopez-Franco, C., Hernandez-Barragan, J., Alanis, A. Y., and Arana-Daniel, N. (2018). A soft computing approach for inverse kinematics of robot manipulators. *Engineering Applications of Artificial Intelligence*, 74:104–120.
- Müller, B., Reinhardt, J., and Strickland, M. T. (1995). *Neural networks: an introduction*. Springer Science & Business Media.
- Niloy, M. A., Shama, A., Chakraborty, R. K., Ryan, M. J., Badal, F. R., Tasneem, Z., Ahamed, M. H., Moyeen, S. I., Das, S. K., Ali, M. F., et al. (2021). Critical design and control issues of indoor autonomous mobile robots: A review. *IEEE Access*, 9:35338–35370.
- Raibert, M. (1978). Manipulator control using the configuration space method. *Industrial Robot: An International Journal*, 5(2):69–73.
- Raja, R. and Nagasubramani, P. (2018). Impact of modern technology in education. *Journal of Applied and Advanced Research*, 3(1):33–35.
- Royackers, L. and van Est, R. (2015). A literature review on new robotics: automation from love to war. *International journal of social robotics*, 7:549–570.
- Xu, D., Acosta Calderon, C. A., Gan, J. Q., Hu, H., and Tan, M. (2005). An analysis of the inverse kinematics for a 5-dof manipulator. *International Journal of Automation and Computing*, 2(2):114–124.
- Yardimci, A. (2007). A survey on use of soft computing methods in medicine. In *Artificial Neural Networks-ICANN 2007: 17th International Conference, Porto, Portugal, September 9-13, 2007, Proceedings, Part II 17*, pages 69–79. Springer.