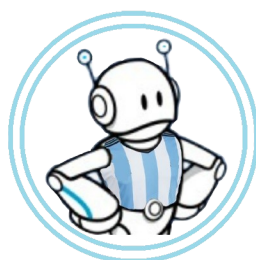


Actas de las Jornadas Argentinas de Robótica (JAR2024)

del 4 al 7 de Junio

Buenos Aires, Argentina

BUENOS AIRES
JAR
2024
del 4 al 7 de junio



Sesión 1: Visión en Robótica

Computer vision solution applied to a fixed-station robot for target tracking

Gustavo Arthur Dutra
Universidade Federal de Santa Maria
Santa Maria - RS, BRASIL
email: gustavo.dutra@acad.ufsm.br

Claudenir Rocha Alves Filho
Universidade Federal de Santa Maria
Santa Maria - RS, BRASIL
email: claudenir.filho@acad.ufsm.br

Ricardo Dias Schirmer
Universidade Federal de Santa Maria
Santa Maria - RS, BRASIL
email: ricardo.schirmer@gmail.com

Anselmo R. Cukla
Universidade Federal de Santa Maria
Santa Maria - RS, BRASIL
email: anselmo.cukla@ufsm.br

Marcelo Serrano Zanetti
Universidade Federal de Santa Maria
Santa Maria - RS, BRASIL
email: marcelo.zanetti@ufsm.br

Gabriel Tarnowski
Universidad Nacional de Misiones
Obera, Misiones, Argentina
email: gabriel.tarnowski@fio.unam.edu.ar

Abstract—This article addresses the development and integration of artificial intelligence (AI) technologies, such as Tracking Learning Detection (TLD) and Google MediaPipe (MP), into a military robotic system known as Robotic Sniper (SR). The SR aims to replace the sniper and observer with cameras equipped with AI algorithms, reducing soldiers' exposure in surveillance missions and increasing operational efficiency. TLD is used for target detection and tracking, while MP is employed for recognizing keypoints of the target's face. The AIs control the robot's motors. The article details the methodology for developing the operator interface software, as well as the integration of TLD and MP technologies, and discusses the results obtained, the system's effectiveness in identifying and tracking targets in different military scenarios. Ethical concerns related to the increasing use of robotic weapons and the importance of maintaining human control over target selection and attack decisions are highlighted.

I. INTRODUCTION

Technological advancement has significantly contributed to the evolution of conflicts throughout the centuries. With the rise of robotics and AI, the military landscape has witnessed an unprecedented transformation. Armed forces are facing the reality of accessible equipment, challenging and sometimes surpassing sophisticated equipment, as seen in current scenarios where low-cost drones can neutralize vehicles, aircraft, or ships whose costs are orders of magnitude higher. This contrast underscores the urgency of adaptation and the importance of quickly incorporating these technologies by the countries or groups involved, as is the case with the conflict between Russia and Ukraine.

Historically, the role of the sniper and observer has been decisive in military operations. As described in the United States Army Sniper Manual (1994) [1], the sniper's mission is to provide precise long-range fire support on selected targets, while the observer assists in locating targets on the ground and determining environmental conditions. However, both are exposed to the risk of being identified and targeted by enemy fire. The development of the Robotic Sniper (SR) can help minimize soldiers' exposure by replacing the sniper and observer with cameras equipped with AI-based algorithms.

Snipers are also employed in hostage situations, and SRs should minimize the risk of injuring hostages by increasing the accuracy of shots through more robust weapon control via a more realistic modeling of local weather conditions.

In the Robotic Sniper (SR), the roles of the sniper and observer are replicated by two cameras integrated into a computer vision system that utilizes AI, such as MP and TLD, respectively, for target detection and tracking. The TLD will be used in the observer's camera to distinguish objects, while the MP will be used in the sniper's camera to accurately determine the center of the eyes for targeting the subject.

When it comes to studies on target tracking based on AI technologies, two notable ones are MP and TLD. According to [2] and [3], MP is a framework developed to enable users to quickly and easily utilize its resources through the Machine Learning (ML) process. This algorithm provides a structure for sensory data inference, performance evaluation tools, and reusable processing components called Calculators, which allow improvements in perception through their language configuration. MP is capable of recognizing various pieces of information, such as people's orientation and limb arrangement, and determines, through a convolutional network, which person deserves the detection layer, prioritizing the one more centrally located in the image.

However, in cases of multiple people on the screen or a cluttered center where facial perception is ambiguous, MP may encounter difficulties in identifying the correct target, which is resolved by using TLD. According to studies [4] and [5], the Region of Interest (ROI) process is used with real-time object tracking, combining technologies such as the CamShift algorithm, OpenCV, and ConvNet. This allows for precise identification of individuals, even in complex situations such as kidnappings, human shields, protests, among others. ROI technologies have a limitation related to defining a generic rectangle that does not specify what is contained within it. This can result in the exclusion of other objects that are not entirely within the area delimited by the rectangle. For example, if a human is within the ROI, the center of the

rectangle will be close to their navel. However, if the person raises an arm, the rectangle will expand, and the center will no longer be aligned with the body. Thus, the combination of these two technologies provides a robust and effective system for detecting and tracking targets in various defense or public security contexts.

Previous studies have explored the use of cameras for gesture recognition purposes to transmit information to control systems, such as prototypes of robotic arms. These investigations aim to achieve more intuitive and efficient control of physical devices, leveraging the keypoint recognition capabilities of MP. The articles [6] and [7] present an approach congruent with the project discussed in this study. Both works focus on identifying single-hand gestures and subsequently transmitting this information to control systems. Although the mentioned studies demonstrate the feasibility of this approach, it is important to note that comprehensive tests have not yet been conducted, especially in scenarios involving additional challenges, such as simultaneous detection of gestures from multiple hands or gestures in unconventional positions.

The technology of TLD can be observed in [8], where a mobile robot is tasked with recognizing and tracking a human. This study utilizes not only TLD but also Median-Flow, another native AI algorithm of OpenCV, and a generic facial recognition algorithm. Here, there was also a concern with scattering components such as the tracked human being in motion or obstructed. The approach of this study is related to the creation of a skeleton of 20 joints. The system appears to resemble the keypoints of MP. Tests demonstrate the technology's ability to continue recognizing the target even after it disappears from the screen and reappears. The study also addresses the high rate of false positives of TLD. The developed technology, called SIFace-TLD, can successfully track the target even when the target is facing away from the camera. Furthermore, the Monte Carlo Particle Filter was adopted to reduce the influence of measurement noise and obtain an accurate estimate of the state for robot motion control. Lastly, a simple and effective controller for human tracking was designed. In the case of SR, the project uses MP to fulfill this need.

On the other hand, robotic systems utilizing the concept of sentinels already exist in the military domain, such as the SGR-A1 developed by Samsung Heavy Industries [9]. It is a robot with autonomous shooting capability, allowing it to identify and shoot targets without human intervention. This robot has various levels of human interaction, operating in both semi-autonomous mode, where the final decision on shooting is made by a human, and fully autonomous mode, where the robot makes independent decisions on when and where to shoot. Its operating interface consists of two main screens, a camera with night vision, thermal imaging, and zoom capabilities.

Additionally, Dodaam Systems Ltd., a company recognized for its leadership in manufacturing military robots, also invests in research, as evidenced in [10], which presents a surveillance system equipped with a target tracking algorithm. This system

utilizes a direction vector calculated from the position error between the center of the viewport and the center of the object in the image captured by the camera. The company is not limited to a static robot but has a range of lethal and non-lethal robots, both static and mobile, with or without humans. It has a control center and a variety of complementary technologies.

In this In this paper, we cannot fail to mention the manifesto by Human Rights Watch [11], in partnership with the International Committee of the Red Cross [12], regarding their concerns about the increasing proliferation of robotic weapons that have the potential to replace humans in combat situations. The SR robot aims to position itself at the safest level of human interaction, known as "Human-in-the-Loop," where robots require human authorization to select targets and inflict damage, following established international norms.

Thus, the study was initiated in 2022, resulting in an SR, as shown in Figure 1, with two axes, equipped with a Taurus 9mm pistol and two cameras, one for terrain vision and another equipped with a scope. The robot has compact dimensions and a modular structure consisting of two distinct parts: a base and a weapon-carrying module. It features two Dynamixel Robotis MX-106R motors, each responsible for a specific axis. In the SR, the functions of the shooter and observer are replicated by two cameras integrated into a computer vision system that utilizes AI, such as MP and TLD, respectively, for target detection and tracking. TLD will be used in the observer's camera to distinguish objects, and MP in the shooter's camera to accurately determine the center of the eyes for targeting the subject. The cameras used in this study are *Webcam Full HD 1080p*, which have autofocus and good contrast, adapting to the lighting conditions.



Fig. 1. Profile image of the SR prototype.

The proposed article aims to explain the operation of the operator interface software, as well as the development and integration of TLD and MP technologies, computer vision algorithms based on artificial intelligence, which control the movements of the robot's motors presented in [13]. The main objective is to explain the algorithms that enable the operation of the interface, where the user can indicate a target, and the robot identifies and tracks it, keeping it centered on the

screen. Specific objectives include how AIs interact with the SR, including the implementation of real-time ROI in the TLD algorithm, integration between the various AI tools employed, the use of keypoints from the MP algorithm for robot motor control, and the development of an operational interface for the user.

This article is structured into four distinct sections. In addition to the introduction, which presents relevant concepts and technologies related to the topic, it includes a detailed methodology on the development and construction of the software. The third section addresses the obtained results and the discussions arising from these results. Finally, the fourth section encompasses the conclusions, highlighting potential future applications related to facial differentiation.

II. METHODOLOGY

This study provides a detailed analysis of the operation of the software and user interface created for the SR, as well as describing the process by which the robot is created, the exchange of Artificial Intelligences (AIs), and finally, the execution of the shot. The entire procedure is described sequentially, as illustrated in the flowchart shown in Figure 2.

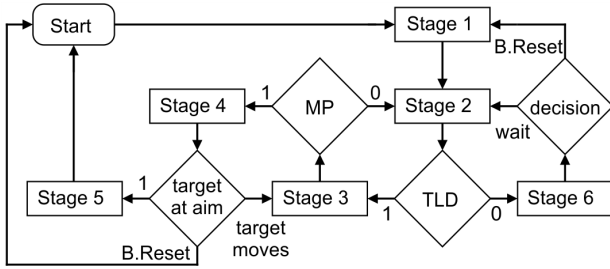


Fig. 2. Steps flowchart.

In the operation of the SR, initially when the system is initialized (Start), the robot is in a surveillance state and responds to the operator's commands, which must be provided via the keyboard and can move it in any direction or activate the ROI algorithm that leads to the first stage. At this moment (Stage 1), on the screen where the user views the actions of the two cameras, the creation of a small green rectangle, controlled by the mouse, around the desired target selected by the mouse is made available, which can be moved or resized to select the target quickly and accurately. After the rectangle is created by the operator and clicked on the target, the TLD comes into action. For better visualization of the beginning of this phase, the rectangle changes from green to blue. The TLD has two internal cycles, search or track. The search is when the target is not detected, and the track is when the target is detected.

The TLD algorithm uses pixel values to transform this information into motor speed, and for this conversion, a mapping function is used, which we call here the MAP function. The pseudocode for this function is presented in Figure 3.

As mentioned, the MAP function is responsible for transforming input values from a pixel displacement range to an output range of velocity displacement, which translates

```

function MAP(refer){
    A ← minimum camera position
    B ← maximum camera position
    C ← minimum engine speed
    D ← maximum engine speed

    vel ← ((refer - A) × (D - C)) / (B - A) + C
    return vel
}
  
```

Fig. 3. Map function.

the information coming from the camera into commands understandable for the motors and was adapted from the C language library of Arduino to Python language. The algorithm uses the reference of the distance between the central point of the target, given by the TLD or MP algorithms, and the central point of the camera, and uses this distance to calculate the velocity.

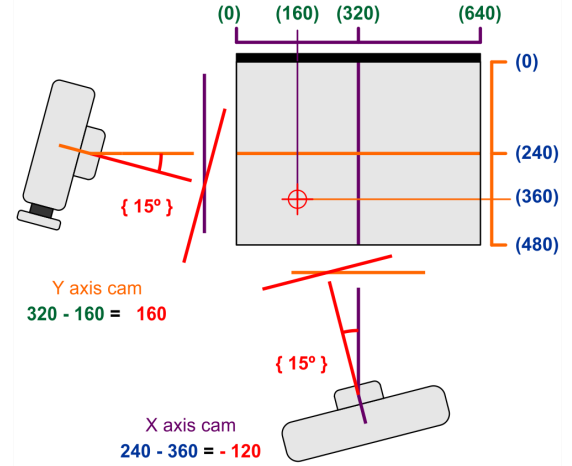


Fig. 4. Capture of Pixels that will be converted by the Map function.

As illustrated in Figure 4, each motor has a different speed limit because their behaviors vary according to the central points of the cameras. The cameras have a resolution of 640x480 pixels; therefore, the central point of the camera for the x-axis is 320, and for the y-axis, it is 240 pixels. Another variant is the x-axis motor, with a movement ratio of 1:1, while for the y-axis motor, there is a movement ratio of 20:1, due to a worm gear. This variation in robot movement is addressed with the Map function and different speed limits on the Dynamixel motors, being for the x and y axes, respectively, 60 and 1000.

Continuing with the flowchart of the software operation methodology of this work, we observe that the second stage (Stage 2) is based on target selection and confirmation. The central x and y coordinates of the rectangle, that is, of the target, are identified. With the use of the Map function, it becomes possible to calculate the motor speeds. The robot will move autonomously and continuously (TLD) until it reaches the central point of the target, adjusting the motor

speeds according to the verification of the distance between the desired central point and the camera's central point, which occurs with the receipt of each new frame. An error rate can be defined as the minimum difference between the central point of the target and the camera's central point. The sixth stage (Stage 6) occurs if the operator decides to turn off the TLD algorithm. This situation may occur if the target is outside the camera's coverage area or if it is erroneously selected by the operator, not corresponding to the desired one.

Subsequently, in the third stage (Stage 3), with the robot centered on the rectangular target of the ROI, the operator has the option to switch the identification algorithm via the keyboard. Thus, the robot will perform a more accurate identification using the MP, which results in the identification of key points on the target's face and body, differentiating yellow points for the left side and green for the right side. In this phase, the displacement of the target in relation to a reference point between the eyes is calculated, allowing the "sniper" camera to control the robot's movement assertively (MP). If disabled or fails to find the target within a timeout of 2 seconds, it will return to Stage 2.

In the fourth stage (Stage 4), the target is in sight, and the robot precisely awaits the operator's decision to interrupt and return to the initial stage or proceed to the final stage of the mission. If the target moves, the MP automatically centers the target in real-time, using the Map function to pass data to the motors. Finally, in the fifth stage (Stage 5), the operator fires, and the robot returns to the initial surveillance position, awaiting new instructions. Thus, completing the cycle that the SR goes through from visual identification to the actual shot, carried out under operator supervision.

The supervision and control of the system are carried out by a designated operator, whose operations are conducted through an interface, as shown in Figure 5. In the image, two main screens stand out: the primary screen for the observer and a secondary screen for the shooter. Given the academic context of this study, the interface is equipped with a series of interactions, including buttons to activate AIs, reset and emergency buttons, firing buttons, as well as a text block for code monitoring, among other essential functionalities.

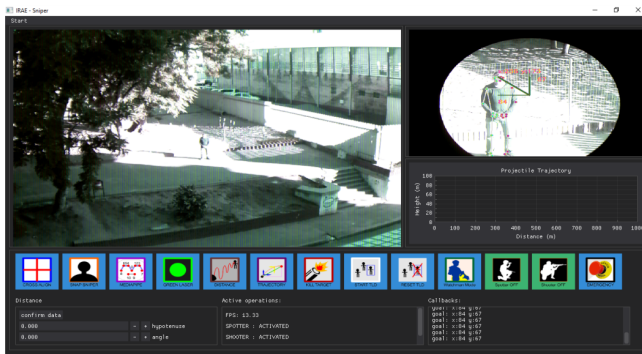


Fig. 5. Interface between operator and SR.

This interface allows observations of the behavior of the AIs



Fig. 6. Result of Stage 1.

and the response of the motors. Analyzing the interactions between the operator and the system can provide valuable insights to optimize operational efficiency and ensure the safety and reliability of operations.

III. RESULTS AND DISCUSSION

As described in the methodology presented in Figure 2, the process begins with the selection of a target by the operator. Then, it proceeds to Stage 1, as illustrated in Figure 6.

The camera shown in Figure 6 has a wide capture area to cover a larger number of objects and people that can be selected as targets. Selecting an area of interest allows the Sniper's camera to provide greater detail of the target. It is noticeable that the image does not have high quality, as the camera was configured for a resolution of 640x480 pixels. Additionally, the image used corresponds to the observer, which does not have an appropriate zoom system.

In Stage 2, it is important to highlight that the area of interest can be created in any area of the image, making it possible to select a non-human object and pursue it. This could be advantageous from the perspective of a broader scope target search. However, the technology developed is aimed at human targets. Therefore, it may be necessary to repeat the target selection stage to proceed with activating the MP for motor control and target pursuit. The result of this stage is illustrated in Figure 7.

Thus, with the use of MP in Stage 3, illustrated in Figure 8, in conjunction with TLD, it is possible to work with great precision. However, the system does not have effective and specific techniques for real distinction between people. This ability could be acquired with the use of other tools that work in conjunction with the OpenCV library, using the calculation of the distance between specific points on the face for differentiation.

Finally, after the transition from target tracking to firing in Stage 4, a laser pointer was used in place of a weapon. When the firing was activated, the laser pointer would activate as well, in order to demonstrate the tests conducted in Stage 5.



Fig. 7. Result of Stage 2.

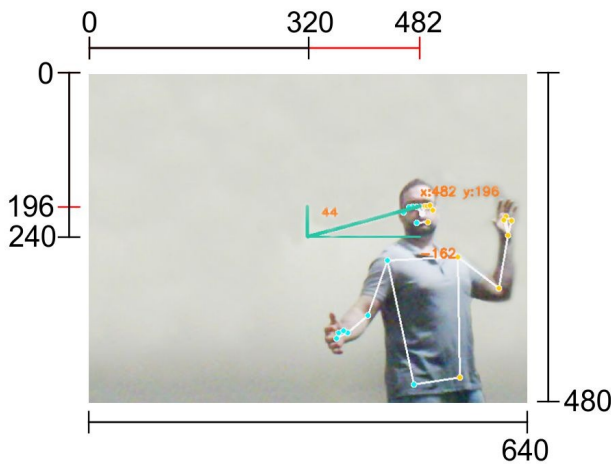


Fig. 8. Result of Stage 3.

Figure 9 demonstrates the test conducted during the laser activation. In the wider image on the left, the observer's view is shown, with the target already selected and centered. On the right is the sniper's view, providing greater detail of the target and the exact position of the sight.

Regardless of its movement, the sight tends to track the target, which can also be used as a powerful tool for intimidation and confusion of the target due to the difficulty in opening the

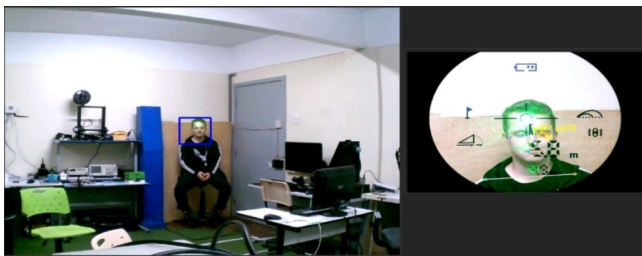


Fig. 9. Result of Stage 5.

eyes considering the location of the light beam.

IV. CONCLUSIONS

The study highlights that the combination of AI and computer vision technologies in the SR represents a promising solution for military operations. The specific results emphasize the real-time use of a ROI converted to the TLD algorithm, as well as motor control and alignment with the target through AI. The operator interface harmonizes with combat robotic equipment, ensuring ethical care in a topic still under discussion.

Future studies focused on optimizing AI algorithms and developing a proprietary combat system are necessary for future advancements in this area. These new studies open the possibility of facial recognition through the replacement or blending of real-time facial recognition technologies with those currently used. Articles such as [2] and [7] demonstrate that better results are obtained with combined AI algorithms.

REFERENCES

- [1] Department of the Army. *Sniper Training*. Washington: U.S. Government Printing Office: 1994—528-027/80156, 1994, p. 329.
- [2] Camillo Lugaresi et al. "MediaPipe: A Framework for Building Perception Pipelines". In: (June 2019).
- [3] Valentin Bazarevsky et al. "BlazePose: On-device Real-time Body Pose tracking". In: (June 2020).
- [4] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. "Tracking-Learning-Detection". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34.7 (2012), pp. 1409–1422. DOI: 10.1109/TPAMI.2011.239.
- [5] Nadja Dardagan et al. "Multiple Object Trackers in OpenCV: A Benchmark". In: Oct. 2021. DOI: 10.1109/ISIE45552.2021.9576367.
- [6] P D Rathika et al. "Gesture Based Robot Arm Control". In: *Nat. Volatiles & Essent. Oil* (2021). DOI: 8(5):3133-3143.
- [7] Muneera Altayeb. "Hand Gestures Replicating Robot Arm based on MediaPipe". In: *Indonesian Journal of Electrical Engineering and Informatics (IJEI)* 11 (Sept. 2023), pp. 727–737. DOI: 10.52549/ijeai.v11i3.4491.
- [8] Jing Yuan et al. "Fusing Skeleton Recognition With Face-TLD for Human Following of Mobile Service Robots". In: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 51.5 (2021), pp. 2963–2979. DOI: 10.1109/TSMC.2019.2921974.
- [9] Mark Prigg. *Who goes there? Samsung unveils robot sentry that can kill from two miles away. SGR-1 has heat and motion detectors to identify potential targets more than 2 miles away Being used in the Demilitarised Zone*. Acessado em 01 jan. 2023. 2014. URL: <https://www.dailymail.co.uk/sciencetech/article-2756847/Who-goes-Samsung-reveals-robot-sentry-set-eye-North-Korea.html>.

- [10] Bong-Cheol Seo, Sung-Soo Kim, and Dong-Youm Lee. “Target-Tracking System for Mobile Surveillance Robot Using CAMShift Image Processing Technique”. In: *Transactions of the Korean Society of Mechanical Engineers A* (2014). DOI: 10.3795/KSME-A.2014.38.2.129.
- [11] Bonnie Docherty. *Losing Humanity. The Case against Killer Robots*. Acessado em 01 jan. 2023. 2012. URL: <https://www.hrw.org/report/2012/11/19/losing-humanity/case-against-killer-robots>.
- [12] CICV. *Armas autônomas: os Estados devem discutir os grandes desafios éticos e humanitários*.) Acessado em 01 jan. 2023. 2013. URL: <https://www.icrc.org/pt/content/armas-autonomas-os-estados-devem-discutir-os-grandes-desafios-eticos-e-humanitarios>.
- [13] Ricardo Dias Schirmer et al. “Project of a Sentinel Robot Controlled with a Tracking Algorithm”. In: *2022 Latin American Robotics Symposium (LARS), 2022 Brazilian Symposium on Robotics (SBR), and 2022 Workshop on Robotics in Education (WRE)* (2022), pp. 241–246. DOI: 10.1109/LARS/SBR/WRE56824.2022.9995943.

Comparisson of different Neural Networks Architectures for Arm Gesture Recognition

Daniel Fernando Tello Gamarra
Electrical Energy Processing
Department (DPEE)
Universidade Federal de Santa Maria
Santa Maria - RS - Brazil

Anselmo Rafael Cukla
Electrical Energy Processing
Department (DPEE)
Universidade Federal de Santa Maria
Santa Maria - RS - Brazil

Solon Bevilacqua
Production Engineering Department
Universidade Federal de Goiás
Goiânia - GO - Brazil

Guillermo Alejandro Mudry
Mechatronic Engineering Department
Universidad Nacional de Misiones
Oberá, Misiones - Argentina

Abstract—The microsoft kinect sensor has open new doors in the field of human machine interaction, one of the features of kinect sensor is its ability of recognizing human skeleton joints. This work explores the use of the recognition human gestures using neural networks. Three different neural networks architectures will be tested in the paper, backpropagation, Radial Basis Function and ANFIS neural networks, experiments and results are shown in order to validate the proposed system for gesture recognition.

I. INTRODUCTION

Technology is improving the way in which humans can interact with machines, an example of this optimization is the microsoft kinect sensor, originally created to be used in videos games, kinect sensor potential for other applications is in development, bringing the possibility of employing the microsoft kinect sensor for a plethora of diverse areas such as commerce, medicine, sales, education, industry, physiotherapy and robotics.

An important part of human machine interaction is gesture recognition, the microsoft kinect offers the possibility of identifying the human body skeleton. The human body skeleton has different joints and in order to identify gestures that are a sequence of movements is necessary to create a model that relates gestures and movements. Different machine learning algorithms have been proposed to build the referred model.

Some recent works have explored the use of Kinect in motion pattern identification. Rimkus in [1] proposes a systems based on a Kinect for hand motion recognition using feedforward neural networks. Sorce in [2] proposed a method based on a Kinect device and neural networks to identify whether a hand is open or close. Zhang in [3] uses a Kinect system based on Hidden Markov Models and Neuro-Fuzzy systems to model the movements adopted by players in Golf. Heickal in [4] uses the Kinect sensor to recognize full body skeleton used by the umpires in a cricket match, backpropagation and Naïve Bayes Classifier are used for gesture recognition. Also some new architectures using deep learning have been applied to the problem as in the works of [12], [13], [15], [?]

This work proposes a system based in neural networks for gesture recognition, the gesture acquisition will be done with a Kinect sensor that will capture the skeleton data, so Multilayer Perceptron, Radial Basis Function and ANFIS neural networks will be employed for the gesture recognition. Besides, the application of these three different neural network architectures for gesture recognition, a comparison of the performance of the neural networks is analyzed. The paper is divided in 6 sections, after a brief introduction in the first section; the second section explains the theoretical background used in the paper; the third section describes the experimental setup; section fourth details the data storage phase and design of the neural networks; section fifth summarizes the results of the system and finally the last section elucidates the main conclusions of the paper and future works.

II. THEORETICAL BACKGROUND

Neural networks have been employed in different ways such as regression, system identification and classification. It will be reviewed the principles and structure of the neural networks that have been used in our experiments.

A. Multilayer Perceptron (MLP)

Exist different kinds of neural networks and their classification is related to their learning algorithms or architecture, a multi-layer perceptron neural network (MLP) will be used, MLP neural networks are classified as networks with a supervised learning algorithm. The MLP has an architecture based in 3 layers, an input layer, a hidden layer and an output layer. In [5] is stated that networks with just two layers of weights are capable of approximating any continuous functional mapping. Based on the work written by Terra in [6] will be described a MLP neural network, for one hidden layer, presenting in the sample n , the input vector $x_n = [x_{1n}, x_{2n}, \dots, x_{pn}]^T$, for a p -dimensional input vector and a q -dimensional output vector, the MLP output of the neuron k (where $k = 1, 2, \dots, q$) is given by:

$$\hat{y}_k = \varphi_k(\sum w_{kj} \varphi_j(\sum w_{ji} x_{in})) \quad (1)$$

Where m is the number of neurons in the hidden layer, w_{kj} is the weight between the neuron j (hidden layer) and the neuron k (output layer), w_{ji} is the weight between the neuron i (input layer) and the neuron j (hidden layer), φ_k is the nonlinear activation function in the output layer and φ_j is the nonlinear activation function in the hidden layer.

B. Adaptive Neuro-Fuzzy Inference System (ANFIS)

As mentioned in [7], an ANFIS is a kind of adaptive network that acts as a framework for adaptive fuzzy inference systems. A fuzzy inference system is a popular computing framework based on the concepts of fuzzy set theory, fuzzy if-then rules, and fuzzy reasoning. The ANFIS neural network will be reviewed here briefly based in [7].

For a first-order Sugeno fuzzy model, a typical rule set with two fuzzy if-then rules can be expressed as:

Rule 1: if x is A_1 and y is B_1
Then $f_1 = p_1x + q_1y + r_1$

Rule 2: if x is A_2 and y is B_2
Then $f_2 = p_2x + q_2y + r_2$

In the structure of a neural network, nodes of the same layer have similar functions, we will denote the output node i in layer l as $O_{i,j}$. Where x and y are referred to any input to the network, and can be any kind of data, they could be image features or atmospherical information, depends of the network application. Every node i in this layer is an adaptive node with a node output defined by:

$$\begin{aligned} O_{1,i} &= \mu_{A_i}(x), \quad \text{for } i = 1, 2, \text{ or} \\ O_{1,j} &= \mu_{B_{i-2}}(y), \quad \text{for } i = 3, 4, \end{aligned} \quad (2)$$

Where x (or y) is the input to the node, and A_i or B_{i-2} is a fuzzy set associated with this node. A_i and B_i can be any parameterized membership function. For example, A_i can be characterized by the generalized bell function:

$$\mu_{A_i}(x) = \frac{1}{1 + \left[\left(\frac{x - c_i}{a_i} \right)^2 \right]^{b_i}} \quad (3)$$

Where a_i, b_i, c_i are the bell function parameters. Training the network consists of finding suitable parameters for the layers, gradient descend methods and least squares methods are used in the training phase.

C. Radial Basis Function (RBF)

These kind of neural networks compute the activation of a hidden unit calculating the distance between the input vector and a prototype vector as stated in [5], the training of these networks is faster than the multilayer perceptron, these networks have a two stage training procedure, in the first stage are calculated the parameters of the radial basis function that are related to the hidden neurons, this first stage uses an unsupervised training technique, in the second stage are calculated the final weights of the layer. The radial basis function approach introduces a set of N basis functions, one for each point, that takes the form $\phi(\|x - x^n\|)$ where ϕ is some non-linear function, and depends of the Euclidean



Fig. 1. Kinect Sensor (image taken from [9]).

distance between x and x^n , where x and x_n are input vectors, the output of the mapping is then represented as a linear combination of the basis function:

$$h(x_n) = \sum w \sum w_n \phi(\|x - x^n\|) \quad (4)$$

and its matrix form is represented as:

$$\Phi \mathbf{w} = \mathbf{t} \quad (5)$$

t , represents the network targets, if exists the inverse matrix, we can obtain:

$$\mathbf{w} = \Phi^{-1} \mathbf{t} \quad (6)$$

There are many kinds of basis functions but the most common is the Gaussian:

$$\phi = \exp\left(-\frac{x^2}{2\sigma^2}\right) \quad (7)$$

III. EXPERIMENTAL SETUP

The kinect sensor will be employed in the paper, the kinect sensor is manufactured by the microsoft, the sensor was projected initially as a complementary device for the xbox videogames platform, the kinect sensor was created for the Jew company primesense, the kinect sensor has a camera and a sensor that is emitter and receiver of infrared rays and a microphones array, these different sensors generate the possibility of working with RGB digital images and the images depth information as is stated in [8]. Figure 1, shows the microsoft kinect sensor, that is the hardware equipment used for the paper.

Processing is a flexible software sketchbook [10], a program written in Java and using this software platform is employed to capture the human skeleton, the program will let us capture the skeleton data and store it. It was used as an initial program the code developed by Bryan Chung in [11], this program was modified in order to store the data that is of our interest and adapt it to our experiments this program uses processing and the library kinect4WinSDK. In Figure 2 it is possible to observe 4 subfigures obtained with the kinect, the first subfigure corresponds to the skeleton image, the second subfigure is the RGB image, the third subfigure of the left above is a segmented image of the person, the fourth subfigure is the depth image. The Matlab toolbox of neural networks will be used for the experiments of training and testing the captured data.

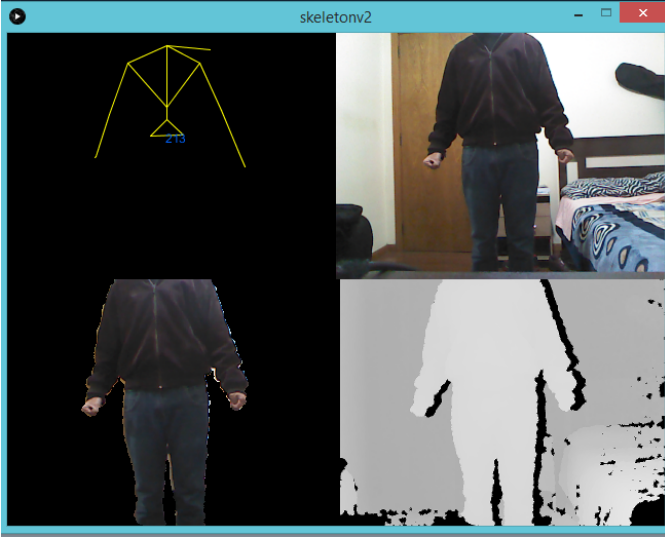


Fig. 2. Skeleton image, RGB and RGBD images capture with the microsoft Kinect sensor.

IV. GESTURE DATA COLLECTION AND NEURAL NETWORK DESIGN

The RGBD images taken from Kinect were captured using the program developed in the software processing, the joint data captured from the skeleton was stored. The joint positions chosen for the network input are the elbow, wrist and shoulder positions in coordinates x, y and z . The data captured in the experiments was integrated by 3840 samples. The data was divided in testing and training data, 2560 samples were chosen as training data and 1280 samples were chosen as testing data.

Afterwards, The Matlab toolbox of neural networks will be used for the experiments of training and testing the captured data. The training and sample data was normalized. The multilayer perceptron (MLP) neural network used 10 hidden neurons, the neural network will have 9 inputs corresponding to the wrist, elbow and shoulder positions in the coordinates x, y and z and one output that will be the gesture generated. The ANFIS neural network also used the same inputs and outputs for its architecture construction, and it constructed its membership functions with Gaussians. Table 1 shows the ANFIS neural network parameters constructed for the experiments.

TABLE I
ANFIS NEURAL NETWORK PARAMETERS

Parameter	Value
Number of nodes	92
Number of linear parameters	40
Number of nonlinear parameters	72

The radial basis function (RBF) neural network will employ the same number of inputs (9 inputs) based on the arm joints and one output based on the kind of gesture executed by the person. Six different gestures were captured with four

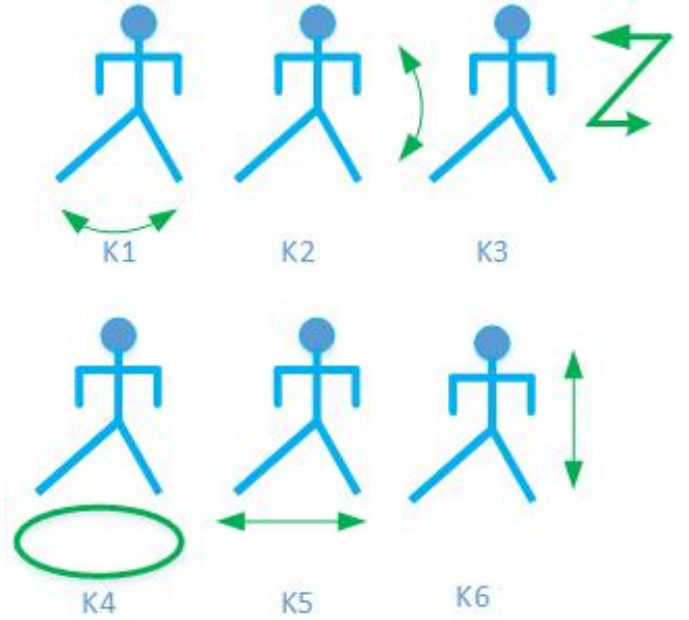


Fig. 3. Arm gestures trajectories .

different human subjects of different ages; the subject ages ranges were between 9 to 74 years old. Figure 3 shows the different gestures executed by the subjects, the gestures are labeled as $K1, K2, K3, K4, K5$ and $K6$ and encrypt linear and circular trajectories as shown in the figure.

Gestures Defined: $K1$ Gesture: Waving the arm in a horizontal plane. $K2$ Gesture: Waving the hand in a vertical plane. $K3$ Gesture: Drawing the letter "Z". $K4$ Gesture: Moving in a circular trajectory. $K5$ Gesture: Following a vertical line trajectory. $K6$ Gesture: Following a horizontal line trajectory.

V. EXPERIMENTAL RESULTS

The results of the experiments with the neural networks are shown in this section. Figure 4 shows the multilayer perceptron neural network mean square error, it is possible to observe that the error is very low and the system will reach a solution in less than 12 epochs, the error will converge to less than 0.0001. The radial basis function neural network converges to a mean squared error of 0.00005 using 50 neurons, with a spread constant of 0.2 as it can be observed in figure 5.

The same experiments and data were used with the ANFIS neural network, Figure 6 shows that also the ANFIS neural network will converge to a solution after a finite number of epochs. Table number 2 resumes the different mean squared errors obtained with the neural network architectures used in the paper. Also table 3 shows the number of epochs employed for every neural network to converge to a determined error.

The results shown in the experiments indicate that the three different neural architectures can be used to the task of gesture recognition with the kinect sensor. However, the networks have different performances. If the mean squared error is analysed , the network with the minor mean squared error

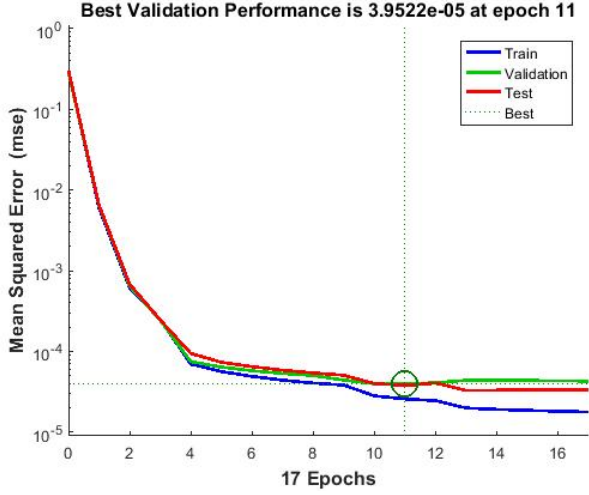


Fig. 4. Multilayer perceptron mean squared error.

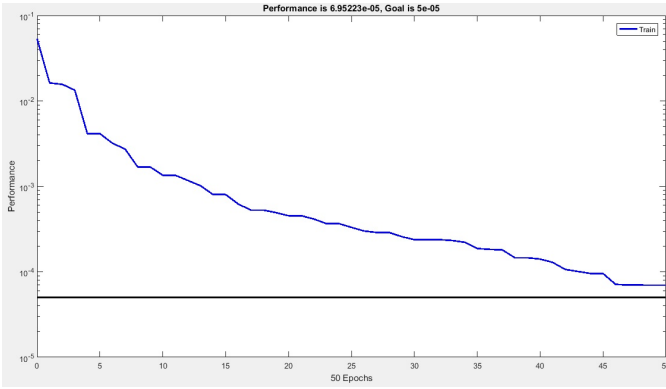


Fig. 5. Radial basis function mean square error.

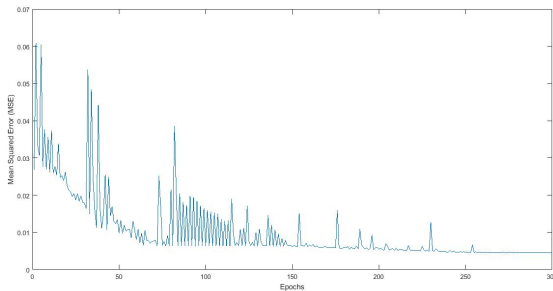


Fig. 6. Anfis neural network error.

TABLE II
NEURAL NETWORK-MEAN SQUARE ERROR

Neural Network	MSE
Multilayer Perceptron	0.000039522
RBF	0.0000695
ANFIS	0.00455096

TABLE III
NEURAL NETWORK-EPOCHS NUMBER

Neural Network	Epochs
Multilayer Perceptron	11
RBF	50
ANFIS	300

is the Multilayer Perceptron, followed by the Radial Basis Function and the ANFIS network. If the focus of the analysis is the number of epochs to reach a determine error, the Multilayer Perceptron uses just 11 epochs, followed by the Radial Basis Function with 50 epochs and the ANFIS neural network with 300 epochs.

Based on these results, we could summarize that the Multilayer Perceptron is the one the shows the best performance, followed by the Radial Basis Function network in second place, and the ANFIS network in third place. Also the Multilayer Perceptron and Radial Basis Function networks have a simpler architecture compared to the ANFIS network that shows a more complex architecture and more number of parameters.

VI. CONCLUSIONS

This work shows the use of three different neural architectures for the task of gesture recognition, the data was capture using the microsoft kinect sensor, the skeleton data was identified and the skeleton joint data was extracted from the RGBD images generated with the Kinect sensor. Three different gestures of a person were used and the neural networks were trained. Results shown that both the Multilayer Perceptron, Radial Basis Function and ANFIS neural networks have a good performance for the task of gesture recognition in that order. But, the Multilayer Perceptron shows a better performance. The next step will be to use the gesture identification system developed in this work for the control of a mobile robot, that will recognize the gestures generated by a human and will move forward, backward or follow a predefined trajectory according to the hand movements.

REFERENCES

- [1] K. Rimkus and A. Bukis and A. Lipnickas and S. Sinkevicius, *3D Human Hand Motion Recognition*, In 6th International Conference on Human System Interactions(HSI), pp. 180-183. IEEE Press, Sopot, Poland, 2013.
- [2] S. Sorce and V. Gentile and A. Gentile, *Real-time Hand Pose Recognition Based on Neural Network Using Microsoft Kinect*, In 8th International Conference on Broadband, Wireless Computing, Communication and Application, pp. 344-350. IEEE Press, Compiegne, France, 2013.

- [3] L. Zhang and J. Hsieh and J. Wang, *A Kinect-Based Golf Swing Classification System Using HMM and Neurofuzzy*, In 8th International Conference on Computer Science and Information Processing (CSIP), pp. 1163-1166. IEEE Press, Shaanxi, china, 2012.
- [4] H. Heickal and T. Zhang and M. Hasanuzzaman, *Real-Time 3D Full Body Motion Gesture Recognition*, In IEEE International Conference on Robotics and Biomimetics (ROBIO), pp. 798-803. IEEE Press, Shenzhen, china, 2013.
- [5] C. Bishop, *Neural Networks for Pattern REcognition*, Evolutionary Computation, 2012.
- [6] M.H. Terra and R. Tinos, *Fault detection and isolation of robotic systems using a multilayer perceptron and a radial basis function*, In Proc. of the IEEE Conference on Systems, Man and Cybernetics, pp. 1880-1885. IEEE Press, San Diego, California, USA, 1998.
- [7] J.S.R. Jang and C.T. Sun, *Neuro-Fuzzy Modeling and Control*, In Proc. of the IEEE , vol. 83, pp. 378-406, 1995.
- [8] L. Cruz and D. Lucio and L. Velho, *Kinect and RGBD Images: Challenges and Applications*, In: Conference on Graphics, Patterns and Images Tutoriais (SIBGRAPI-T) , pp. 36-49, Ouro Preto, Brazil, 2012.
- [9] Smoothing Kinect Depth Frames in REal-Time. <http://www.codeproject.com/Articles/317974/KinectDepthSmoothing>
- [10] Processing. <https://www.processing.org>
- [11] Bryan Chung <https://www.magicandlove.blog/research/kinect-for-processing-library/>
- [12] M. Pfitscher and D. Welfer and M.A.D.L. Cuadros and D.F.T. Gamarra, *Activity Gesture Recognition on Kinect Sensor Using Convolutional Neural Networks and FastDTW for the MSRC-12 Dataset*, In: Intelligent Systems Design and Applications (ISDA) , pp. 230-239, 2020.
- [13] J.S. Peixoto and A.R. Cukla and M.A.D.L. Cuadros and D. Welfer and D.F.T. Gamarra, *Gesture Recognition using FastDTW and Deep Learning Methods in the MSRC-12 and the NTU RGB+D Databases*, In: IEEE Latin America Transactions , pp. 2189-2195, v. 20, 2022.
- [14] J.S. Peixoto and A.R. Cukla and M.A.D.L. Cuadros and D. Welfer and D.F.T. Gamarra, *Comparison of Different Processing Methods of Joint Coordinates Features for Gesture Recognition with a CNN in the MSRC-12 Database*, In: Intelligent Systems Design and Applications (ISDA) , pp. 590-599, 2021.
- [15] J.S. Peixoto and A.R. Cukla and D. Welfer and D.F.T. Gamarra, *Comparison of Different Processing Methods of Joint Coordinates Features for Gesture Recognition with a RNN in the MSRC-12*, In: Intelligent Systems Design and Applications (ISDA) , pp. 498-507, 2022.

Hacia una Detección de Surcos Utilizando Visión y Odometría de Ruedas

1st Pablo Alonso

Departamento de Ciencias de la Computación

FCEIA-UNR

Rosario, Argentina

pabloalonsoxx@gmail.com

2nd Ernesto Kofman

CIFASIS

CONICET-UNR

Rosario, Argentina

kofman@cifasis-conicet.gov.ar

3rd Taihú Pire

CIFASIS

CONICET-UNR

Rosario, Argentina

pire@cifasis-conicet.gov.ar

Resumen—La robótica agrícola viene siendo impulsada en los últimos años como consecuencia de los avances en las distintas áreas que la componen y de la necesidad de optimizar el trabajo rural. Con el fin de automatizar las actividades agrícolas resulta indispensable que los robots autónomos sean capaces de circular por el campo sin pisar o dañar el cultivo. En general, los cultivos se caracterizan por tener una estructura lineal y se disponen en paralelo, formando hileras, por lo que los métodos de navegación deben poder encontrar estos patrones y guiar al robot por los surcos (espacios entre hileras de cultivo), permitiendo al robot circular a través del campo sin dañar el cultivo.

Este trabajo presenta un método de detección de surcos basado en visión y odometría de ruedas. El mismo consiste en identificar las hileras de cultivo de manera de utilizarlas para la navegación a través de los surcos del campo. El método desarrollado es validado tanto en secuencias de simulación como secuencias reales en un campo de soja. Los resultados muestran que el método es suficientemente robusto como para evitar detectar surcos distintos a los que se venían siguiendo aun cuando el robot se desvíe de los mismos.

Keywords—Detección de Surcos, Navegación Autónoma, Odometría de Ruedas, Agricultura de Precisión, Robótica agrícola

Sesión 2: Localización, Mapeo y Planificación de trayectorias I

Seguimiento Robusto de Características Visuales para Sistemas SLAM

Hernán Galarði
Universidad Nacional de Rosario
FCEIA-UNR
Rosario, Argentina
hgalarði@fceia.unr.edu.ar

Gastón Castro
Universidad de San Andrés
CONICET-UDESA
Buenos Aires, Argentina
gcastro@udesa.edu.ar

Taihú Pire
Laboratorio de Robótica
CIFASIS (CONICET-UNR)
Rosario, Argentina
pire@cifasis-conicet.gov.ar

Resumen—

En este trabajo se presenta un método de seguimiento robusto de características visuales, el cual utiliza una ventana deslizante de frames, *brute-force matching*, un sistema de votación para gestionar candidatos en reidentificaciones, una estrategia de selección de candidatos que emplea búsqueda de consensos por similitud de descriptores, y una etapa de validación de correspondencias reidentificadas, para construir un *front-end* versátil y adaptable para sistemas SLAM. El método tiene la capacidad de seguir *características visuales* en condiciones desafiantes, como por ejemplo frente a breves oclusiones o movimientos bruscos. Se presentan resultados que validan el método logrando reidentificar más del 50 % de características visuales que KLT (*Kanade-Lucas-Tomasi feature tracker*) pierde en la secuencia de calibración inicial. La capacidad del *front-end* para reidentificar *características visuales* previamente detectados lo convierte en un módulo útil para ser incorporado en aplicaciones futuras que involucren sistemas SLAM.

Keywords—Características Visuales, Optical Flow, Localización, SLAM

SLAM Bioinspirado con Neuronas Tipo Spiking Memristivas

1st Bernardo Manuel Pirozzo

INTELYMEC, Centro de Investigaciones
en Física e Ingeniería del Centro
UNICEN – CICpBA – CONICET
B7400JWI, Olavarría, Argentina
berpirozzo@gmail.com

2nd Mariano De Paula

INTELYMEC, Centro de Investigaciones
en Física e Ingeniería del Centro
UNICEN – CICpBA – CONICET
B7400JWI, Olavarría, Argentina
mariano.depaula@fio.unicen.edu.ar

3rd Sebastián Aldo Villar

INTELYMEC, Centro de Investigaciones
en Física e Ingeniería del Centro
UNICEN – CICpBA – CONICET
B7400JWI, Olavarría, Argentina
svillar@fio.unicen.edu.ar

4th José Fernández León

INTIA, Centro de Investigaciones
en Física e Ingeniería del Centro
UNICEN – CICpBA – CONICET
B7000JWI, Tandil, Argentina
jafernandez@intia.exa.unicen.edu.ar

5th Gerardo Gabriel Acosta

INTELYMEC, Centro de Investigaciones
en Física e Ingeniería del Centro
UNICEN – CICpBA – CONICET
B7400JWI, Olavarría, Argentina
ggacosta@fio.unicen.edu.ar

Resumen—La incorporación de Inteligencia Artificial (IA) en robótica permitió disponer de capacidades cognitivas que mejoraron sus desempeños en múltiples aplicaciones. A partir de investigaciones de la Neurociencia, se sabe que la información es procesada a nivel neuronal mediante impulsos eléctricos, que luego son transportados a las neuronas vecinas mediante procesos electroquímicos. Este descubrimiento inspiró el desarrollo de las redes neuronales artificiales, y entre ellas, las denominadas Redes Neuronales Spiking (RNS). Con el propósito de implementar RNS a nivel *hardware*, con circuitos electrónicos cuyo comportamiento sea equivalente al de las estructuras neurobiológicas del sistema nervioso, en este trabajo presentamos el modelo *Leaky Integrate-and-Fire*, que es ampliamente utilizado en RNS. La capacidad de memoria necesaria para almacenar pesos sinápticos luego de los entrenamientos se implementó con memristores. Este resistor con memoria es considerado el cuarto elemento pasivo de la electrónica. Hemos seleccionado como caso de estudio para probar estas ideas, un sistema de localización y mapeo simultáneos (SLAM) de inspiración biológica, conocido como Rat-SLAM, para ser evaluado en un robot móvil autónomo (RMA). En efecto, la investigación del cerebro de los roedores, permitió conocer la existencia de zonas encargadas de la localización y orientación espacial, que aprovecharemos en los módulos de guiado y navegación del RMA. Para ésto, en este trabajo proponemos un tipo de RNS, las *Memristive Leaky Integrate-and-Fire* (MLIF) para conformar las *pose cells* del Rat-SLAM.

Palabras clave—Rat-SLAM, Memristores, Robótica Móvil Autónoma, Neurociencias, Computación Neuromórfica.

I. INTRODUCCIÓN

Los robots actuales disponen de capacidades cognitivas que les permiten aprender a resolver con éxito diversas tareas, ampliando por ende, sus capacidades de aplicación. Esta importante cualidad es resultado de la incorporación de la Inteligencia Artificial (IA) a la robótica. La IA, si bien tiene un origen bioinspirado, no procesa la información

de manera análoga al sistema nervioso de los seres vivos. Este último conformado por el cerebro, la médula espinal y las redes de células nerviosas sensitivas o motoras, llamadas neuronas, ha sido ampliamente investigado por la Neurociencia. De esta manera se sabe que la información tratada en el sistema nervioso es codificada en el tiempo o frecuencia de impulsos eléctricos. Estos son generados por las neuronas en respuesta al estímulo de entrada que reciben. El resultado del procesamiento de la información recibida se transmite mediante la liberación de neurotransmisores. Estos son “mensajeros químicos” que transportan la información de las neuronas presinápticas a las postsinápticas, propagando o no así la información en toda la red de neuronas.

La forma biológica de comunicación neuronal inspiró el desarrollo de distintos modelos de redes neuronales artificiales. Uno de ellos son las Redes Neuronales Spiking (RNS) [1], cuya forma de procesar la información no se produce en cada ciclo de propagación, sino que transmiten información sólo cuando el potencial de membrana de la neurona alcanza un valor específico llamado umbral. Cuando esto ocurre, la neurona genera un pulso que se transmite a otras produciendo un aumento o disminución de sus potenciales en respuesta a esta señal. Posteriormente, el potencial de membrana de la neurona activada se restablece a su valor predeterminado.

La Computación Neuromórfica (CN) es una forma de realizar cálculos con elementos de procesamiento cuyo soporte *hardware* imita las neuronas biológicas. Esta forma de implementación tiene la ventaja de reducir el esfuerzo de cómputo necesario para resolver ciertos problemas, como en nuestro caso, localización y mapeo simultáneo (SLAM). Uno de los modelos de neuronas más utilizados debido a su similitud con el comportamiento de las biológicas son las de tipo *spiking*. Estas son implementadas con el circuito *Leaky*

Integrate-and-Fire (LIF) [2], formado por una resistencia y un capacitor vinculados entre sí a través de una conexión eléctrica del tipo paralelo, motivo por el cual, se lo conoce como circuito RC paralelo. El nivel de activación es dado por la ecuación diferencial que modela el comportamiento circuital en función de las entradas a la neurona.

Una solución muy adoptada consiste en reemplazar la resistencia del modelo LIF por un memristor [3], dando lugar al modelo *Memristive Leaky Integrate-and-Fire* (MLIF) [4]. El memristor es considerado el cuarto elemento de la electrónica y fue propuesto por León Chua en 1971. Su nombre es una contracción de “memory resistor (resistor con memoria)”, por lo que dispone de la capacidad de variar y recordar su último valor de resistencia sin consumo de energía, que lo hacen ideal para aquellas aplicaciones donde el consumo es un punto crítico de implementación.

Para probar este tipo de CN con MLIF, en este trabajo presentamos una implementación de SLAM bioinspirado en el cerebro de los roedores. Los estudios realizados por las Neurociencias revelaron la existencia de ciertas zonas cerebrales conocidas como *células de lugar o place cells* [5], *células de dirección de la cabeza o head direction cells* [6], y *células de grilla o grid cells* [7]. Estas zonas reciben señales sensoriales tales como visión, olfato, entre otras, para localizar al animal en un entorno nunca visitado y recordar la localización de lugares por donde ya ha pasado, con lo que se conoce como mapa de experiencias. Este tipo de ubicación espacial, inspiró el desarrollo de sistemas que permitan resolver el problema de SLAM en la Robótica Móvil Autónoma (RMA). Esta aproximación es conocida como Rat-SLAM [8].

Para que la capacidad de memoria intrínseca de las redes neuronales implementadas tenga además características de mínimo consumo de energía, y posibilidad de implementación en un microchip, en este trabajo proponemos una *Pose Cell Memristive Spiking* (PCMS), para conformar una red de células nerviosas o red neuronal artificial con circuitos MLIF, que implemente las *pose cell* del Rat-SLAM [8], como se explicará en la sección IV.

II. MODELO DE MEMRISTOR

El modelo de memristor utilizado es el propuesto por HP Labs [9]. Físicamente es un elemento pasivo de dos terminales formado por dos electrodos de platino que contienen entre sí un material de longitud D formado por dos capas. Una de ellas se compone de dióxido de titanio (TiO_2) altamente resistivo, y la otra de dióxido de titanio pobre en oxígeno (TiO_{2-x}) altamente conductor. Cuando se hace circular una corriente eléctrica en una dirección, el límite entre los dos materiales w se desplaza, produciendo un aumento de la conductividad ocasionado por un mayor porcentaje de la capa TiO_{2-x} . Mientras que, cuando la corriente eléctrica fluye en la dirección contraria a la anterior, se produce un aumento de la resistencia del memristor ocasionada por el desplazamiento de w que aumenta el porcentaje de la capa TiO_2 . De lo anterior se deduce que, cuando no existe circulación de corriente,

w no cambia y el memristor mantiene su estado, es decir, que “recuerda” el último valor de resistencia adoptado. El comportamiento del memristor antes mencionado es dado por la Ec.(1), donde R_{OFF} es el valor de resistencia máxima que se puede tener cuando $w(t) = 0$, y R_{ON} es el mínimo valor de resistencia correspondiente a la condición $w(t) = D$. Es importante notar que, en el instante inicial $t = 0$, $w(0)$ tendrá un valor entre $[0, D]$, por lo cual el valor del memristor ($M(0)$) se encontrará en el intervalo $[R_{OFF}, R_{ON}]$.

$$M(t) = R_{OFF} + (R_{ON} - R_{OFF}) \frac{w(t)}{D} \quad (1)$$

La velocidad de cambio de w , considerando un modelo de deriva lineal con movilidad media de las deficiencias de oxígeno μ_v , es dado por la Ec.(2). Por lo cual, integrando con respecto al tiempo, se obtiene el valor de la ubicación del límite entre las dos zonas del memristor w , como se muestra en la Ec.(3). De esta manera, sustituyendo la Ec.(3) en la Ec.(1), se llega a la Ec.(4), donde $k = (R_{ON} - R_{OFF}) \left(\frac{\mu_v R_{ON}}{D^2} \right)$.

$$\frac{dw}{dt} = \frac{\mu_v R_{ON}}{D} i(t) \quad (2)$$

$$w(t) = \frac{\mu_v R_{ON}}{D} q(t) + w(0) \quad (3)$$

$$M(t) = M(0) + kq(t) \quad (4)$$

Despejando la carga eléctrica $q(t)$ de la Ec.(4) y sabiendo que los valores límites que puede adoptar del memristor son R_{ON} y R_{OFF} , es posible observar la relación existente entre estas dos magnitudes, como se muestra en la Ec.(5). Luego, teniendo en cuenta que la variable k siempre es menor a cero, y que el valor del memristor es una función monótona decreciente de la carga eléctrica, se puede decir que el modelo de memristor controlado por la carga eléctrica es un función continua que se calcula con la Ec.(6).

$$\underbrace{\frac{R_{OFF} - M(0)}{k}}_{c_1} \leq q(t) \leq \underbrace{\frac{R_{ON} - M(0)}{k}}_{c_2} \quad (5)$$

$$M(t) = \begin{cases} R_{OFF} & \text{si } q(t) < c_1 \\ M(0) + kq(t) & \text{si } c_1 \leq q(t) < c_2 \\ R_{ON} & \text{si } q(t) \geq c_2 \end{cases} \quad (6)$$

Por otra parte, es posible relacionar el flujo con la carga eléctrica mediante la Ec.(7). Con lo cual se obtiene una expresión que vincule flujo con carga eléctrica, tal como se muestra en la Ec.(8). Despejando el flujo de esta última expresión, obtenemos la Ec.(9), que sustituyéndola en la Ec.(6), permite obtener las ecuaciones que rigen el comportamiento del modelo de memristor controlado por el flujo, tal como se muestra en la Ec.(10), donde los valores de las constantes c_3 a c_6 son dados por las Ec.(11).

$$\varphi(t) = \int_0^t M(t)i(t)dt = \int_{q(0)}^{q(t)} M(t)dq(t) \quad (7)$$

$$\varphi(t) = \begin{cases} R_{OFF}q(t) + c_3 & \text{si } q(t) < c_1 \\ \frac{k}{2}q^2(t) + M(0)q(t) & \text{si } c_1 \leq q(t) < c_2 \\ R_{ON}q(t) + c_4 & \text{si } q(t) \geq c_2 \end{cases} \quad (8)$$

$$\varphi(t) = \begin{cases} \frac{\varphi(t)-c_3}{R_{OFF}} & \text{si } \varphi(t) < c_5 \\ \frac{\sqrt{2k\varphi(t)+M^2(0)}-M(0)}{k} & \text{si } c_5 \leq \varphi(t) < c_6 \\ \frac{\varphi(t)-c_4}{R_{ON}} & \text{si } \varphi(t) \geq c_6 \end{cases} \quad (9)$$

$$M(t) = \begin{cases} R_{OFF} & \text{si } \varphi(t) < c_5 \\ \sqrt{2k\varphi(t)+M^2(0)} & \text{si } c_5 \leq \varphi(t) < c_6 \\ R_{ON} & \text{si } \varphi(t) \geq c_6 \end{cases} \quad (10)$$

$$\begin{aligned} c_3 &= -\frac{[R_{OFF}-M(0)]^2}{2k} \\ c_4 &= -\frac{[R_{ON}-M(0)]^2}{2k} \\ c_5 &= \frac{R_{OFF}^2-M^2(0)}{2k} \\ c_6 &= \frac{R_{ON}^2-M^2(0)}{2k} \end{aligned} \quad (11)$$

En este trabajo se utilizó el modelo del memristor controlado por flujo con los valores paramétricos propuestos en [4]. De esta manera, los valores adoptados para las variables son $R_{OFF} = 20000\Omega$, $R_{ON} = 100\Omega$, $M(0) = 16000\Omega$, $\mu_v = 1e^{-14}m^2s^{-1}V^{-1}$ y $D = 10nm$.

III. MODELO NEURONAL

Una de las primeras representaciones de las neuronas biológicas propuso un elemento de procesamiento (EP) que realiza la suma de sus entradas ponderadas. A esta suma la hace atravesar un función no lineal o función de activación, y si supera cierto umbral predefinido (v_{th}), hace que el EP dispare un impulso, en un típico comportamiento de *on-off*. Luego del disparo, la actividad del EP permanece refractaria a nuevas excitaciones durante un tiempo, y su potencial es reiniciado a un valor establecido (v_{reset}), tal y como sucede con las neuronas estudiadas en la biología por McCulloch y Pitts [10]. Este modelo no tiene en cuenta el tiempo, es decir, no tiene una noción de cuándo ocurren las señales de entrada. La información se almacena en el valor de los pesos intersinápticos, que proporciona una plasticidad enorme para modelar una transferencia de las señales de entrada al impulso de salida. Imaginemos qué pasaría si pudiésemos incorporar información temporal a las señales de entrada a la neurona. La información podría almacenarse en el tiempo o frecuencia en la que suceden los impulsos o *spikes* a lo largo del tiempo. De esta forma, la actividad de la neurona se caracterizaría por su generación de *spikes* en momentos determinados, como una respuesta a las entradas que está recibiendo el EP. Y de esta forma, la comunicación entre neuronas en este modelo se realizaría a través de estos *spikes*, lo que permite una codificación temporal más rica y eficiente en comparación con los modelos basados únicamente en activación binaria. Estas consideraciones tiene en cuenta el modelo de redes neuronales *spiking* (RNS), presentado en [2]. Vale decir, mientras que el modelo de McCulloch-Pitts es una simplificación binaria del comportamiento neuronal, las neuronas de tipo *spiking* son más sofisticadas y tienen en

cuenta el tiempo y la naturaleza de las señales neuronales, lo que las hace más adecuadas para modelar el funcionamiento real de las redes neuronales biológicas. Por lo tanto, este último modelo proporcionó una gran comprensión sobre la codificación de la información en las redes neuronales, la memoria, su comportamiento dinámico y, más recientemente, el aprendizaje profundo. Las RNS pueden implementarse en *hardware* mediante un circuito RC, en el que se reemplaza al resistor por un memristor, conocido como el modelo de neurona MLIF, [4]. En la Fig.1, se muestra el circuito de la neurona, donde I_{ext} es la corriente de excitación que atraviesa la membrana de la neurona desde su interior hasta su exterior, C es un capacitor de valor igual a $9e^{-7}$ Faradios, M es un memristor con las características explicadas en la sección anterior, y v_{th} es la tensión de umbral ajustada a 20mV. Cuando el valor de la tensión en el memristor supera a este umbral de 20mV, el interruptor se cierra reiniciando el potencial de la neurona a 0V, y luego vuelve a abrirse hasta que la condición anterior vuelva a cumplirse.

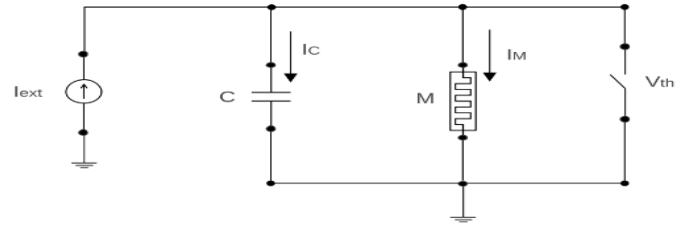


Fig. 1. Modelo de neurona MLIF utilizado.

La ecuación diferencial que modela el comportamiento de este tipo de neurona se obtiene de la siguiente manera. Sabiendo que la corriente externa es igual a la suma de la corriente que circula a través del capacitor y la que circula a través del memristor, y a su vez, sabiendo que la primera obtenida de multiplicar el valor del capacitor por la derivada de la tensión respecto del tiempo, y la segunda de dividir la tensión aplicada por el valor del memristor, se obtiene la expresión mostrada en la Ec.(12). Luego, la ecuación diferencial que rige el comportamiento de la neurona se obtiene despejando la derivada de la tensión respecto del tiempo, como se muestra en la Ec.(13), donde $\tau = M(t)C$. Finalmente, para resolver esta ecuación diferencial se utilizó el método de las diferencias finitas, como se muestra en la Ec.(14), donde Δt es el período de muestreo, cuyo valor adoptado en este trabajo es $1e^{-3}$ s.

$$I_{ext} = C \frac{dV(t)}{dt} + \frac{V(t)}{M(t)} \quad (12)$$

$$\frac{dV(t)}{dt} = \frac{1}{\tau} (-V(t) + I_{ext}M(t)) \quad (13)$$

$$V(t + \Delta t) = \frac{\Delta t}{\tau} (-V(t) + I_{ext}M(t)) + V(t) \quad (14)$$

IV. RAT-SLAM CON PCMS

El algoritmo de Rat-SLAM se programó en base a las PCMS. Los aportes de nuestro trabajo son los siguientes:

- Resolver un problema bien definido de la RMA con un sistema inspirado en el comportamiento del cerebro de los roedores utilizando un modelo de neurona de impulsos.
- Codificar las poses del robot en las frecuencias o tiempos de estos impulsos o *spikes*.
- Posibilidad de implementar las *pose cell* en un dispositivo de *hardware*, disminuyendo los requerimientos de esfuerzo de cómputo requerido para la operación del sistema.
- Disponer de una *pose cell* con memoria intrínseca, que aprenda a relacionar un nivel de excitación a cada unidad con una pose del robot.

En el diagrama de bloques de la Fig.2 se muestra esquemáticamente el funcionamiento del Rat-SLAM implementado. Para poder operar, el módulo de percepción del entorno debe proporcionar imágenes monoculares en escala de grises, velocidad lineal, velocidad angular y rumbo. Esta información proviene de una cámara monocular y un odómetro (*encoder*), montados sobre el RMA terrestre, que fue la plataforma de pruebas experimentales de este trabajo. Las *Local View Cell* reciben la imagen y el centroide de actividad de la PCMS para establecer una relación entre ambas. Este módulo está formado por celdas que almacenan una plantilla visual para cada uno de los lugares visitados. Cuando se ve una escena familiar, la celda correspondiente a esa vista se activa, caso contrario una nueva celda es creada. Estas plantillas visuales se obtienen utilizando el perfil de intensidad de la imagen en escala de grises. Cuando la diferencia entre los perfiles almacenados y el actual es menor que un umbral, la plantilla actual no se almacena, caso contrario, se crea una nueva. La comparación entre el perfil actual I^j y los almacenados I^k se realiza utilizando la función de diferencia de intensidad absoluta promedio como se muestra en la Ec.(15), donde f es dada por la Ec.(16) en la que ω es el ancho de la imagen. Esta comparación se lleva a cabo en un pequeño rango de desplazamientos de píxeles s para proporcionar cierta robustez frente a la rotación.

$$k_m = \arg \min_k f(s, I^j, I^k) \quad (15)$$

$$f(s, I^j, I^k) = \frac{1}{\omega - |s|} \left(\sum_{n=1}^{\omega - |s|} |I_{n+\max(s,0)}^j - I_{n-\min(s,0)}^k| \right) \quad (16)$$

La PCMS es un arreglo 3D de *grid cells*, cuya dinámica es la de Redes de Atractores Continuos (RAC) [11]. Tiene una forma prismática toroidal de tres dimensiones. Dos dimensiones corresponden a la posición en x e y , y la tercera dimensión al rumbo o *heading* del robot. Cada una de las neuronas componentes de estas redes se modelan utilizando el modelo explicado en la sección III, donde el vínculo mediante conexiones excitadoras e inhibitoras es

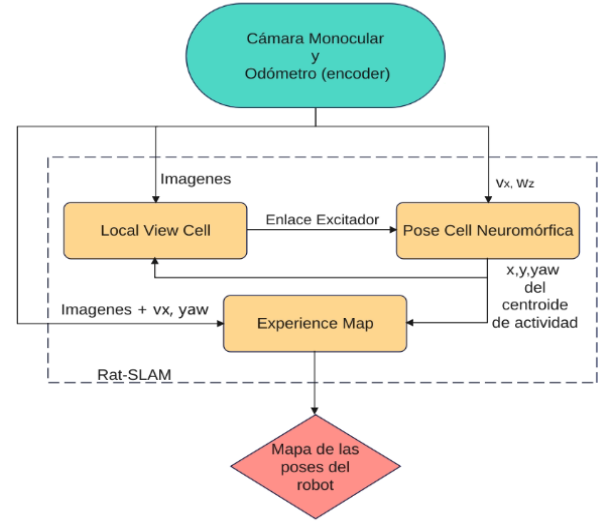


Fig. 2. Diagrama de bloques y flujo de información del Rat-SLAM.

dado por la corriente de excitación que ingresa a cada una, permitiendo que en cada posible estado, un único grupo de unidades se encuentren activas. De esta manera, se establece el denominado paquete de actividad o paquete de energía, cuyo centroide codifica la estimación de la pose actual del robot. La actividad en la PCMS es gobernada por las tres etapas de la dinámica de RAC 3D. Estas son: excitación local, inhibición local-global y normalización de la actividad. La primera de ellas, necesita una matriz de pesos de excitación modelados por distribuciones Gaussianas tridimensionales, como se muestra en la Ec.(17). En esta expresión, var_{eje} es la varianza de cada eje coordenado, constantes para la distribución espacial. El cambio en la corriente de excitación es dado por la Ec.(18), donde $n_{\hat{x}}$, $n_{\hat{y}}$ y $n_{\hat{\theta}}$ son las dimensiones de la matriz de actividad, y $\varepsilon_{a,b,c}$ es la multiplicación de la distribución Gaussiana de cada eje.

$$\varepsilon_j = \left(\frac{1}{var_{eje} \sqrt{2\pi}} \right) e^{-j^2 / 2var_{eje}^2} \quad (17)$$

$$\Delta i^{(exc)} = \sum_i^{n_{\hat{x}}} \sum_j^{n_{\hat{y}}} \sum_k^{n_{\hat{\theta}}} P_{\hat{x}, \hat{y}, \hat{\theta}}^{(t)} \cdot \varepsilon_{a,b,c} \quad (18)$$

Los índices a , b y c , representan las distancias entre las coordenadas $\hat{x}$, $\hat{y}$ y $\hat{\theta}$, y se calculan con las expresiones mostradas en Ec.(19).

$$a = (\hat{x} - i) \cdot \text{mod}(\text{dim}(\hat{x})) \quad (19)$$

$$b = (\hat{y} - j) \cdot \text{mod}(\text{dim}(\hat{y}))$$

$$c = (\hat{\theta} - k) \cdot \text{mod}(\text{dim}(\hat{\theta}))$$

Obtenida la corriente de excitación, se calcula la corriente de inhibición mediante una matriz de pesos inhibitorios $\psi_{a,b,c}$ modelados de la misma forma que $\varepsilon_{a,b,c}$, pero con valores de pesos negativos, tal como se muestra en la Ec.(20).

$$\Delta i^{(inh)} = \sum_i^{n_{\hat{x}}} \sum_j^{n_{\hat{y}}} \sum_k^{n_{\hat{\theta}}} P_{\hat{x}, \hat{y}, \hat{\theta}}^{(t)} \cdot \psi_{a,b,c} \quad (20)$$

Con estos dos procesos completados, la corriente que se aplicará a cada neurona de la PCMS en el siguiente tiempo de muestreo es dada por la Ec.(21). Como consecuencia de aplicar una corriente a cada neurona, se obtiene el potencial de la membrana celular de cada una de ellas. Cada potencial representa la matriz de actividad de la PCMS. A esta se le debe restar un valor constante para realizar la inhibición global, y finalmente se la normaliza.

$$i^{(t+1)} = \Delta i^{(exc)} + \Delta i^{(inh)} \quad (21)$$

Hasta aquí, el proceso de actualización de la actividad en la PCMS ha completado los pasos de la dinámica del atractor 3D, sin tener en cuenta la percepción del entorno. Para incorporar esta información a la actualización se realizan dos pasos. El primero es *path integration*, y el segundo, *local view calibration*. El *path integration* proyecta la actividad de las celdas 3D a las cercanas desplazando el paquete de actividad en el plano $(\hat{x}, \hat{y})$ de acuerdo con la velocidad de translación lineal, y también se moverá en la en función de la velocidad angular para representar el rumbo $(\hat{\theta})$. En la Ec.(22) se muestra el cambio en la matriz de actividad debido a la percepción del entorno, donde $\delta_{\hat{x}_0}$, $\delta_{\hat{y}_0}$ y $\delta_{\hat{\theta}_0}$, se calculan de acuerdo a la Ec.(23), con $k_{\hat{x}}$, $k_{\hat{y}}$, y $k_{\hat{\theta}}$ representado los offset iniciales para cada eje, y γ se calcula de acuerdo con la Ec.(24), donde $\delta_{\hat{x}_f}$, $\delta_{\hat{y}_f}$ y $\delta_{\hat{\theta}_f}$ son los offset finales dados por la Ec.(25).

$$P_{l,m,n}^{(t+1)} = \sum_{x=\delta_{\hat{x}_0}}^{\delta_{\hat{x}_0}+1} \sum_{y=\delta_{\hat{y}_0}}^{\delta_{\hat{y}_0}+1} \sum_{\hat{\theta}=\delta_{\hat{\theta}_0}}^{\delta_{\hat{\theta}_0}+1} \gamma \cdot P_{(l+x),(m+y),(n+\theta)}^{(t+1)} \quad (22)$$

$$\begin{aligned} \delta_{\hat{x}_0} &= k_{\hat{x}} \cdot v \cdot \cos(\hat{\theta}) \\ \delta_{\hat{y}_0} &= k_{\hat{y}} \cdot v \cdot \sin(\hat{\theta}) \\ \delta_{\hat{\theta}_0} &= k_{\hat{\theta}} \cdot \omega \end{aligned} \quad (23)$$

$$\gamma = f(\delta_{\hat{x}_f}, \hat{x} - \delta_{\hat{x}_0}) f(\delta_{\hat{y}_f}, \hat{y} - \delta_{\hat{y}_0}) f(\delta_{\hat{\theta}_f}, \hat{\theta} - \delta_{\hat{\theta}_0}) \quad (24)$$

$$f(a, b) = \begin{cases} a & \text{if } b = 1 \\ 1 - a & \text{if } b = 0 \end{cases}$$

$$\begin{aligned} \delta_{\hat{x}_f} &= k_{\hat{x}} \cdot v \cdot \cos(\hat{\theta}) - \delta_{\hat{x}_0} \\ \delta_{\hat{y}_f} &= k_{\hat{y}} \cdot v \cdot \sin(\hat{\theta}) - \delta_{\hat{y}_0} \\ \delta_{\hat{\theta}_f} &= k_{\hat{\theta}} \cdot \omega - \delta_{\hat{\theta}_0} \end{aligned} \quad (25)$$

Por otra parte, la calibración de vista local o *local view calibration* es un proceso cuyo objetivo es restablecer los errores acumulativos de la integración del camino. Para ello, las *local view cell* están asociadas con la PCMS de modo que cuando el robot se posiciona en un lugar ya visitado, la *local view cell* asociada a esta ubicación activa, por medio de

un enlace de excitación, las mismas PCMS que se activaron cuando esta ubicación se visitó por primera vez. Para aprender la relación entre el vector de las *local view cell* y la actividad en la PCMS, la matriz ψ almacena sus interconexiones sinápticas, utilizando una versión modificada de la ley de Hebb para el aprendizaje. La conexión entre $P_{\hat{x}, \hat{y}, \hat{\theta}}$ y V_i se obtiene mediante la Ec.(26), donde ξ es el learning rate y V_i es el vector de actividad de las *local view cell*.

$$\psi_{i, \hat{x}, \hat{y}, \hat{\theta}}^{t+1} = \max(\psi_{i, \hat{x}, \hat{y}, \hat{\theta}}^t, \xi V_i P_{\hat{x}, \hat{y}, \hat{\theta}}) \quad (26)$$

El mapa de experiencias o *experience map* es un mapa topológico que se compone de muchas experiencias individuales, e , cada una de ellas conectadas por transiciones, t . Una experiencia individual e_i , dada por la Ec.(27), contiene la posición de las celdas activas en la PCMS ($P_{\hat{x}, \hat{y}, \hat{\theta}}^i$), el vector de actividad de la *local view cell* (V^i), la posición espacial ($\mathbf{p}^i$) estimada a partir de las medidas de odometría. La primera experiencia se crea en un punto de inicio arbitrario, y las experiencias subsiguientes se construyen a partir de la primera a través de transiciones.

$$e_i = \{P_{\hat{x}, \hat{y}, \hat{\theta}}^i, V^i, \mathbf{p}^i\} \quad (27)$$

Cuando la métrica de comparación de la actividad en los dos módulos anteriores supera un umbral S_{max} , se crea una nueva experiencia, con una transición asociada. La métrica S se obtiene con la Ec.(28), donde μ_p y μ_v son constantes que ponderan las respectivas contribuciones de la PCMS y la *local view cell* a la puntuación.

$$S = \mu_p |P_{\hat{x}, \hat{y}, \hat{\theta}}^i - P_{\hat{x}, \hat{y}, \hat{\theta}}| + \mu_v |V^i - V| \quad (28)$$

La transición t_{ij} almacena el cambio de posición medido a partir de la integración de las velocidades proveniente de la odometría. Como la velocidad de traslación debe estar descompuesta en el plano, es necesario el rumbo brindado por la odometría para llevar a cabo esta operación. En la Ec.(29) se muestra las transiciones almacenadas, donde $\Delta \mathbf{p}^{ij}$ es el cambio en la pose del vehículo estimada. El vínculo entre la experiencia previa e_i y la nueva experiencia e_j se forma según la Ec.(30).

$$t_{ij} = \{\Delta \mathbf{p}^{ij}\} \quad (29)$$

$$e_i = \{P_{\hat{x}, \hat{y}, \hat{\theta}}^j, V^j, \mathbf{p}^i + \Delta \mathbf{p}^{ij}\} \quad (30)$$

En este módulo, el cierre de bucle ocurre cuando la actividad en la PCMS y *local view cell* coinciden lo suficiente con una experiencia almacenada. Cuando esto ocurre, es muy poco probable que el cambio sumado en la posición de las transiciones conduzca a la experiencia del cierre de bucle a coincidir con la almacenada previamente para misma posición. Para lograr esta coincidencia, las posiciones de todas las experiencias se actualizan usando la Ec.(31), donde α es una de tasa de corrección constante, N_f es el número de enlaces desde la experiencia e_i hacia otras experiencias, y

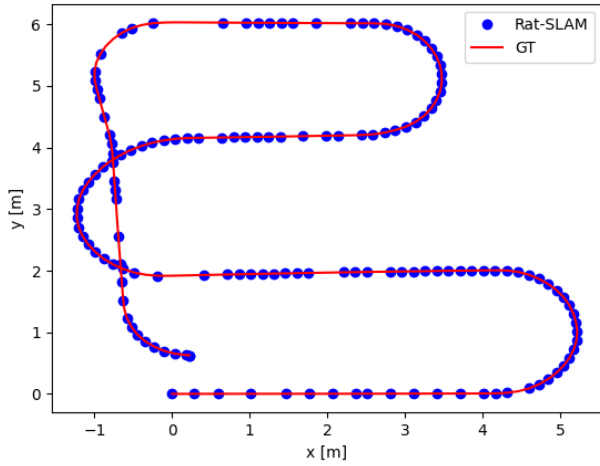


Fig. 3. Trayectoria de prueba con forma de zig-zag.

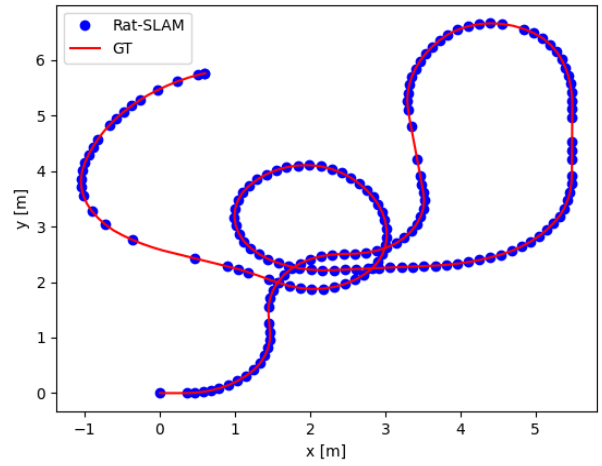


Fig. 4. Trayectoria de prueba con forma compleja.

N_t es el número de enlaces desde otras experiencias hacia la experiencia e_i .

$$\Delta \mathbf{p}^i = \alpha \left[\sum_{j=1}^{N_f} (\mathbf{p}^j - \mathbf{p}^i - \Delta \mathbf{p}^{ij}) + \sum_{k=1}^{N_t} (\mathbf{p}^k - \mathbf{p}^i - \Delta \mathbf{p}^{ki}) \right] \quad (31)$$

V. RESULTADOS EXPERIMENTALES

En esta sección presentamos los resultados obtenidos para trayectorias de prueba realizadas en simulación por computadora. En este trabajo se utilizó Robot Operating System (ROS) [12], ya que dispone de un simulador incorporado llamado Gazebo. El robot que se utilizó es el TurtleBot3 modelo *waffle_pi*, disponible en <https://wiki.ros.org/turtlebot3>.

Las trayectorias de prueba que se ensayaron son algunas de las más utilizadas en las evaluaciones del funcionamiento de aplicaciones robóticas. La primera tiene forma similar a un zig-zag, que es un tipo de trayectoria útil para tareas de prospección y monitorización de zonas, mientras que la segunda, es de mayor complejidad y una combinación de las anteriores. Los resultados de estas evaluaciones se muestran en las Fig. 3 y Fig. 4, cuyo error cuadrático medio (RMSE) igual a 0.14 m y 0.11 m., respectivamente. El error fue calculado con la diferencia entre las predicciones realizadas por nuestra propuesta de Rat-SLAM y los valores medidos por el odómetro.

VI. CONCLUSIONES

De todas estas evaluaciones, se puede concluir que las estimaciones realizadas por el algoritmo propuesto en este trabajo se aproximan cualitativa y cuantitativamente, con un bajo grado de error como se mencionó, con las lecturas obtenidas del odómetro del robot. Estos resultados validan que la PCMS puede almacenar las poses del robot en las frecuencias o tiempos de los impulsos o *spikes*, tal como ocurre en la actividad cerebral de los mamíferos. Además,

valida la posibilidad de aplicar CN para abordar el problema de SLAM, permitiendo sustituir la *pose cell* del algoritmo original implementada en *software* por una implementada en *hardware*. Este pasaje a *hardware* acerca la propuesta a los requisitos de tiempo real y potencia de cómputo a bordo de un RMA. Por otra parte, el uso de los memristores permite que el modelo neuronal implementado disponga, por un lado, de la capacidad intrínseca de memoria, gracias a la cual es posible relacionar una corriente de excitación con una pose, y por otro, disponer de mínimo consumo de energía, comparado con otras soluciones de *hardware* e incluso emulaciones digitales. Finalmente, constatamos que aplicar la idea de RNS hace posible codificar las distintas poses del robot en los tiempos o frecuencias de los impulsos o *spikes*.

REFERENCIAS

- [1] Jilles Vreeken et al. Spiking neural networks, an introduction. 2003.
- [2] Wulfram Gerstner and Werner M Kistler. *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge university press, 2002.
- [3] Leon Chua. Memristor-the missing circuit element. *IEEE Transactions on circuit theory*, 18(5):507–519, 1971.
- [4] Xiaoyan Fang, Derong Liu, Shukai Duan, and Lidan Wang. Memristive lif spiking neuron model and its application in morse code. *Frontiers in Neuroscience*, 16:853010, 2022.
- [5] Marianne Fyhn, Sturla Molden, Menno P Witter, Edvard I Moser, and May-Britt Moser. Spatial representation in the entorhinal cortex. *Science*, 305(5688):1258–1264, 2004.
- [6] John O’Keefe and Dulcie H Conway. Hippocampal place units in the freely moving rat: why they fire where they fire. *Experimental brain research*, 31:573–590, 1978.
- [7] JB Rank. Head-direction cells in the deep layers of dorsal presubiculum of freely moving rats. In *Soc. Neuroscience Abstr.*, volume 10, page 599, 1984.
- [8] Michael J Milford and Gordon F Wyeth. Mapping a suburb with a single camera using a biologically inspired slam system. *IEEE Transactions on Robotics*, 24(5):1038–1053, 2008.
- [9] Dmitri B Strukov, Gregory S Snider, Duncan R Stewart, and R Stanley Williams. The missing memristor found. *nature*, 453(7191):80–83, 2008.
- [10] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5:115–133, 1943.
- [11] Alexei Samsonovich and Bruce L McNaughton. Path integration and cognitive mapping in a continuous attractor neural network model. *Journal of Neuroscience*, 17(15):5900–5920, 1997.
- [12] Stanford Artificial Intelligence Laboratory. Robotic operating system.

Una Nueva Perspectiva para los Sistemas de SLAM Activo mediante Mapas de Incertidumbre

Sebastian Sansoni¹, Javier Gimenez¹, Gaston Castro², Santiago Tosetti¹, Flavio Capraro¹

¹Instituto de Automática (INAUT), San Juan, Argentina, ssansoni,jgimenez,stosetti,fcapraro at inuat.unsj.edu.ar

²CIFASIS, French Argentine International Center for Information and Systems Sciences (CONICET – UNR)
, Rosario, Argentina, gcastro at cifasis-conicet.gov.ar

Abstract—Los sistemas de *Active Simultaneous Localization and Mapping (ASLAM)* utilizan indicadores para planificar la exploración de un entorno desconocido. Estos indicadores se basan en la incertidumbre del mapa, del agente, la distancia recorrida, el costo energético, la cobertura, el tiempo requerido, entre otros. Fusionar estos indicadores es un desafío, ya que cada uno de ellos tiene un rango de valores y unidades diferentes, requiriendo de factores de ponderación que deben ser ajustados empíricamente. Este trabajo presenta una nueva forma de modelar la variabilidad de los datos, que acompañado de un Mapa de Incertidumbre (MI), permite fusionar tanto la dispersión del mapa como la del agente en un único indicador. Usualmente las fronteras entre lo conocido y lo desconocido son utilizadas como objetivos de exploración. En este trabajo y mediante los MI es posible redefinir estos objetivos introduciendo el concepto de Fronteras de Incertidumbre (FI). Al igual que las fronteras usuales las FI sirven como criterio de parada de la exploración incluso en espacios abiertos, un problema sin resolver en la literatura. Esta metodología es independiente del sistema de SLAM subyacente y puede ser adaptada para cualquier tipo de sensor. En este trabajo se presentan resultados experimentales en un entorno simulado. Todo el sistema se encuentra implementado mediante nodos de ROS, los cuales están disponibles en <https://github.com/Seba-san/UncertaintyMap>.

Index Terms—Entropía relativa, Mapa de incertidumbre, SLAM Activo

Sistema de Odometría Visual-Inercial para Aplicaciones XR

1st Bruno Zanotti

Departamento de Ciencias de la Computación

FCEIA-UNR

Rosario, Argentina

brunozanotti96@gmail.com

2nd Mateo De Mayo

Collabora Ltd.

Cambridge, Reino Unido

mateodemayo@gmail.com

3rd Taihú Pire

CIFASIS

CONICET-UNR

Rosario, Argentina

pire@cifasis-conicet.gov.ar

Resumen—

En aplicaciones de realidad extendida (XR) es indispensable que el sistema de localización del dispositivo sea robusto frente a movimientos bruscos y oclusiones en las cámaras. A su vez para que la experiencia de usuario sea fluida e inmersiva debe ser liviano para que pueda correr en tiempo real. Los dispositivos XR, como los cascos de realidad virtual (VR), combinan múltiples cámaras e IMU que pueden ser usados por métodos de Odometría Visual-Inercial (VIO) para estimar su localización.

En este paper se propone un sistema VIO capaz de estimar la localización de dispositivos XR. El sistema propuesto tiene como punto de partida al algoritmo de Basalt, con lo cual mantiene una ventana de keyframes que optimiza y marginaliza de forma inteligente para mantener el sistema computacionalmente eficiente, y la vez lo extiende permitiendo trabajar con cascos con más de dos cámaras no paralelas, utiliza la IMU para predecir la localización y es capaz de reidentificar features perdidas por oclusiones o movimientos bruscos.

Se realizaron experimentos sobre un dataset de un casco de XR para validar el sistema VIO desarrollado. Los resultados muestran que el sistema logra una mayor robustez y precisión, y una trayectoria suave, requerimientos necesarios para aplicaciones XR.

Keywords—Realidad Extendida, Realidad Virtual, Localización, VIO, SLAM.

Sesión 3: Localización, Mapeo y Planificación de trayectorias II

Hacia un sistema SLAM Visual e Inercial con Estimación Consistente de Incertidumbre en Tiempo Acotado

Gastón Castro
Universidad de San Andrés
CONICET-UDESA
Buenos Aires, Argentina
gcastro@udesa.edu.ar

Pablo F. Alcantarilla
SLAMcore¹
Londres, Reino Unido
pablo@slamcore.com

Taihú Pire
Laboratorio de Robótica
CIFASIS (CONICET-UNR)
Rosario, Argentina
pire@cifasis-conicet.gov.ar

Abstract—

Este trabajo aborda el problema de SLAM Visual e Inercial con el objetivo de presentar una solución consistente en términos de estimación de la incertidumbre. Se propone un método capaz de calcular incertidumbres globales aproximadamente consistentes sin aumentar la complejidad con el tamaño total del área explorada. El método permite la selección arbitraria de áreas locales para optimización, introduciendo una metodología para construir un *prior* virtual en tiempo acotado. Se utiliza una topología de factores aproximando el *prior* utilizando árboles de Chow-Liu, obteniendo un problema de optimización local que permite calcular estimaciones de incertidumbre consistentes. Esto supera varias deficiencias de enfoques anteriores que se basan en condicionamiento (fijación de variables) y/o marginalización con ventana deslizante. El trabajo se basa en el sistema de SLAM Visual descrito en [1] con las siguientes contribuciones: i) tratamiento de mediciones inerciales, ii) experimentos basados en un nuevo framework de simulación, iii) una implementación eficiente en C++.

Index Terms—SLAM, Estimación de estado, Optimización, Incertidumbre, Consistencia, Marginalización

I. INTRODUCCIÓN

En un mundo donde a los robots se les asignan periódicamente tareas cada vez más diversas, resulta esencial proporcionar capacidades de localización y mapeo eficientes, sólidas y precisas. Este problema se denomina localización y mapeo simultáneos (SLAM), y es un requisito fundamental para lograr una navegación autónoma, y representa uno de los principales temas de investigación en la comunidad de robótica [2].

Un método de SLAM capaz de trabajar en el mundo real requiere manejar medidas con diversos grados de incertidumbre y error debido a entornos impredecibles, sensores contaminados por ruido, representaciones discretas entre otras. Por tanto, es deseable que los sistemas SLAM proporcionen una medida de incertidumbre contenida en cada variable estimada. El consenso es tratar estas propiedades físicas (poses de los robots y puntos de referencia) como variables aleatorias que siguen una determinada distribución probabilística [3], [4], [5], con formulaciones muy novedosas entre las que destaca [6].

¹ El trabajo fue realizado mientras el autor trabajaba para Slamcore Ltd.

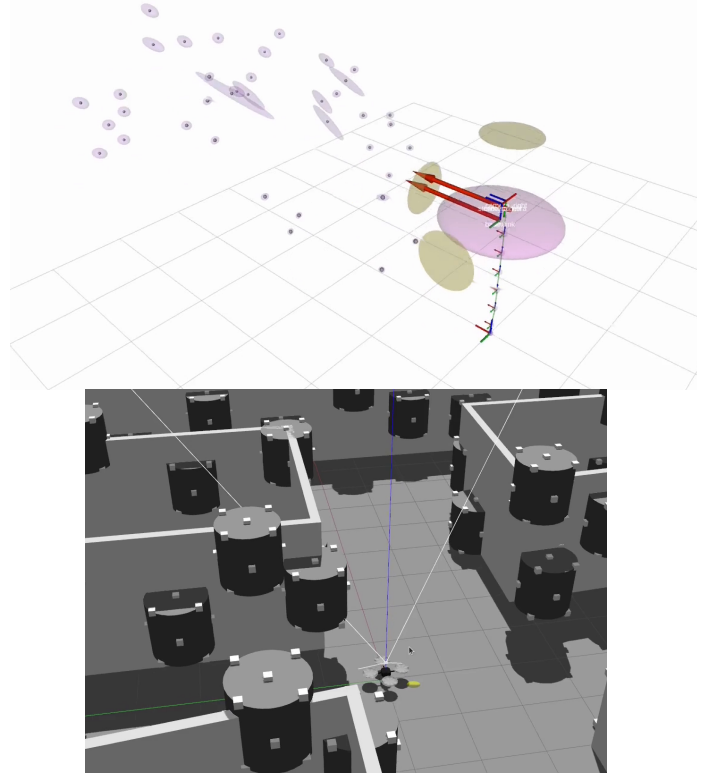


Fig. 1: Método de estimación funcionando en un escenario simulado. El método estima las variables de estado de un dron simulado, los *landmarks* del mapa y su incertidumbre representada como elipsoides en color magenta. La imagen inferior muestra el entorno de simulación en Gazebo con *landmarks* simulados.

Como se define en [7], un estimador de estado se considera *consistente* si los errores de estimación son de media cero y la covarianza estimada es mayor o igual, en el sentido matricial positivo-definido, a la covarianza calculada mediante un enfoque óptimo dado, como la resultante de optimizar el del grafo completo (*batch-optimization*). La *batch-optimization* como línea base de consistencia también tiene una mayor significancia teórica, ya que la matriz Hessiana resultante del sistema se aproxima a la matriz de información de Fisher, que a su vez es el límite inferior de Cramér-Rao (CRLB) [5].

En este trabajo se propone un método de SLAM Visual e Inercial capaz de lograr covarianzas consistentes. El método se basa en la construcción de *priors* virtuales, que son representaciones simplificadas del mapa global. Estos *priors* se utilizan para contextualizar las optimizaciones locales, lo que resulta en estimaciones de incertidumbre más consistentes. En la Figura 1 se puede observar un escenario donde el método estima las variables de estado de un dron simulado, los landmarks del mapa y las incertidumbre involucradas en las estimaciones (representadas como elipsoides de confianza).

II. TRABAJO RELACIONADO

Lograr un estimador de estado consistente que no aumente en complejidad con el tamaño del problema es un área de gran interés científica. Varios enfoques pueden contener la escala de complejidad del problema SLAM, pero caen en metodologías con inconvenientes que generan inestabilidades a futuro. Se pueden mencionar principalmente dos tipos de enfoques de complejidad constante o acotada, para mantener la operación en tiempo real: (a) métodos que realizan optimizaciones de área local mediante acondicionamiento, también conocido como fijación de variables; y (b) métodos que realizan optimizaciones de ventana deslizante marginalizando toda información previa.

Es posible realizar una optimización local sobre un área del grafo condicionando las variables seleccionadas a aquellas directamente relacionadas que componen la frontera del área. El principal supuesto es que las estimaciones de las variables fijas ya han convergido y son correctas. Varios trabajos de investigación en Visual e Inercial SLAM han reportado un gran éxito utilizando optimización y condicionamiento local [8], [9], [10], [11]. Un inconveniente predominante de este enfoque es que, como se supone que las variables fijas son completamente certeras, no es posible recuperar estimaciones de incertidumbre globalmente consistentes. Debe haber un factor (llamado factor *prior*) en el grafo local que responda a las distribuciones de probabilidad marginal y modele la correlación entre las variables seleccionadas, impuestas por las mediciones omitidas.

En el caso de una solución de ventana deslizante, es posible eliminar adecuadamente las variables que quedan fuera de una ventana de estimación de tamaño fijo; este proceso se conoce como “marginalización de variables”. La información recopilada se integra recursivamente y se transporta en un factor de distribución *prior* que correlaciona las variables activas dentro de la ventana de estimación [12], [13]. Qin et al. [11] traslada la ventana de estimación y continua usando el *prior* construido hasta ese momento, incluso después del cierre de un ciclo. En [14] se descarta el *prior* y construye uno nuevo usando condicionamiento como medio para cortar la historia anterior del grafo pero, aunque esto es posible, la consistencia de la incertidumbre global no se mantiene.

Hay tres consideraciones importantes sobre este enfoque: (i) conlleva densificar matrices originalmente dispersas; (ii) surgen problemas de linealización a medida que los modelos jacobianos se integran y nunca se vuelven a re-calcular; y

(iii) el factor *prior* se vuelve inexacto, o incluso inválido, cuando se cierra un ciclo en la trayectoria, ya que significa nueva información relacionada con variables ya marginalizadas. Esto implica que los enfoques de marginalización de ventana deslizante solo son capaces de proporcionar estimaciones consistentes de incertidumbre que pueden deteriorarse progresivamente.

El presente trabajo propone un nuevo enfoque de optimización local para SLAM que aborda todos los inconvenientes antes mencionados y proporciona una solución que calcula las incertidumbres globales manteniendo las ventajas computacionales de las optimizaciones locales. El método permite la selección de áreas locales arbitrarias para la actualización preservando al mismo tiempo una consistencia aproximada tanto en los valores estimados como en las incertidumbres. La solución define una topología de factores virtuales aproximando la covarianza marginal del área local utilizando el algoritmo de Chow-Liu [15]. Se destaca la inclusión del tratamiento de mediciones inerciales siendo un incremental sobre un trabajo previamente publicado [11].

REFERENCIAS

- [1] G. Castro, F. Pessacq, and P. De Cristóforis, “Consistent slam using local optimization with virtual prior topologies,” in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 3321–3328.
- [2] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. Leonard, “Simultaneous Localization And Mapping: Present, Future, and the Robust-Perception Age,” *IEEE Transactions on Robotics*, vol. 32, 06 2016.
- [3] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. MIT press, 2005.
- [4] F. Dellaert and M. Kaess, “Factor Graphs for Robot Perception,” *Foundations and Trends in Robotics*, vol. 6, no. 1-2, pp. 1–139, 2017.
- [5] T. D. Barfoot, *State Estimation for Robotics*. Cambridge University Press, 2017.
- [6] F. Bai, T. Vidal-Calleja, and G. Grisetti, “Sparse pose graph optimization in cycle space,” *IEEE Transactions on Robotics*, 2021.
- [7] S. J. Julier and J. K. Uhlmann, “A non-divergent estimation algorithm in the presence of unknown correlations,” in *Proceedings of the 1997 American Control Conference (Cat. No. 97CH36041)*, vol. 4. IEEE, 1997, pp. 2369–2373.
- [8] V. Usenko, N. Demmel, D. Schubert, J. Stückler, and D. Cremers, “Visual-inertial mapping with non-linear factor recovery,” *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 422–429, 2019.
- [9] T. Qin, S. Cao, J. Pan, and S. Shen, “A General Optimization-based Framework for Global Pose Estimation with Multiple Sensors,” in *Computing Research Repository*, 01 2019.
- [10] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, “ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial, and Multimap SLAM,” *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.
- [11] T. Qin, P. Li, and S. Shen, “VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator,” *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.
- [12] G. Sibley, L. Matthies, and G. Sukhatme, “Sliding window filter with application to planetary landing,” *Journal of Field Robotics*, vol. 27, no. 5, pp. 587–608, 2010.
- [13] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, “Keyframe-based visual-inertial odometry using nonlinear optimization,” *The International Journal of Robotics Research*, vol. 34, no. 3, pp. 314–334, March 2015.
- [14] R. Mur-Artal and J. D. Tardós, “Visual-Inertial Monocular SLAM With Map Reuse,” *IEEE Robotics and Automation Letters*, April 2017.
- [15] C. Chow and C. Liu, “Approximating discrete probability distributions with dependence trees,” *IEEE transactions on Information Theory*, vol. 14, no. 3, pp. 462–467, 1968.

Hacia una Fusión Visual-Inercial-GNSS con Restricciones de Distancia

1st Javier Cremona
CIFASIS
CONICET-UNR
Rosario, Argentina
cremona@cifasis-conicet.gov.ar

2nd Javier Civera
I3A
Universidad de Zaragoza
Zaragoza, España
jcivera@unizar.es

3rd Taihú Pire
CIFASIS
CONICET-UNR
Rosario, Argentina
pire@cifasis-conicet.gov.ar

Resumen—La fusión de datos de cámaras, sensores inerciales y GNSS es una estrategia empleada para la localización y mapeo de robots móviles, especialmente en entornos abiertos, no estructurados y visualmente complejos. Este trabajo presenta un enfoque novedoso que incorpora las mediciones de GNSS como restricciones de distancia en la optimización conjunta visual-inercial, eliminando la necesidad de alinear los sistemas de coordenadas global y visual-inercial, y simplificando así el proceso de fusión. La evaluación se realiza con datos reales recolectados en un robot agrícola. Los resultados comparativos con ORB-SLAM3 y otros sistemas de fusión visual-inercial-GNSS demuestran la eficacia de la propuesta, con mejoras en la robustez de la estimación.

Keywords—Fusión de Sensores, GNSS, Localización, SLAM, Agricultura de Precisión

Abordando los Desafíos de la Detección de Ciclos en Entornos Agrícolas

Nicolas Soncini
Laboratorio de Robótica
CIFASIS (UNR-CONICET)
Rosario, Santa Fe, Argentina
soncini@cifasis-conicet.gov.ar

Javier Civera
Robotics, Perception and Real-Time Group
Universidad de Zaragoza
Zaragoza, España
jcivera@unizar.es

Taihú Pire
Laboratorio de Robótica
CIFASIS (UNR-CONICET)
Rosario, Santa Fe, Argentina
pire@cifasis-conicet.gov.ar

Resumen

La detección de ciclos es un componente fundamental de los sistemas de localización de los robots autónomos, especialmente en operaciones a largo plazo. Sin embargo, escasa literatura puede encontrarse sobre el tópico de detección de ciclos en campos agrícolas. En este trabajo abordamos la detección de ciclos en campos agrícolas, donde se espera que el uso de robots autónomos para diversas tareas agrícolas tenga un impacto fundamental en los próximos años, basada únicamente en información visual. En particular, mostramos resultados de evaluar métodos de detección de ciclos en entornos agrícolas y proponemos mejoras basadas en información por cámaras estéreo para robustecer los mismos.

AGRADECIMIENTOS

Este trabajo fue apoyado por CONICET (PIBAA N° 0042), AGENCIA I+D+i (PICT 2021-570) y Universidad Nacional de Rosario (PCCT-UNR 80020220600072UR).

Sistema SLAM acelerado por GPU para mapeo volumétrico globalmente consistente en tiempo real

Emiliano Höss

Departamento de Computación
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires
Buenos Aires, Argentina
ehoss@dc.uba.ar

Pablo De Cristóforis

Departamento de Computación
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires
Buenos Aires, Argentina
pdecristo@dc.uba.ar

Resumen—El uso de GPUs integradas en robots móviles está en expansión debido a la creciente demanda de procesamiento embebido. La adaptación de algoritmos originalmente diseñados para CPUs a GPUs no es sencilla debido a heterogeneidad en las distintas arquitecturas de hardware y los recursos disponibles. En este trabajo, presentamos una versión acelerada del sistema de SLAM Voxgraph para mapeo volumétrico utilizando procesamiento en paralelo en GPUs. El sistema presentado se evaluó en una PC de escritorio con GPU y se comparó con la versión original de Voxgraph. Los resultados muestran que el sistema propuesto supera en 4x al front-end y en 8x al back-end a la versión original en términos de tiempos de ejecución, manteniendo la misma precisión. El nuevo sistema, acuñado coVoxgraph, se encuentra disponible como código abierto en GitHub¹.

Index Terms—Mapeo, SLAM, SDF, GPU.

I. INTRODUCCIÓN

Para alcanzar el objetivo de navegar e interactuar de manera autónoma en entornos desconocidos, los sistemas de SLAM (Simultaneous Localization and Mapping) necesitan construir un modelo interno del mundo observado, es decir, un mapa. Las estimaciones que producen los robots de su propio movimiento se realiza de manera incremental a medida que se mueven por el entorno, esto conduce a errores de aproximación en la pose y a un mapeo inconsistente. Para mitigar los efectos inevitables que produce la incertidumbre sobre estos cálculos y mantener un mapa globalmente consistente se utilizan técnicas de cierre de ciclo (*loop closure*).

El mapeo basado en características convierte los datos crudos del sensor en un conjunto de características del ambiente que utiliza para la construcción del mapa. Muchos sistemas SLAM bien conocidos siguen este enfoque y han demostrado crear mapas basados en características globalmente consistentes en tiempo real [1]–[3]. Aun así, son de uso restringido para otras tareas más allá de la localización, debido a las dificultades para extraer la forma y la conectividad de las superficies de los objetos en el entorno a partir de una representación dispersa del mapa.

Por otro lado, los mapas densos son adecuados no solo para la estimación de la pose, sino también para la reconstrucción

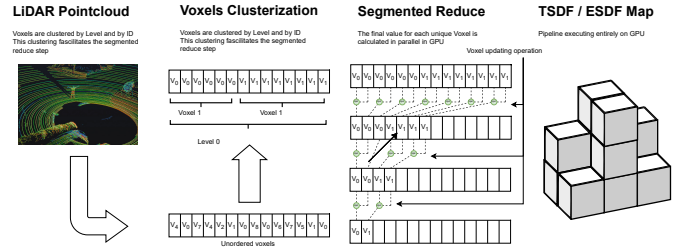


Figura 1: Flujo de datos de la implementación en GPU de la integración de nube de puntos del sensor en el sistema front-end de coVoxgraph.

de escenas, la planificación de trayectorias, la detección de objetos, la evasión de obstáculos y el control del movimiento. El primer sistema SLAM denso que utiliza la consistencia fotométrica de cada píxel para estimar la trayectoria de una cámara en mano fue [4]. Posteriormente, se desarrollaron varios sistemas SLAM densos basados en la técnica de ajuste de paquetes fotométricos (*Bundle Adjustment*) [5], [6] y actualmente utilizando aprendizaje profundo [7]–[9]. Sin embargo, estos sistemas aún requieren muchos recursos computacionales y no logran ejecutar en tiempo real en escenarios de gran escala. Tener la capacidad de crear mapas densos y globalmente consistentes en tiempo real sigue siendo un desafío para resolver el problema de la navegación autónoma a largo plazo en entornos complejos.

Las superficies implícitas modeladas como funciones de distancia con signo (SDF por su nombre en inglés *Signed Distance Functions*), introducidas por primera vez en [10], han demostrado ser una representación efectiva para el mapeo denso [11]. Sin embargo, producir un mapa globalmente consistente usando SDFs es costoso, ya que la optimización global del mapa se vuelve rápidamente intratable a medida que aumenta la cantidad de datos. Una alternativa para abordar este problema es representar el entorno reconstruido como una colección de submapas. La ventaja de este enfoque es que la pose del sensor, en el momento en que se realiza una nueva registración de un nuevo submapa, solo necesita ser registrada con respecto al submapa actual. Si la trayectoria completa no se considera durante la optimización del mapa

¹<https://github.com/lrse-uba/covoxgraph>

denso, los submapas pueden ser restringidos entre sí a través de la alineación geométrica [12]. En [13] esta idea se extiende proponiendo usar una alineación libre de correspondencia basada en la Función de Distancia con Signo Euclidiana (ESDF) que representa una cuadrícula de voxels donde cada punto contiene su distancia euclidiana al obstáculo más cercano. El sistema presentado en este trabajo, llamado Voxgraph, formula el problema como una optimización de grafos de pose, incluyendo restricciones de odometría y cierre de ciclo para mantener la consistencia global ante la posibilidad de error introducidos por cierres de ciclo que involucren una larga cadena de submapas en su trayectoria. Comenzando con la implementación original de CPU de Voxgraph, desarrollamos una nueva versión basada en GPU acuñada coVoxgraph que supera en $8\times$ al back-end y $4\times$ al front-end del sistema original considerando los tiempos de ejecución manteniendo la misma precisión. Las contribuciones de este trabajo se resumen de la siguiente manera:

- Un módulo de Estructura-de-Arrays (SoA) para facilitar la gestión de memoria heterogénea y la coalescencia; y para reducir el costo de transferencia de datos entre CPU y GPU.
- Reimplementación parcial de la biblioteca Eigen para álgebra lineal y la biblioteca Kindr para operaciones de cuaterniones utilizando programación genérica para soportar el diseño de datos SoA.
- Una tabla hash de múltiples valores GPU-CPU que almacena valores e índices organizados en SoA para permitir el intercambio flexible de diseños de datos subyacentes.
- Un nuevo modelo de flujo de datos para convertir los datos de la nube de puntos en un mapa de Función de Distancia Firmada Truncada (TSDF) y para actualizar el mapa ESDF a partir del mapa TSDF.
- La implementación en GPU de un método de aproximación por cuadrados mínimos, similar a Ceres, pero solo enfocado en la resolución del problema presente en este trabajo.
- Un repositorio de acceso público con todo sistema desarrollado: <https://github.com/lrse-uba/covoxgraph/>

II. TRABAJOS PREVIOS

El sistema de SLAM Voxgraph prioriza la legibilidad y la extensibilidad sobre el rendimiento, con una arquitectura que fue diseñada para ser ejecutada en CPU. El sistema es principalmente de un solo hilo y la concurrencia se utiliza solo para acelerar el procesamiento por lotes de datos dentro de los módulos de integración de nube de puntos del sensor, tal como se describe en [14]. Este sistema Voxblox se usa para construir incrementalmente las ESDFs directamente a partir de Campos de Distancia Truncada con Signo (TSDFs) y aprovechar la información de distancia ya contenida dentro del radio de truncamiento. Los mapas TSDFs resultan más rápidos que los Octomaps para construir y suavizar el ruido del sensor en muchas observaciones. Voxgraph se divide en un front-end y un back-end (ver Fig. 2).

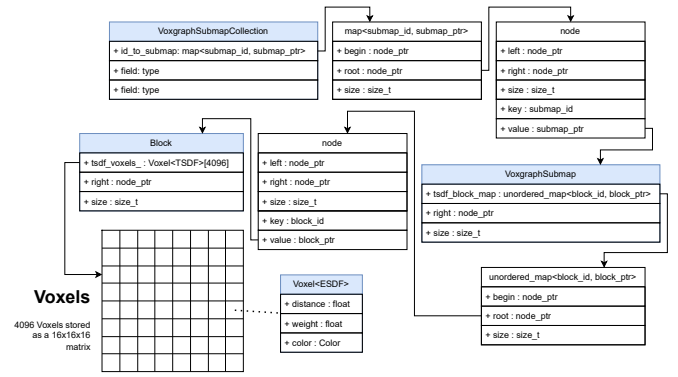


Figura 2: Ordenamiento de memoria en el sistema Voxgraph, las flechas representan las direcciones necesarias para acceder a los datos, y los nodos sombreados en celeste representan los objetos definidos en el código de Voxgraph

El **front-end** de Voxgraph podría dividirse en dos módulos: uno responsable de convertir las mediciones de sensores en submapas y otro para crear las restricciones que se pasarán al back-end para corregir los errores durante la creación de los submapas. Por su parte, el primer módulo también se divide en dos etapas. Primero, los datos del sensor entrante se integran en un volumen TSDF para incluir o actualizar los voxels que construyen el mapa TSDF y el segundo propaga los voxels actualizados desde el TSDF a la representación de volúmenes ESDF. Los voxels creados durante ambos pasos se agrupan en bloques de tamaño fijo que a su vez se indexan utilizando una tabla hash de bloques que representa internamente el mapa esparso global del entorno explorado. Este mapa global podría verse como una tabla hash de bloques dispersa en memoria (ver Fig. 2). El tamaño de los bloques puede ser cambiado por un parámetro de configuración, pero permanece igual durante la ejecución. Cuando una nueva nube de puntos llega desde el sensor, se lanzan múltiples hilos de ejecución que procesan un punto dentro de la nube y de manera serializada incorpora todos los voxels que intersectan con el rayo que se proyecta desde ese punto. Una vez que termina de incorporar todos los voxels, continúa con otro punto dentro de la nube de puntos. Este procesamiento se repite hasta alcanzar todos los puntos de la nube de puntos.

El **back-end** de Voxgraph es responsable de mantener el conjunto de restricciones generadas por el front-end (odometría, cierre de ciclo y registro de submapas) y estima la alineación de la colección de submapas más probable minimizando el error total de todas las restricciones del grafo de poses. Para hacer eso, tiene que resolver la aproximación por mínimos cuadrados no lineales calculando las matrices de residuos y jacobianos para cada voxel y vértice de los submapas superpuestos. El cálculo no ponderado de los valores de los elementos de estas matrices es completamente independiente entre sí, el único punto en común es el valor de la sumatoria del peso de los voxels o vértices relevantes, necesarios para calcular la media aritmética ponderada de los valores finales de las matrices de residuos y jacobianas. Sin

embargo, como este valor se precalcula para cada submapa cuando este es finalizado, no añade ninguna dependencia al cálculo, y por lo tanto, su implementación en GPU es perfectamente paralelizable [15].

Durante la última década, se han hecho esfuerzos para implementar algoritmos SLAM en múltiples unidades de procesamiento en sistemas heterogéneos CPU-GPU dado que esto representa un enfoque más realista para su uso en la vida real, destacando los beneficios y desventajas de las soluciones de computación embebidas para robots móviles. Aún así, la portabilidad de algoritmos originalmente diseñados para CPUs a GPUs no es sencilla debido a las diferencias en las arquitecturas de hardware y recursos disponibles. Tal es el caso de [16]–[18], donde se evalúan versiones aceleradas por GPU de sistemas SLAM basados en características. La GPU también ha sido utilizada para algunos componentes de SLAM RGB-D denso [11], [19]. En el contexto de SLAM LiDAR, el uso de GPU se limitó principalmente a acelerar la coincidencia de escaneo en el front-end [20], [21]. Sin embargo, en la mayoría de los trabajos anteriores, el back-end, y específicamente, la optimización del grafo de poses sigue siendo realizada en una CPU. En este trabajo se introduce coVoxgraph: un sistema de SLAM completo, enteramente acelerado por GPU (en su front-end y back-end) para el mapeo volumétrico globalmente consistente en tiempo real.

II-A. Profiling

Se realizó un análisis exhaustivo de los tiempos de ejecución de cada módulo de Voxgraph. Esto proporciona información valiosa sobre el consumo de tiempo y también sobre la distribución del uso de la CPU, detectando cuellos de botella, y guiando los esfuerzos para mejorar el sistema en términos de rendimiento para la versión acelerada en GPU.

Voxgraph requiere poder ser ejecutado en tiempo real para garantizar la estabilidad de la cantidad de datos de cada submapa. Las nubes de punto de entrada, destinadas a ser parte del submapa actual, se omitirán si el sistema no puede procesarlas en tiempo real. Esta variabilidad afecta la cantidad final de datos que se utilizan para cada submapa y, por lo tanto, el tiempo que demanda. Por esta razón, para realizar el análisis de tiempos de ejecución, se modificó el código original para evitar la omisión de nube de puntos, de manera similar a lo que se conoce como *process-every-frame* [22]. El conjunto de datos utilizado para el análisis de tiempos fue el mismo que Voxgraph (para más detalles, consulte la Sección IV).

Cada determinado tiempo se crea un nuevo submapa en Voxgraph, en ese momento el sistema necesita realizar la integración de las nubes de puntos recolectadas en esa ventana de tiempo, la actualización del mapa ESDF y la optimización del grafo de poses. La Fig. 3 muestra la distribución de tiempo en el momento en el que se crea un nuevo submapa. Considerando que, entre una medición y otra, hay alrededor de 20 ejecuciones de la integración de la nube de puntos, los tres módulos que demandan más tiempo de ejecución son: la integración de la nube de puntos, la actualización del mapa ESDF y la optimización del grafo de poses, como ya fuera

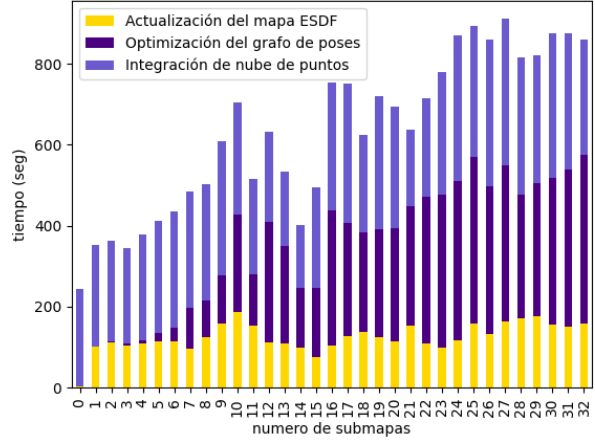


Figura 3: Analisis de mediciones de tiempo de cada modulo de Voxgraph.

mencionado tambien por los autores de Voxgraph [13]. Como puede observarse, el tiempo de ejecución del front-end (actualización del mapa ESDF y la integración de nube de puntos), se mantiene constante debido a que no hay cambios significativos en la cantidad de nube de puntos que se integran. Sin embargo, el tiempo de ejecución del back-end (optimización de grafo de poses) aumenta a medida que crece la cantidad de submapas totales en el mapa global.

III. SISTEMA PROPUESTO

En las siguientes subsecciones detallamos las principales contribuciones que permitieron portar el sistema Voxgraph de su versión original para CPU a la versión acelerada en GPU.

III-A. Struct of Arrays

Los elementos de una colección de objetos pueden disponerse en memoria siguiendo dos enfoques distintos: por un lado SoA (por su nombre en inglés Struct of Arrays) donde los objetos se separan de acuerdo a sus campos en distintos arreglos; y por otro lado, AoS (por su nombre en inglés Array of Structs) donde los objetos son almacenados individualmente y de manera consecutiva. Para este trabajo se decidió utilizar el enfoque SoA debido a que ofrece varias ventajas. Primero, mejora la localidad de los datos, ya que cada componente de la estructura de datos de un objeto se almacena en un array separado, aumentando la utilización de la memoria caché y la coalescencia de memoria. En segundo lugar, la disposición SoA puede facilitar la vectorización, que es la ejecución paralela de operaciones en múltiples elementos de datos utilizando núcleos SIMD en procesadores tanto de GPU como CPU. En tercer lugar, mejora el rendimiento de los datos, minimizando el número de transacciones de memoria y disminuyendo el tamaño total de cada transacción, debido a la reducción del relleno que se utiliza dentro de la estructura de cada objeto para alinearlo en memoria (*padding*).

III-B. Hashing Espacial

Voxgraph utiliza una tabla hash de tamaño dinámico que hace uso del enfoque presentado por [23]. Hay tres niveles de particiones al indexar posiciones en la representación del mapa global. El primer nivel utiliza una colección de submapas superpuestos, junto con sus restricciones y poses relativas para estimar su alineación más probable. Esta colección se implementa utilizando una tabla hash de la Biblioteca Estándar C++ para indexar cada submapa. Dentro de cada submapa hay un segundo nivel de partición representado por bloques, que son esencialmente una matriz de un número fijo de voxels. Los bloques se almacenan en una tabla de hash de la Biblioteca Estándar C++ utilizando dos funciones de hash. La primera función transforma las coordenadas continuas representadas como una tupla de puntos flotantes de 64 bits, a coordenadas globales discretas representadas por una tupla de enteros sin signo de 64 bits. Esto crea un segundo nivel de partición llamado Discrete Global Grid (DGG), después de eso se aplica la segunda función hash denominada DECO para mejorar la eficiencia al recorrer los datos, siguiendo el metodo de [24]. El problema con este enfoque es que cuando se crea un bloque, se crea un nuevo objeto y la tabla de hash guarda el puntero a su posición en memoria. Creando así un patrón de memoria dispersa dentro de cada submapa. El último nivel de partición se realiza dentro de los bloques, utilizando una matriz cuadrada densa de voxels del mismo tamaño (ver Fig. 2). El problema con este modelo de tablas hash utilizado por Voxgraph es que sufren de un rendimiento deficiente inducido por patrones de acceso a memoria irregulares. Los avances recientes [25], [26] han demostrado que la adopción de SoA para las tablas de hash y sus claves, mejora sustancialmente el rendimiento de las operaciones de inserción y búsqueda. La disposición AoS proporciona una localidad de caché relativamente alta si se accede tanto a la clave como al valor simultáneamente. Sin embargo, si sólo se accede a la clave, la efectividad de la caché se reduce. Esto es especialmente crítico si el tipo de valor es grande en comparación con el tipo de clave, como es en el caso de Voxgraph. En este trabajo, adoptamos estas ideas junto con el módulo SoA para implementar la tabla de hash.

III-C. Integración de nubes de puntos

El front-end de Voxgraph es responsable de generar los voxels del mapa TSDF utilizando los datos crudos del sensor. Dado que el mapa TSDF utiliza una representación menos detallada para almacenar la información de la nube de puntos, un solo voxel intersectará la información de varios puntos del sensor. Por lo tanto, la información almacenada en el voxel necesita ser actualizada a partir de los nuevos puntos que lo intersectan en cualquier orden. Al introducir la ejecución concurrente, estas actualizaciones necesitan sincronizarse cada vez que se actualice el voxel en cuestión, como se ve en la Fig. 4, donde se representan las primitivas de sincronización en rojo. Tomando en cuenta que el método de actualización de cada voxel es una operación conmutativa y asociativa, se puede utilizar la técnica de reducción en GPU introducida por

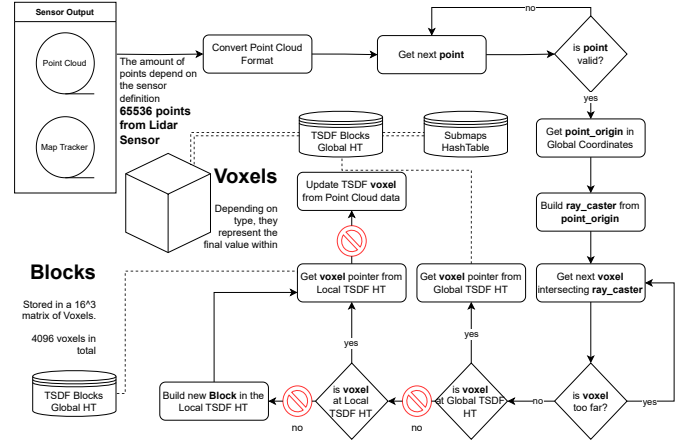


Figura 4: Diagrama de flujo de la integración de las nubes de puntos del sensor. Los puntos de sincronización están señalizados en rojo.

[27], donde cada paso en este algoritmo reduce a la mitad el número de voxels que se actualizan, resultando en un orden de complejidad $O(\log(n))$, donde n es el número de voxels a actualizar. Proponemos un flujo de datos representado en la Fig. 1 que comienza en la transformación de los datos crudos del sensor de la nube de puntos hasta que todos los voxels son generados y actualizados dentro del sistema utilizando la técnica de Reducción por Segmentos en GPU para acelerar el proceso. Los pasos intermedios del flujo de datos son responsables de agrupar el voxel de tal manera que estén disponibles para realizar el paso de reducción de manera óptima, como se propone en [28]. Finalmente, cada nuevo voxel único se asigna en la tabla hash de manera sincronizada. La asignación o actualización de voxels en la tabla de hash consume mucho tiempo ya que depende de la sincronización cada vez que escribimos un nuevo valor, por lo que retrasar esta operación hasta el último momento permite una reducción drástica en el número de elementos que requieren ser sincronizados y por ende en su tiempo de ejecución.

III-D. Aproximación por cuadrados mínimos

Voxgraph realiza una aproximación por cuadrados mínimos no lineal para el grafo de poses cada vez que se finaliza un submapa nuevo. Este proceso es el primer paso del back-end y utiliza un modelo esparso por definición, dado que los submapas creados son el resultado de la intersección entre la trayectoria arbitraria del robot y la representación interna del mapa global.

Para resolver el problema de aproximación por cuadrados mínimos resultante, Voxgraph utiliza la biblioteca Ceres, la cual solo ofrece una versión acelerada en GPU para problemas de aproximación de sistemas lineales densos. Por este motivo, presentamos en este trabajo una implementación del método iterativo de gradiente conjugado para GPU fuertemente basada en implementación en CPU de Ceres, pero limitada al problema de actualización del grafo de poses que se presenta en Voxgraph.

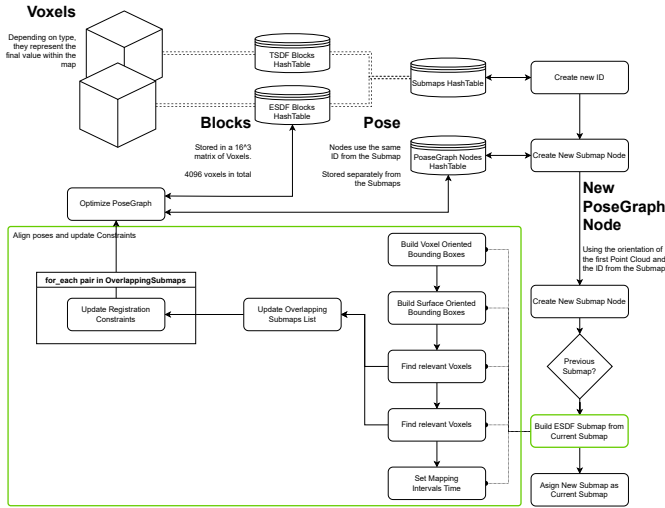


Figura 5: Diagrama de flujo de ejecución del back-end en Voxgraph.

La estrategia general al resolver problemas de optimización no lineales es a través de una secuencia de aproximaciones al problema original [29]. En cada iteración, se resuelve la aproximación para determinar una corrección Δx al vector x . Para determinar el paso siguiente Δx en cada iteración se pueden diferenciar dos grandes estrategias: Line Search y Trust Region. Para este trabajo decidimos utilizar la estrategia de Trust Region, en particular implementamos el algoritmo de Levenberg-Marquardt (LMA), que puede verse como una interpolación entre el algoritmo de Gauss-Newton (GNA) y el método de Descenso de Gradiente. LMA por su parte es más robusto que GNA ya que no requiere proveerle de una solución inicial que este demasiado cercana a la solución final. Por su parte, el sistema lineal se representa utilizando el Complemento Schur, lo que simplifica y acelera las operaciones a realizarse durante las iteraciones del algoritmo y a su vez facilita la paralelización por bloques en GPU. Para la representación interna de las matrices utilizamos la Biblioteca de Nvidia cuSparse, que provee la mayoría de las funcionalidades que se requieren para el manejo de matrices esparsas en memoria de GPU, también utilizamos la Biblioteca de Nvidia cuBlas que ofrece el conjunto de funcionalidad estandar de algebra lineal implementadas en GPU que utilizamos para los algoritmos mencionados.

IV. RESULTADOS

Para la evaluación experimental, se utilizó el mismo conjunto de datos de dominio público² presentado junto a Voxgraph. Este conjunto de datos corresponde a cuatro vuelos realizados por un hexacóptero Micro Aerial Vehicle (MAV) equipado con un LiDAR Ouster OS1, cada uno con una trayectoria aproximada de 400m, alrededor de un área de desastre diseñada para entrenar rescatistas. El ground-truth de campo se generó a partir de un sistema RTK-GNSS.

²Disponible en: http://robotics.ethz.ch/~asl-datasets/2020_voxgraph_arche

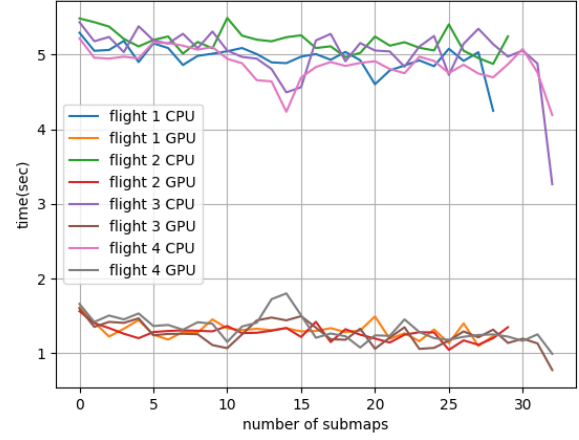


Figura 6: Tiempo de ejecución del módulo de integración de la nube de puntos del sensor (front-end) para cuatro vuelos distintos.

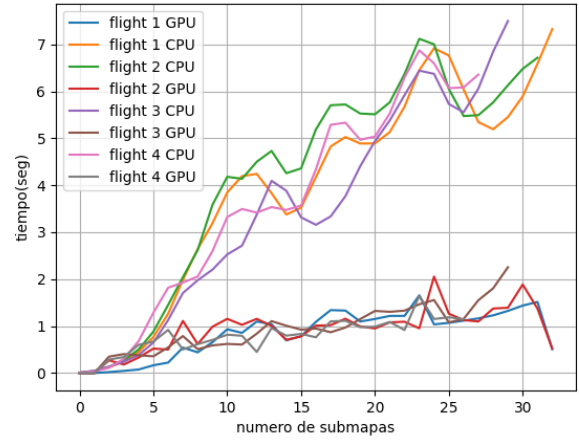


Figura 7: Tiempo de ejecución de módulo de aproximación por cuadrados mínimos (back-end) para cuatro vuelos distintos.

En primer lugar se analizaron los tiempos de ejecución del front-end y back-end en ambas versiones, la original basada en CPU y la acelerada en GPU. Las unidades de procesamiento utilizadas para realizar los experimentos fueron una PC de escritorio estándar (CPU con un procesador AMD Ryzen 9 5950x y una GPU Nvidia GeForce RTX 2060). Como puede verse en la Fig. 6 el tiempo de ejecución del front-end (el integrador de nubes de puntos) se mantiene relativamente estable porque el tamaño de la entrada no varía de una instancia de ejecución a otra, por otro lado la mejora introducida por coVoxgraph supera a la versión original por 4x. Como se puede ver en la Fig 7 el tiempo de ejecución para el proceso de back-end aumenta junto con el número de submapas superpuestos. Esto ocurre porque el número de bloques por submapa se distribuye homogéneamente, y por lo tanto también el número de vóxeles o píxeles relevantes precalculados cada vez que se termina un submapa. En este

caso, la versión acelerada por GPU supera a la versión original hasta en $8\times$ al final del vuelo.

Para evaluar la estimación de la trayectoria de Voxgraph y coVoxgraph, se realizó una comparación contra el ground-truth utilizando el error cuadrático medio (RMSE) del Error Absoluto de Trayectoria (ATE). Las mediciones de RMSE calculadas siguiendo [30] se pueden comparar en la Tabla I. Como se esperaba, tanto la versión original como la versión basada en GPU resultan en un RMSE muy similar.

Flight Version	Voxgraph	coVoxgraph
1	1.01	1.26
2	0.78	0.42
3	1.16	0.92
4	0.54	1.95

Cuadro I: Raíz del error cuadrático medio (metros) del Error Absoluto de Trayectoria (ATE) para Voxgraph y coVoxgraph.

V. CONCLUSIONES Y TRABAJO FUTURO

En este trabajo presentamos un novedoso sistema acelerado por GPU denominado coVoxgraph para construir mapas volumétricos globalmente consistentes en tiempo real basados en Voxgraph. Tanto el front-end como el back-end fueron rediseñados para aprovechar al máximo el poder de procesamiento paralelo de la GPU. Los resultados experimentales, llevados a cabo utilizando los mismos conjuntos de datos de Voxgraph, demuestran que la implementación presentada es hasta $8\times$ veces más rápida que la original al ejecutarse en una PC de escritorio. En todos los experimentos, la precisión obtenida fue la misma que los resultados generados por Voxgraph. Como trabajo futuro llevaremos a cabo un estudio de rendimiento del nuevo sistema coVoxgraph en placas GPU embebidas tales como la Jetson Xavier AGX que pueden ser montadas a bordo de pequeños robots móviles aéreos y terrestres. De esta forma buscamos analizar la factibilidad de alcanzar un sistema de SLAM para mapeo volumétrico globalmente consistente, que pueda ser ejecutado en tiempo real a bordo de estos robots.

REFERENCIAS

- [1] R. Mur-Artal and J. D. Tardós, “Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras,” *IEEE transactions on robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.
- [2] M. Labbé and F. Michaud, “Rtab-map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation,” *Journal of field robotics*, vol. 36, no. 2, pp. 416–446, 2019.
- [3] A. Cramariuc, L. Bernreiter, F. Tschopp, M. Fehr, V. Reijgwart, J. Nieto, R. Siegwart, and C. Cadena, “maplab 2.0—a modular and multi-modal mapping framework,” *IEEE Robotics and Automation Letters*, vol. 8, no. 2, pp. 520–527, 2022.
- [4] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, “Dtam: Dense tracking and mapping in real-time,” in *2011 international conference on computer vision*. IEEE, 2011, pp. 2320–2327.
- [5] C. Kerl, J. Sturm, and D. Cremers, “Dense visual slam for rgb-d cameras,” in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 2100–2106.
- [6] T. Whelan, S. Leutenegger, R. Salas-Moreno, B. Glocker, and A. Davison, “Elasticfusion: Dense slam without a pose graph,” *Robotics: Science and Systems*, 2015.
- [7] J. Czarowski, T. Laidlow, R. Clark, and A. J. Davison, “Deepfactors: Real-time probabilistic dense monocular slam,” *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 721–728, 2020.

- [8] L. Koestler, N. Yang, N. Zeller, and D. Cremers, “Tandem: Tracking and dense mapping in real-time using deep multi-view stereo,” in *Conference on Robot Learning*. PMLR, 2022, pp. 34–45.
- [9] X. Yang, H. Li, H. Zhai, Y. Ming, Y. Liu, and G. Zhang, “Vox-fusion: Dense tracking and mapping with voxel-based neural implicit representation,” in *2022 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 2022, pp. 499–507.
- [10] B. Curless and M. Levoy, “A volumetric method for building complex models from range images,” in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, 1996, pp. 303–312.
- [11] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon, “Kinectfusion: Real-time dense surface mapping and tracking,” in *2011 10th IEEE international symposium on mixed and augmented reality*. Ieee, 2011, pp. 127–136.
- [12] N. Fioraio, J. Taylor, A. Fitzgibbon, L. Di Stefano, and S. Izadi, “Large-scale and drift-free surface reconstruction using online subvolume registration,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 4475–4483.
- [13] V. Reijgwart, A. Millane, H. Oleynikova, R. Siegwart, C. Cadena, and J. Nieto, “Voxgraph: Globally consistent, volumetric mapping using signed distance function submaps,” *IEEE Robotics and Automation Letters*, vol. 5, no. 1, pp. 227–234, 2019.
- [14] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto, “Voxblox: Incremental 3d euclidean signed distance fields for on-board map planning,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 1366–1373.
- [15] M. Herlihy, N. Shavit, V. Luchangco, and M. Spear, *The Art of Multiprocessor Programming*. Elsevier Science, 2020. [Online]. Available: <https://books.google.com.ar/books?id=7MqCBAAQBAJ>
- [16] S. Aldegheri, N. Bombieri, D. D. Bloisi, and A. Farinelli, “Data flow orb-slam for real-time performance on embedded gpu boards,” in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 5370–5375.
- [17] R. Giubilato, S. Chiodini, M. Pertile, and S. Debei, “An evaluation of ros-compatible stereo visual slam methods on a nvidia jetson tx2,” *Measurement*, vol. 140, pp. 161–170, 2019.
- [18] J. Song, J. Wang, L. Zhao, S. Huang, and G. Dissanayake, “Mislslam: Real-time large-scale dense deformable slam system in minimal invasive surgery based on heterogeneous computing,” *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 4068–4075, 2018.
- [19] T. Schops, T. Sattler, and M. Pollefeys, “Bad slam: Bundle adjusted direct rgb-d slam,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 134–144.
- [20] J. Behley and C. Stachniss, “Efficient surfel-based slam using 3d laser range data in urban environments,” in *Robotics: Science and Systems*, vol. 2018, 2018, p. 59.
- [21] K. Koide, M. Yokozuka, S. Oishi, and A. Banno, “Globally consistent 3d lidar mapping with gpu-accelerated gicp matching cost factors,” *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 8591–8598, 2021.
- [22] L. Nardi, B. Bodin, M. Z. Zia, J. Mawer, A. Nisbet, P. H. Kelly, A. J. Davison, M. Luján, M. F. O’Boyle, G. Riley *et al.*, “Introducing slambench, a performance and accuracy benchmarking methodology for slam,” in *2015 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2015, pp. 5783–5790.
- [23] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger, “Real-time 3d reconstruction at scale using voxel hashing,” *ACM Transactions on Graphics (ToG)*, vol. 32, no. 6, pp. 1–11, 2013.
- [24] L. Buckley, J. Byrne, and D. Moloney, “Investigating the impact of suboptimal hashing functions,” in *2018 IEEE Games, Entertainment, Media Conference (GEM)*. IEEE, 2018, pp. 324–331.
- [25] D. Jünger, R. Kobus, A. Müller, C. Hundt, K. Xu, W. Liu, and B. Schmidt, “Warpcore: A library for fast hash tables on gpus,” in *2020 IEEE 27th international conference on high performance computing, data, and analytics (HiPC)*. IEEE, 2020, pp. 11–20.
- [26] W. Dong, Y. Lao, M. Kaess, and V. Koltun, “Ash: A modern framework for parallel spatial hashing in 3d perception,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 5, pp. 5417–5435, 2022.
- [27] E. E. Santos, “Optimal and efficient algorithms for summing and prefix summing on parallel machines,” *Journal of Parallel and Distributed Computing*, vol. 62, no. 4, pp. 517–543, 2002. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0743731500916988>

- [28] R. W. Larsen and T. Henriksen, "Strategies for regular segmented reductions on gpu," in *Proceedings of the 6th ACM SIGPLAN International Workshop on Functional High-Performance Computing*, ser. FHPC 2017. New York, NY, USA: Association for Computing Machinery, 2017, p. 42–52. [Online]. Available: <https://doi.org/10.1145/3122948.3122952>
- [29] J. Nocedal and S. Wright, *Numerical Optimization*, ser. Springer Series in Operations Research and Financial Engineering. Springer New York, 2006. [Online]. Available: <https://books.google.com.ar/books?id=VbHYoSyeIFcC>
- [30] Z. Zhang and D. Scaramuzza, "A tutorial on quantitative trajectory evaluation for visual (-inertial) odometry," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 7244–7251.

Sesión 4: Diseño, arquitecturas, modelado y control I

Autonomous Navigation and Collision Avoidance for Mobile Robots: Classification and Review

Marcus V. L. Carvalho
PPGEE
Poli USP
São Paulo, Brazil
marcusvini178@usp.br

Roberto Simoni
POSECM - UFSC
UFSC
Joinville, Brazil
roberto.simoni@ufsc.br

Leopoldo R. Yoshioka
PPGEE
Poli USP
São Paulo, Brazil
leopoldo.yoshioka@usp.br

Abstract—This paper introduces a novel classification for Autonomous Mobile Robots (AMRs), into three phases and five steps, focusing on autonomous collision-free navigation. Additionally, it presents the main methods and widely accepted technologies for each phase of the proposed classification. The purpose of this classification is to facilitate understanding and establish connections between the independent input variables of the system (hardware, software) and autonomous navigation. By analyzing well-established technologies in terms of sensors and methods used for autonomous navigation, this paper aims to provide a foundation of knowledge that can be applied in future projects of mobile robots.

Keywords—Autonomous Mobile Robots, Navigation, Sensors, Methods, Obstacle Avoidance

I. INTRODUCTION

Autonomous Mobile Robots (AMRs) are becoming increasingly essential in various sectors. They assist humans in performing complex, hazardous, or repetitive tasks. Initially created to improve productivity and safety in industrial settings, their scope has significantly broadened. From initially focusing on path planning for industrial manipulators [1], AMRs now use advanced algorithms to navigate without collisions. This expansion has allowed them to operate in diverse and dynamic environments beyond just industrial settings [2], [3].

Despite considerable advancements, existing navigational strategies for Autonomous Mobile Robots (AMRs) often remain focused on specific domains: terrestrial, aerial, and aquatic. These strategies typically adopt layered approaches from perception to control, each tailored to distinct operational environments such as industrial settings [4], uneven terrains [5], [6], and underwater exploration [7], [8]. All these applications suggest a lack of a unified framework that can seamlessly be integrated across all domains, a gap this paper aims to address. By adopting modular packages, the proposed classification enhances the reusability and interoperability of components, facilitating easier integration across all domains of autonomous navigation [9], [10].

This paper introduces a new, comprehensive classification system aimed at streamlining the various aspects of autonomous navigation. The system acts as a fundamental framework, organizing the intricate relationships between phases, modules, and layers. It improves the comprehension and execution of autonomous navigation strategies, offering clear

insights, and ultimately offering a complete set of tools for practitioners to choose the best solution for a wide range of operational scenarios.

The paper is organized as follows: Section II presents the methodology and the process undertaken to develop our classification and review of components and technologies. Section III outlines the unified classification. Section IV discusses technological integrations and their applicability within various domains. Section V explores potential future directions and innovations, and Section VI concludes with key findings and implications for future research.

II. METHODOLOGY FOR LITERATURE REVIEW

This study applied a systematic literature review approach to summarize the existing classifications and technologies in the field of autonomous navigation. In parallel with the analytical rigor demonstrated by [11] in the autonomous vehicle domain. We aim to systematically identify and categorize significant contributions across the spectrum of autonomous mobile robots.

The objectives were to identify key classifications of autonomous navigation and analyze the integration of hardware, software, and the robot-environment dynamic.

The research criteria focused on:

- Peer-reviewed papers that provided foundational insights and demonstrated long-term impact in the field.
- Clear frameworks for autonomous navigation, aiming to bridge foundational theories with contemporary advancements.
- Practical applications of technologies with significant developments in autonomous navigation.

Searches were conducted across major databases such as IEEE Xplore, ScienceDirect, Web of Knowledge, and SpringerLink using keywords like "autonomous navigation," "robot classification," and "obstacle avoidance." This strategic approach facilitated the inclusion of seminal, consolidated, and cutting-edge contributions.

This literature review allows to propose a novel classification intended to bridge existing gaps and facilitate future research in the field of autonomous navigation.

III. A UNIFIED CLASSIFICATION FOR DIVERSE AUTONOMOUS NAVIGATION APPLICATIONS

This study has synthesized techniques, methodologies, and technologies for autonomous navigation, focusing on the dynamic interaction between hardware, software, and the robot environment. Based on this synthesis, we propose a unique classification system comprising three layers and five interconnected phases for autonomous navigation. The details of this classification system can be found on Figure 1.

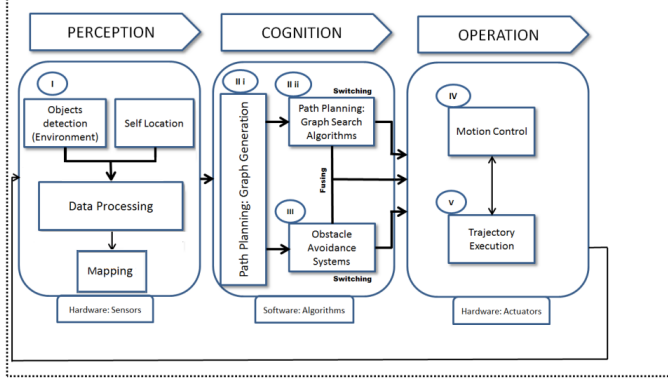


Fig. 1. Layers and phases of autonomous navigation.

This classification interconnects the layers and phases as follows:

- 1) Layer 1 - Perception
 - a) Phase I: Environment Perception, Self Location, Data Processing, and Mapping.
- 2) Layer 2 - Cognition
 - a) Phase II: Path Planning, including Graph Construction and Graph Search.
 - b) Phase III Obstacle Avoidance and Trap landscape.
- 3) Layer 3 - Operation
 - a) Phase IV Motion control,
 - b) Phase V Path Execution

The phases are not isolated but interconnected, covering the perception, cognition, and operation layers of mobile robotics. For instance, in the perception layer, mapping is integrated with data processing to generate a high-level environment representation, which is then used in the cognition layer for path planning and obstacle avoidance. This pre-map is then refined in the cognition layer to incorporate detailed terrain characteristics and navigability information. In the cognition layer, adaptive behavior is facilitated by integrating a Graph Search Algorithm with a Collision Avoidance System (CAS). This integration optimizes path planning to work in harmony with motion control, ultimately leading to the execution of a seamless trajectory.

A. Phase I Environment Perception, Self Location, and Data Processing.

To initiate autonomous navigation, the robot must recognize its surrounding environment. This involves using different

sensors to collect data, which is then processed to create an initial pre-map. Mapping algorithms like occupancy grid mapping [12], SLAM [2], and topological mapping generate this high-level representation of the environment. The pre-map distinguishes navigable and non-navigable areas and, plays a crucial role in subsequent stages, enabling the robot to determine its current location and plan a path to reach its destination.

Since autonomous robots rely on multiple sensors to perceive their environment, it is important to use filtering techniques to merge and refine collected data from these sensors.

Table I shows well-established sensors, while table II shows the most common filters used for data acquisition in environment perception, self-location, data processing, and mapping.

TABLE I
PHASE I - WELL-ESTABLISHED SENSORS.

Sensors	Reference
Geo-referencing Systems	
Inertial Navigation System (INS): IMU, Gyroscope, Compass, Altimeter	[2], [13]
Attitude and Heading Reference System (AHRS): MEMS Gyroscopes, Accelerometers, Magnetometers.	[14], [15]
Self Location Apparatus (for Dead Reckoning estimation)	
Odometer, Encoder	[16]–[20]
Optical Encoder	[21]
Ultrasonic Sensor	[22]
Electromagnetic Waves Based Devices	
Radar	[14], [15], [23]
Ground Penetrating Radar (GPR)	[13], [14], [24], [25]
Global Positioning System (GPS) GPS and/or DGPS	[10], [13], [15], [16], [21], [23], [26]
Cooperative Location Sharing Devices	
Automatic dependent surveillance-broadcast (ADS-B), Zigbee, Wireless.	[16], [19]
PetriNet Model	[21]
Ground beacons based Position Locators Apparatus	
Radio Frequency Identification (RFID)	[27], [28]
Bluetooth wireless	[18]
Laser Rangefinders (light waves propagation)	
LiDAR	[4], [13]–[17], [19], [21], [23], [26], [29], [30]
Infrared Sensor	[21], [31]
Camera Visual Sensors	
Kinect (Depth), RGB Camera	[4], [13], [15], [16], [18], [23], [26], [30], [32], [33] [29]

TABLE II
PHASE I - COMMON FILTERS.

Filters	Reference
Multi Sensor Fusion based Filters	
Kalman Filter	[16], [23]
Extended Kalman Filter (EKF)	[4], [29]
Vision System Filters	
High Dynamic Range (HDR) Algorithms	[33]
Gaussian-based filters	[9], [34]
Bayesian-based filters	[22], [28]

This paper does not explore sensor data processing libraries and localization methods. However, it is important to men-

tion that established tools such as YOLO and OpenCV are widely used for detection and localization in the robotics Perception layer [35]. Visual SLAM algorithms [36], [37] and localization methods like Iterative Closest Point (ICP) [5] and Normal Distributions Transform (NDT) [35] scan matching are also crucial in this layer. Segmentation techniques, utilizing Gaussian-based models [34], further, enhance localization and visual navigation by accurately classifying navigable paths and reducing spatial requirements for map storage.

B. Phase IIA - Path Planning: Graph Construction

Once the perception processing unit in the perception layer extracts meaningful data and creates an initial pre-map using sensor data, the cognition layer refines it. Techniques like surfel-based mapping [30], Delaunay triangulation, and visibility constraints refine the map into a dense 3D representation [37], ensuring accuracy and navigability. This detailed mapping allows the cognition layer to plan precise paths. Table III presents usual map-building techniques for autonomous navigation.”

TABLE III
PHASE IIA PATH PLANNING: GRAPH CONSTRUCTION

Method	Reference
Graph Search Maps	
Voronoi Diagram	[38], [39]
Exact Cell Decomposition	[9], [39]
Height Segmented Map	[40]
Surfel-Based Map	[30]
Approximate Cell Decomposition	[13], [14], [17], [23], [39]
Lattice Graph	[41]–[44]
Potential Field Maps	
Extended Potential Field Approach	[39], [45], [46]
Others Methods of Map Building	
Genetic Algorithm (GA)	[16], [39]
CNN Feature Map	[47]
Spatio-Temporal Voxel Layer (STVL)	[4]
Dense 3D Mapping	[5], [6], [37]

As shown in Table III, the Graph Search technique is frequently applied in AMRs, particularly within structured environments like indoor settings. This method effectively utilizes predefined grid or mesh maps for precise navigation.

C. Phase IIB - Path Planning: Graph Search Algorithms

Once the robot knows its position, environment features, and target, it starts path planning. Roboticians focus on two main parameters in this phase, as noted in [48]:

- *completeness*: The ability to find a solution within a finite time.
- *optimality*: The ability to compute the most efficient path considering time, energy, or distance. Various strategies for achieving these goals are extensively discussed [39].

Table IV presents well-established path-planning algorithms that have been applied in recent works on autonomous navigation.

For path planning, autonomous vehicles mainly use the A* algorithm and its variants. Traditional methods like BFS and

TABLE IV
PHASE IIB PATH PLANNING: GRAPH SEARCH ALGORITHMS

Trajectory Generation	Reference
Deterministic Graph Search	
Breadth-First Search (BFS)	[9], [35], [49]
Depth-First Search (DFS)	[9]
Dijkstra’s Algorithm	[35], [50], [51]
A* Algorithm	[14], [35], [38], [40]
D*Algorithm	[17], [43]
Smac Planner	[52]
Randomized Graph Search	
Rapidly Exploring Random Tree (RRT)	[5], [35], [53]
Spline Sample RRT*	[26]
Probabilistic Roadmap	[5], [18]
OMPL and SBO Planners	[5]
Derived Algorithms from the previous graph search methods	
Potential Field based Algorithms	[13], [54], [55]
Artificial Potential Field (APF)	[19], [56]
Spline Path Planning	[29]
Fuzzy Heuristic Search	[7], [31]
Firefly Algorithm (FA)	[21]
High Autonomous Driving (HAD) Algorithms	[57]
Smoothed A* Algorithm	[58]

DFS are less preferred due to their inefficiency in optimizing paths. Instead, roboticists develop customized heuristic methods that balance path optimality and computational costs, addressing processing power and decision time constraints [59].

D. Phase III: Obstacle Avoidance and Trap Landscapes

During autonomous operations, mobile robots must navigate around obstacles. If not programmed in the initial phases, collision avoidance systems (CAS) become crucial. These systems are adapted to the robot’s operational environment and kinodynamics, ensuring safe maneuverability. Table V details established CAS algorithms, reflecting their varied response times, safety distances, and specific functionalities.

Collision Avoidance Systems (CAS) play a vital role in ensuring the safe and effective operation of mobile robots in dynamic environments. By integrating these systems, robots are equipped to dynamically navigate through complex terrains, thereby enhancing their reliability and operational scope for real-world applications.

E. Phase IV: Motion Control And Robot Relocation

To control the movement, speed, position, and orientation of the AMR, various controllers are integrated into the robot, addressing both hardware and software requirements. Vehicles face unique constraints and require specific accuracy and response times due to differing maneuverability capabilities and variable environments. A range of controllers have been developed to meet these needs, as shown in Table VI.

From Table VI, it is evident that complex vehicle dynamics and rapidly changing environments necessitate the use of multiple controller types. For example, [53] and [79] combined nonlinear controllers with PID controllers to enhance steering and speed regulation. Modern approaches increasingly incorporate nonlinear controllers to address complex differential equations more effectively. Additionally, the use of cooperative

TABLE V
PHASE III: OBSTACLE AVOIDANCE AND TRAP LANDSCAPE, /CASS.

STRATEGIES CASS	References
Traditional Algorithms	
Bug Algorithms	[31], [60]–[62]
Vector Field Histogram (VFH)	[13], [45]
VFH+	[63]
VFH*	[64]
The Bubble Band Technique	[14], [26], [65]
Elastic Band Concept	[66]
Curvature Velocities Techniques (CVM)	[47], [67]
Dynamic Windows Approaches	[23], [68]–[70]
The Schlegel Approach	[71]
Nearness Diagram	[17], [72], [73]
Virtual Force Field (VFF) Methods	
Gradient Methods	[31], [56], [74]
Bacterial Potential Field	[54]
Genetic based Algorithms	
Biological Approach	[75]
Bioinspired Neural Network Algorithm	[31], [76]
Hybrid VFF-Genetic Algorithms	
Evolutionary Behaviour based on Genetic Programming	[55]
Geometrical Methods	
Boundary Following	[31]
Collision Cone	[18]
Higher Geometry Maze Routing Algorithm	[49]
Fuzzy / Neurofuzzy Relational Products	[77]
Anti-target Approach Laws	
Cone's Geometry-based Calculated Rule	[78]

TABLE VI
PHASE IV MOTION CONTROL AND ROBOT RELOCATION

Controllers	References
Control-Theory Based Controllers	
Nonlinear Controllers	
Time Elastic Band	[4]
Nonlinear Optimal SDRE	[56]
Pure Pursuit	[53], [79], [80]
Linear Controllers	
Lane Detection and Sliding Mode	[31], [81]
PID (Pose / Velocity)	[13], [14], [79]
Model Predictive Control (MPC)	[6], [31]
Hybrid Controllers	
PID (Pose / Velocity)	[81]
Model Predictive Path Integral Control (MPPI)	[82]
Behaviour Based Controllers	
Machine Learning	
Matlab/hardware Loop	[57]
Receding Horizon (CNN)	[47]
Relocation Techniques And Others Sorts of Controllers	
SLAM	[23]
Hybrid	[55]

predictive controllers has become prevalent, optimizing robot relocation and enhancing motion control efficiency [23].

F. Phase V: Trajectory Execution

AMRs are equipped with specialized hardware to execute planned routes. Some adhere to predefined trajectories (offline trajectory execution), while others use episodic planning to dynamically integrate planning and execution using sensor data [9]. Additionally, AMRs may include real-time replanning modules within the operation layer, eliminating the temporal gap between planning and execution. Behavior trees [83] coordinate diverse planning and execution modules to man-

age dynamic conditions effectively. This approach increases flexibility, enabling AMRs to adapt to varying environments and operational demands. The distinction between offline and online path planning underscores the necessity for dynamic response capabilities in online scenarios, where unpredictable environments pose significant computational challenges [49]. Table VII highlights commonly used methods and algorithms for trajectory execution.

TABLE VII
PHASE V: TRAJECTORY EXECUTION.

Method/Algorithm	References
Offline Planning (Dead Reckoning)	[13], [25], [54], [84]
Episodic Planning (Deferred Planning)	[14], [20], [55], [85]–[87]
Integrated Planning and Execution (Real Time Replanning)	[15], [21], [26], [31], [40], [42]–[44], [47], [53], [57], [83]
Hybrid Layers Switching	[8], [50]

IV. DISCUSSION

The proposed classification of autonomous navigation and the discussion of both well-established and recent techniques and technologies not only contribute to academic understanding but also provide a practical guide for the design and implementation of future mobile robots. This framework also facilitates learning about the various algorithms, components, and sensors that compose autonomous navigation, ranging from perception to operation.

V. TENDENCIES

Recent advancements in Autonomous Mobile Robots (AMRs) underscore trends that enhance their operational capabilities:

- **Enhanced Communication Technologies:** Introduction of 5G technology significantly improves real-time data sharing among AMRs, crucial for industrial automation.
- **Evolution of Navigation Algorithms:** The adoption of evolutionary algorithms and neural networks bolsters real-time decision-making in AMRs.
- **Artificial Intelligence in Cognition:** Integration of deep learning enhances path prediction and decision-making, facilitating advanced cognitive functions.
- **Advanced Sensor Fusion and Data Processing:** Enhanced filtering techniques and new algorithms improve environmental perception and motion planning under challenging conditions.
- **Sensor Diversity and Data Redundancy:** The use of diverse sensors and advanced probabilistic filters increases data reliability, optimizing tasks like SLAM.
- **Nonlinear Filtering Techniques:** Nonlinear filters are critical for accurately processing the complex dynamics of AMRs, enhancing motion control and trajectory prediction.
- **Adaptability Across Environments:** AMRs have advanced in operating across varied environments: terrestrial, aerial, and aquatic, broadening their applications to include disaster response and environmental monitoring.

These innovations highlight the dynamic evolution of AMR technology, where flexibility and improved communication are key drivers of enhanced robotic navigation.

VI. CONCLUSION

This paper presented a novel classification system for the domain of autonomous mobile robots (AMRs), aiming to refine the interconnection between diverse technologies for autonomous navigation. By exploring various techniques (such as methodologies, methods, and strategies) and technologies (including sensors, tools and filters) readers can develop deeper understanding of the different phases of autonomous navigation and they can make informed choices of well-established tools for designing mobile robots.

ACKNOWLEDGMENT

This research was supported by Fundação de Amparo à Pesquisa e Inovação do Estado de Santa Catarina (FAPESC).

REFERENCES

- [1] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *Int. J. Robot.*, vol. 5, no. 1, pp. 90–98, 1986.
- [2] P. K. Panigrahi and S. K. Bisoy, "Localization strategies for autonomous mobile robots: A review," *J. King Saud Univ., Comp. Info. Sci.*, vol. 34, no. 8, pp. 6019–6039, 2022.
- [3] J. R. Sánchez-Ibáñez, C. J. Pérez-del Pulgar, and A. García-Cerezo, "Path planning for autonomous mobile robots: A review," *Sensors*, vol. 21, no. 23, p. 7898, 2021.
- [4] S. Macenski, F. Martin, R. White, and J. Ginés Clavero, "The marathon 2: A navigation system," in *2020 IEEE/RSJ IROS*, 2020.
- [5] F. Atas, L. Grimstad, and G. Cielniak, "Evaluation of sampling-based optimizing planners for outdoor robot navigation," *arXiv preprint arXiv:2103.13666*, 2021.
- [6] F. Atas, G. Cielniak, and L. Grimstad, "Navigating in 3d uneven environments through supervoxels and nonlinear mpc," in *2023 ECMR*, IEEE, 2023, pp. 1–8.
- [7] Y.-i. Lee, Y.-G. Kim, and L. J. Kohout, "An intelligent collision avoidance system for auvs using fuzzy relational products," *Inf. Sci.*, vol. 158, pp. 209–232, 2004.
- [8] G. Antonelli, S. Chiaverini, R. Finotello, and R. Schiavon, "Real-time path planning and obstacle avoidance for rais: an autonomous underwater vehicle," *IEEE J Ocean Eng.*, vol. 26, no. 2, pp. 216–227, 2001.
- [9] R. Siegwart, I. R. Nourbakhsh, D. Scaramuzza, and R. C. Arkin, *Introduction to autonomous mobile robots*, 2011.
- [10] S. S. H. Hajjaj and K. S. M. Sahari, "Bringing ros to agriculture automation: hardware abstraction of agriculture machinery," *Int. J. Appl. Eng. Res.*, vol. 12, no. 3, pp. 311–316, 2017.
- [11] L. Chen, Y. Li, C. Huang, B. Li, Y. Xing, D. Tian, L. Li, Z. Hu, X. Na, Z. Li *et al.*, "Milestones in autonomous driving and intelligent vehicles: Survey of surveys," *IEEE Trans. Intell. Veh.*, vol. 8, no. 2, pp. 1046–1056, 2022.
- [12] J. Esenkanova, H. O. İlhan, and S. Yavuz, "Pre-mapping system with single laser sensor based on gmapping algorithm," *IJOEE*, vol. 1, no. 2, pp. 97–101, 2013.
- [13] Y. He, C. Chen, C. Bu, and J. Han, "A polar rover for large-scale scientific surveys: design, implementation and field test results," *Int. J. Adv. Robot. Syst.*, vol. 12, no. 10, p. 145, 2015.
- [14] C. Urmson, J. Anhalt, D. Bartz, M. Clark, T. Galatali, A. Gutierrez, S. Harbaugh, J. Johnston, P. Koon, W. Messner *et al.*, "A robust approach to high-speed navigation for unrehearsed desert terrain," in *The 2005 DARPA Grand Challenge*. Springer, 2007, pp. 45–102.
- [15] S. Noh, "Decision-making framework for autonomous driving at road intersections: Safeguarding against collision, overly conservative behavior, and violation vehicles," *IEEE Trans. Ind. Electron.*, vol. 66, no. 4, pp. 3275–3286, 2019.
- [16] H. Li, M. Tsukada, F. Nashashibi, and M. Parent, "Multivehicle cooperative local mapping: A methodology based on occupancy grid map merging," *IEEE T-ITS*, vol. 15, no. 5, pp. 2089–2100, 2014.
- [17] I. Iturrate, J. M. Antelis, A. Kubler, and J. Minguez, "A noninvasive brain-actuated wheelchair based on a p300 neurophysiological protocol and automated navigation," *IEEE Trans. Robot.*, vol. 25, no. 3, pp. 614–627, 2009.
- [18] J. Snape, J. Van Den Berg, S. J. Guy, and D. Manocha, "The hybrid reciprocal velocity obstacle," *IEEE Trans. Robot.*, vol. 27, no. 4, pp. 696–706, 2011.
- [19] H. Li and A. V. Savkin, "An algorithm for safe navigation of mobile robots by a sensor network in dynamic cluttered industrial environments," *Robot. Comput.-Integr. Manuf.*, vol. 54, pp. 65–82, 2018.
- [20] A. C. Schultz and W. Adams, "Continuous localization using evidence grids," in *Proceedings. 1998 ICRA (Cat. No. 98CH36146)*, vol. 4. IEEE, 1998, pp. 2833–2839.
- [21] B. Patle, A. Pandey, A. Jagadeesh, and D. Parhi, "Path planning in uncertain environment by using firefly algorithm," *Def. Technol.*, vol. 14, no. 6, pp. 691–701, 2018.
- [22] P. Yang, "Efficient particle filter algorithm for ultrasonic sensor-based 2d range-only simultaneous localisation and mapping application," *IET Wireless Sensor Systems*, vol. 2, no. 4, pp. 394–401, 2012.
- [23] M. Aeberhard, S. Rauch, M. Bahram, G. Tanzmeister, J. Thomas, Y. Pilat, F. Himm, W. Huber, and N. Kaempchen, "Experience, results and lessons learned from automated driving on germany's highways," *IEEE T-ITS*, vol. 7, no. 1, pp. 42–57, 2015.
- [24] E. Trautmann, L. Ray, and J. Lever, "Development of an autonomous robot for ground penetrating radar surveys of polar ice," in *2009 IEEE/RSJ IROS*. IEEE, 2009, pp. 1685–1690.
- [25] L. Ray, A. Adolph, A. Morlock, B. Walker, M. Albert, J. H. Lever, and J. Dibb, "Autonomous rover for polar science support and remote sensing," in *2014 IEEE IGARSS*. IEEE, 2014, pp. 4101–4104.
- [26] S. Yoon, D. Lee, J. Jung, and D. H. Shim, "Spline-based rrt using piecewise continuous collision-checking algorithm for car-like vehicles," *J. Intell. Robot. Syst.*, vol. 90, no. 3–4, pp. 537–549, 2018.
- [27] W. Gueaieb and M. S. Miah, "An intelligent mobile robot navigation technique using rfid technology," *IEEE T Instrum. Meas.*, vol. 57, no. 9, pp. 1908–1917, 2008.
- [28] J. Zhang, Y. Lyu, J. Patton, S. C. Periaswamy, and T. Roppel, "Bfvp: A probabilistic uhf rfid tag localization algorithm using bayesian filter and a variable power rfid model," *IEEE Trans. Ind. Electron.*, vol. 65, no. 10, pp. 8250–8259, 2018.
- [29] H. Xiao, Z. Li, C. Yang, W. Yuan, and L. Wang, "Rgb-d sensor-based visual target detection and tracking for an intelligent wheelchair robot in indoors environments," *IJCAS*, vol. 13, no. 3, pp. 521–529, 2015.
- [30] F. Atas, G. Cielniak, and L. Grimstad, "Elevation state-space: Surfel-based navigation in uneven environments for mobile robots," in *2022 IEEE/RSJ IROS*. IEEE, 2022, pp. 5715–5721.
- [31] M. Hoy, A. S. Matveev, and A. V. Savkin, "Algorithms for collision-free navigation of mobile robots in complex cluttered environments: a survey," *Robotica*, vol. 33, no. 3, pp. 463–497, 2015.
- [32] Z. Yan, J. Li, Y. Wu, and G. Zhang, "A real-time path planning algorithm for auv in unknown underwater environment based on combining pso and waypoint guidance," *Sensors*, vol. 19, no. 1, p. 20, 2019.
- [33] N. Paul and C. Chung, "Application of hdr algorithms to solve direct sunlight problems when autonomous vehicles using machine vision systems are driving into sun," *Comput Ind.*, vol. 98, pp. 192–196, 2018.
- [34] P. De Cristóforis, M. Nitsche, T. Krajník, T. Pire, and M. Mejail, "Hybrid vision-based navigation for mobile robots in mixed indoor/outdoor environments," *Pattern Recognit. Lett.*, vol. 53, pp. 118–128, 2015.
- [35] M. V. L. d. Carvalho *et al.*, "A review of ros based autonomous driving platforms to carry out automated driving functions," 2022.
- [36] T. Pire, T. Fischer, J. Civera, P. De Cristóforis, and J. J. Berles, "Stereo parallel tracking and mapping for robot localization," in *2015 IEEE/RSJ IROS*. IEEE, 2015, pp. 1373–1378.
- [37] Y. Ling and S. Shen, "Building maps for autonomous navigation using sparse visual slam features," in *2017 IEEE/RSJ IROS*. IEEE, 2017, pp. 1374–1381.
- [38] A. Sgorbissa and R. Zaccaria, "Planning and obstacle avoidance in mobile robotics," *Rob. Auton. Syst.*, vol. 60, no. 4, pp. 628–638, 2012.
- [39] A. Gasparetto, P. Boscariol, A. Lanzutti, and R. Vidoni, "Path planning and trajectory planning algorithms: A general overview," *Motion and Operation Planning of Robotic Systems: Background and Practical Approaches*, pp. 3–27, 2015.

- [40] P. Karkowski, S. Obwald, and M. Bennewitz, "Real-time footstep planning in 3d environments," in *2016 IEEE-RAS 16th. IEEE*, 2016, pp. 69–74.
- [41] M. Pivtoraiko, R. A. Knepper, and A. Kelly, "Differentially constrained mobile robot motion planning in state lattices," *J Field Robot.*, vol. 26, no. 3, pp. 308–333, 2009.
- [42] M. Ruffi, D. Ferguson, and R. Siegwart, "Smooth path planning in constrained environments," in *2009 ICRA*. IEEE, 2009, pp. 3780–3785.
- [43] M. Ruffi and R. Siegwart, "On the application of the d* search algorithm to time-based planning on lattice graphs," in *Proc. of The 4th ECMR*. Eidgenössische Technische Hochschule Zürich, 2009, pp. 105–110.
- [44] D. Ferguson, T. M. Howard, and M. Likhachev, "Motion planning in urban environments: Part ii," in *2008 IEEE/RSJ IROS*. IEEE, 2008, pp. 1070–1076.
- [45] J. Borenstein and Y. Koren, "Real-time obstacle avoidance for fast mobile robots," *IEEE Trans. Syst. Man Cybern.*, vol. 19, no. 5, pp. 1179–1187, 1989.
- [46] M. Khatib and R. Chatila, "An extended potential field approach for mobile robot sensor-based motions," in *ICoIAS*, 1995.
- [47] S. Yang, W. Wang, C. Liu, and W. Deng, "Scene understanding in deep learning-based end-to-end controllers for autonomous vehicles," *IEEE Trans. Syst. Man Cybern.: Systems*, vol. 49, no. 1, pp. 53–63, 2019.
- [48] Z. Zeng, L. Lian, K. Sammut, F. He, Y. Tang, and A. Lammas, "A survey on path planning for persistent autonomy of autonomous underwater vehicles," *Ocean Eng.*, vol. 110, pp. 303–313, 2015.
- [49] P. Raja and S. Pugazhenthii, "Optimal path planning of mobile robots: A review," *Int. J. Phys. Sci.*, vol. 7, no. 9, pp. 1314–1320, 2012.
- [50] G. Casalino, A. Turetta, and E. Simetti, "A three-layered architecture for real time path planning and obstacle avoidance for surveillance usvs operating in harbour fields," in *Oceans 2009-Europe*. IEEE, 2009, pp. 1–8.
- [51] H. Niu, Y. Lu, A. Savvaris, and A. Tsourdos, "Efficient path planning algorithms for unmanned surface vehicle," *IFAC-PapersOnLine*, vol. 49, no. 23, pp. 121–126, 2016.
- [52] S. Macenski, M. Booker, and J. Wallace, "Open-source, cost-aware kinematically feasible planning for mobile and surface robotics," *arXiv preprint arXiv:2401.13078*, 2024.
- [53] Y. Kuwata, J. Teo, G. Fiore, S. Karaman, E. Frazzoli, and J. P. How, "Real-time motion planning with applications to autonomous urban driving," *IEEE Trans. Control Syst. Technol.*, vol. 17, no. 5, pp. 1105–1118, 2009.
- [54] O. Montiel, U. Orozco-Rosas, and R. Sepúlveda, "Path planning for mobile robots using bacterial potential field for avoiding static and dynamic obstacles," *Expert Syst App.*, vol. 42, no. 12, pp. 5177–5191, 2015.
- [55] E. Clemente, M. Meza-Sánchez, E. Bugarin, and A. Y. Aguilar-Bustos, "Adaptive behaviors in autonomous navigation with collision avoidance and bounded velocity of an omnidirectional mobile robot," *J. Intell. Robot. Syst.*, vol. 92, no. 2, pp. 359–380, 2018.
- [56] S. M. H. Rostami, A. K. Sangaiah, J. Wang, and H.-j. Kim, "Real-time obstacle avoidance of mobile robots using state-dependent riccati equation approach," *EURASIP JIVP*, vol. 2018, no. 1, p. 79, 2018.
- [57] M. Bahram, Z. Ghandeharioun, P. Zahn, M. Baur, W. Huber, and F. Busch, "Microscopic traffic simulation based evaluation of highly automated driving on highways," in *17th ITSC*. IEEE, 2014, pp. 1752–1757.
- [58] R. Song, Y. Liu, and R. Bucknall, "Smoothed a* algorithm for practical unmanned surface vehicle path planning," *Appl. Ocean Res.*, vol. 83, pp. 9–20, 2019.
- [59] J. L. G. Rosa, "Fundamentos da inteligência artificial," 2011.
- [60] I. Kamon, E. Rivlin, and E. Rimón, "A new range-sensor based globally convergent navigation algorithm for mobile robots," in *Proceedings of ICRA*, vol. 1. IEEE, 1996, pp. 429–435.
- [61] V. J. Lumelsky and A. A. Stepanov, "Path-planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape," *Algorithmica*, vol. 2, no. 1-4, pp. 403–430, 1987.
- [62] V. J. Lumelsky and T. Skewis, "Incorporating range sensing in the robot navigation function," *IEEE Trans. Syst. Man Cybern.*, vol. 20, no. 5, pp. 1058–1069, 1990.
- [63] I. Ulrich and J. Borenstein, "VfH+: Reliable obstacle avoidance for fast mobile robots," in *Proceedings. 1998 ICRA (Cat. No. 98CH36146)*, vol. 2. IEEE, 1998, pp. 1572–1577.
- [64] —, "VfH/sup*: Local obstacle avoidance with look-ahead verification," in *Proceedings 2000 ICRA. Millennium Conference. ICRA. Symposia Proceedings (Cat. No. 00CH37065)*, vol. 3. IEEE, 2000, pp. 2505–2511.
- [65] M. Khatib, H. Jaouni, R. Chatila, and J.-P. Laumond, "Dynamic path modification for car-like nonholonomic mobile robots," in *Proc. - IEEE Int. Conf. Robot. Autom.*, vol. 4. IEEE, 1997, pp. 2920–2925.
- [66] S. Quinlan and O. Khatib, "Elastic bands: Connecting path planning and control," in *[1993] Proceedings ICRA*. IEEE, 1993, pp. 802–807.
- [67] R. Simmons, "The curvature-velocity method for local obstacle avoidance," in *Proceedings of ICRA*, vol. 4. IEEE, 1996, pp. 3375–3382.
- [68] J. Minguez, L. Montano, and O. Khatib, "Reactive collision avoidance for navigation with dynamic constraints," in *IEEE/RSJ IROS*, vol. 1. IEEE, 2002, pp. 588–594.
- [69] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robot Autom Mag.*, vol. 4, no. 1, pp. 23–33, 1997.
- [70] O. Brock and O. Khatib, "High-speed navigation using the global dynamic window approach," in *Proceedings 1999 ICRA (Cat. No. 99CH36288C)*, vol. 1. IEEE, 1999, pp. 341–346.
- [71] C. Schlegel, "Fast local obstacle avoidance under kinematic and dynamic constraints for a mobile robot," in *1998 IEEE/RSJ IROS. PI-TPA (Cat. No. 98CH36190)*, vol. 1. IEEE, 1998, pp. 594–599.
- [72] J. Minguez and L. Montano, "Nearness diagram (nd) navigation: collision avoidance in troublesome scenarios," *IEEE T-RO*, vol. 20, no. 1, pp. 45–59, 2004.
- [73] J. Minguez and L. Montano, "Robot navigation in very complex, dense, and cluttered indoor/outdoor environments," *IFAC Proceedings Volumes*, vol. 35, no. 1, pp. 397–402, 2002.
- [74] K. Konolige, "A gradient method for realtime robot control," in *Proceedings. 2000 IEEE/RSJ IROS 2000 (Cat. No. 00CH37113)*, vol. 1. IEEE, 2000, pp. 639–646.
- [75] C.-T. Chen, R. D. Quinn, and R. E. Ritzmann, "A crash avoidance system based upon the cockroach escape response circuit," in *Proc. - IEEE Int. Conf. Robot. Autom.*, vol. 3. IEEE, 1997, pp. 2007–2012.
- [76] X. Cao, D. Zhu, and S. X. Yang, "Multi-aUV target search based on bioinspired neurodynamics model in 3-d underwater environments," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 11, pp. 2364–2374, 2016.
- [77] C. S. Tzafestas and S. G. Tzafestas, "Recent algorithms for fuzzy and neurofuzzy path planning and navigation of autonomous mobile robots," in *1999 ECC*. IEEE, 1999, pp. 4736–4743.
- [78] A. Chakravarthy and D. Ghose, "Obstacle avoidance in a dynamic environment: A collision cone approach," *IEEE Trans. Syst. Man Cybern.-Part A: Systems and Humans*, vol. 28, no. 5, pp. 562–574, 1998.
- [79] Y. Chen and J. J. Zhu, "Pure pursuit guidance for car-like ground vehicle trajectory tracking," in *ASME 2017 DSCC*. American Society of Mechanical Engineers, 2017, pp. V002T21A015–V002T21A015.
- [80] S. Macenski, S. Singh, F. Martin, and J. Gines, "Regulated pure pursuit for robot path tracking," *Auton. Robots*, 2023.
- [81] X. Zhang and X. Zhu, "Autonomous path tracking control of intelligent electric vehicles based on lane detection and optimal preview method," *Expert Syst App.*, vol. 121, pp. 38–48, 2019.
- [82] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, "Aggressive driving with model predictive path integral control," in *2016 ICRA*. IEEE, 2016, pp. 1433–1440.
- [83] S. Macenski, T. Moore, D. V. Lu, A. Merzlyakov, and M. Ferguson, "From the desks of ros maintainers: A survey of modern & capable mobile robotics algorithms in the robot operating system 2," *Rob. Auton. Syst.*, vol. 168, p. 104493, 2023.
- [84] I. R. Nourbakhsh, J. Bobenage, S. Grange, R. Lutz, R. Meyer, and A. Soto, "An affective mobile robot educator with a full-time job," *Artificial intelligence*, vol. 114, no. 1-2, pp. 95–124, 1999.
- [85] R. Philippsen and R. Siegwart, "Smooth and efficient obstacle avoidance for a tour guide robot," in *Proceedings. 2003 ICRA, September 14-19, 2003, The Grand Hotel, Taipei, Taiwan*, vol. 1. IEEE Operations Center, 2003, pp. 446–451.
- [86] K. O. Arras, J. Persson, N. Tomatis, and R. Siegwart, "Real-time obstacle avoidance for polygonal robots with a reduced dynamic window," in *Proceedings 2002 ICRA (Cat. No. 02CH37292)*, vol. 3. IEEE, 2002, pp. 3050–3055.
- [87] B. Li, W. Zhou, J. Sun, C.-Y. Wen, and C.-K. Chen, "Development of model predictive controller for a tail-sitter vtol uav in hover flight," *Sensors*, vol. 18, no. 9, p. 2859, 2018.

Concurrent Application of Energy-Based Modeling Methodologies to Dynamics Analysis and Control: The Case of a Robotic Omnidirectional Base

Martín Crespo

LAC - FCEIA

Universidad Nacional de Rosario

Rosario, Argentina

crespom@fceia.unr.edu.ar

Sergio Junco

LAC - FCEIA

Universidad Nacional de Rosario

Rosario, Argentina

sjunco@fceia.unr.edu.ar

Matías Nacusse

LAC - FCEIA

Universidad Nacional de Rosario

Rosario, Argentina

nacusse@fceia.unr.edu.ar

Abstract—The purpose of this paper is to show the benefits of approaching the tasks of analysis, simulation and control system design through the consideration of system properties via the concurrent approach of different modeling methodologies. The particular case of an omnidirectional robotic base is chosen to present the approach in an illustrative way. The presentation of its differential kinematics is followed by the derivation of the dynamic models under different modeling approaches: the Bond Graphs, Euler-Lagrange, Newtonian and port-Hamiltonian paradigms. Using the Bond Graphs as a guiding thread, equivalences among the models are obtained which are further used to reinterpret system properties when considering them from the point of view of different formalisms, and also to bring to light some properties that may be hidden in some cases. The paper concludes by designing a velocity controller for the mobile platform starting with a physically heuristic approach on its Bond Graph model and, subsequently, demonstrating its stability in the port-Hamiltonian domain using the correspondences previously established concerning passivity under feedback interconnection.

Index Terms—Dynamic Models, Mobile Robots, Bond Graphs, Euler Lagrange, port-Hamiltonians, Omni Wheels.

I. INTRODUCTION

Mathematical Modeling plays an ever increasing role in Engineering, where the use of physical, behavioral and logical models in conjunction with software tools and methodologies brought-in approaches like Model-Based Systems Engineering (MBSE), the formalized application of modeling to support system requirements, design, analysis, verification and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases [1].

Model-Based Design (MBD), a component of MBSE, is a key methodology in Mechatronics, where a plethora of model types and techniques are mandatory due to the profusion of physical domains, engineering disciplines and technologies involved [2]. Among this multitude of types of models conspicuous in Mechatronics, for each specific task mostly just a single type of model is used, even if more than one are available. Particularly in the domain of control and automation, where dynamic physical models are of uppermost importance, some

modeling methodologies are preferred, according to well-established practices (and/or to the taste of the developers). For instance, the Euler-Lagrange (EL) formalism is the main technique resorted to in Robotics, despite the existence of the also well-established Newton-Euler formulation, or the more recently irrupted alternative techniques as, for instance, the port-Hamiltonian System (pHS) approach.

The EL and pHS models are natural candidates for describing numerous physical systems, as they capture the energy phenomena associated with the physical-based laws, such as (i) storage, (ii) flow, and (iii) dissipation. [3] [4]. We also consider the Bond Graphs (BG), a graphical representation that allows capturing energy phenomena as well as causal relationships between system variables, and is a highly suitable tool primarily conceived for modeling and simulation [5] and later also exploited for controller design [6] [7].

The purpose of this paper is to show that the benefits of a single-modeling approach to the tasks of analysis, simulation and control system design can be advantageously extended and deepened through the consideration of system properties via the concurrent approach of different methodologies. To this aim, the EL, Newton, pHS and BG modeling paradigms will be considered, as applied to the particular case of a mobile omnidirectional robotic platform. Such robots are called omnidirectional because they possess full mobility in the plane which means that they can move at each time instant in any direction without any reorientation [8].

In [9] the BG methodology was used for the development of the dynamic model and control for a mobile manipulator. In [10] a 3D dynamic model of a quadruped robot has been developed using the BG technique. In [11] a BG model of a planar mobile manipulator is presented in correspondence with its EL, BG with storage fields, and pHS models for analysis simulation and control. In [12] and [13] a systematic methodology to convert BG models to EL and pHS is presented respectively. The work [14] achieves a correspondence between port-based modeling of multi-domain physical systems using BG with the framework of geometric dynamical systems and control theory using pHS.

The remainder of this paper is organized as follows. In Section II the kinematics and dynamics of the omnidirectional mobile base is presented. Section III, Equivalences and Synergies, discusses correspondences among these models and exploits them in order to gain insight into system properties and to inspire and facilitate controller design. Section IV presents an application of the previous ideas via the development of a velocity controller for the mobile platform. Finally, the conclusions are discussed in Section V.

II. MODELING THE MOBILE BASE

A. Differential Kinematics

In Fig. 1a, the idealized physical system of an omnidirectional mobile base is depicted. It is composed of a main rigid body, henceforth referred to as the chassis, and three wheels (each of them, in turn, a rigid body), located 120 degrees apart from each other. The variables describing the motion of the center of mass (CM) of the base are the generalized independent coordinates (posture) $q = [x, y, \varphi]^T$ with respect to the fixed global system $O - X_F Y_F Z_F$ and the vector $\eta = [\dot{X}_G, \dot{Y}_G, \dot{\varphi}]^T$ that represents the non-inertial velocities of the CM of the chassis defined with respect to the non-inertial frame $O_G - X_G Y_G Z_G$. The vector of efforts $h_e = [F_{ex}, F_{ey}, \tau_e]^T$ represents the external forces exerted at the CM of the chassis.

From the idealized scheme of omni wheel i , with $i = 1, 2, 3$, shown in Fig. 1b, the non-inertial rotational velocities $\Omega_r = [\Omega_{r1}, \Omega_{r2}, \Omega_{r3}]^T$ and translational velocities $V_r = [V_{r1}, V_{r2}, V_{r3}]^T$ are identified, both defined with respect to the moving frame $O_{ri} - X_{ri} Y_{ri} Z_{ri}$, and the external torques (coming from motors) $\tau = [\tau_{r1}, \tau_{r2}, \tau_{r3}]^T$.

As the movement of the base depends on the wheels, the velocities $\dot{q}$ and V_r are not independent of each other, but rather are related to η through the relationships

$$\begin{cases} \dot{q} = R(q)^T \eta \\ V_r = r E \eta \end{cases} \Rightarrow V_r = r E R(q) \dot{q} \quad (1)$$

where r is the radius of the wheels, L is the distance from the CM of the chassis to the wheel axis, and the matrices¹

$$R(q) = \begin{bmatrix} c_\varphi & s_\varphi & 0 \\ -s_\varphi & c_\varphi & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad E = \begin{bmatrix} 1/2r & \sqrt{3}/2r & L/r \\ -1/r & 0 & L/r \\ 1/2r & -\sqrt{3}/2r & L/r \end{bmatrix}$$

taking into account that $R(q)R(q)^T = I_3$.

If we consider that the motion of the wheel occurs under conditions of perfect rolling and without slipping around the vertical axis passing through the contact point, the velocity vector V_r can be expressed as

$$V_r = r \Omega_r \quad (2)$$

¹The notations c_φ and s_φ are the abbreviations for $\cos \varphi$ and $\sin \varphi$, respectively.

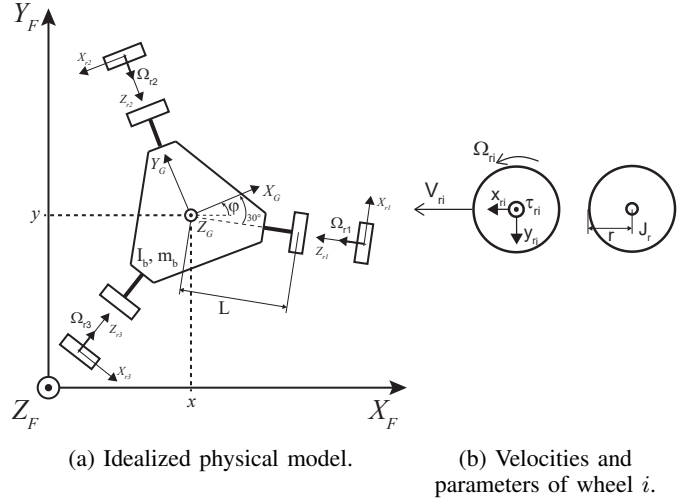


Fig. 1: Omnidirectional mobile base.

Thus, by substituting into (1), we obtain the kinematic relationship between the frame $O - X_F Y_F Z_F$ and the frames $O_{ri} - X_{ri} Y_{ri} Z_{ri}$

$$\Omega_r = G(q)^T \dot{q} \quad (3)$$

where the matrix $G(q)$ has been defined as $G(q) := R(q)^T E^T$.

B. Dynamics

In this section, we develop the dynamic model of the mobile base through four different methodologies: BG, force balance or Newtonian method, EL equations, and pHS.

1) *Bond Graph*: The velocity relationship of equation (3) is represented in the BG domain by interconnecting the 1-junctions associated with these velocities through the modulated transformer **MTF**. Effort sources **Se** are used to represent external forces and torques.

The causalized BG model is presented in Fig. 2, where the convention of positive power flow is taken from the wheels to the chassis. The three parts of the model are clearly distinguished: the omni wheels, the relationship between the inertial velocities, and the chassis. The **I** elements represent the rotational inertia of the wheels J_r , and the mass and inertia of the base, m_b and I_b respectively.

The vectors $F = [F_x, F_y, \tau_1]^T$ and $F_r = [F_{r1}, F_{r2}, F_{r3}]^T$ represent internal efforts and traction forces on the wheels, respectively, related in a power-conserving transformation such that

$$\begin{cases} \Omega_r = G(q)^T \dot{q} \\ F = G(q) r F_r \end{cases} \quad (4)$$

Remark 1. The order of the BG scheme is $n_{BG} = 6$, as it possesses three energy storage elements in integral causality (IC) and an integrator in a three dimensional vector. The state variables of the system are the momentum quantities associated with the energy storage elements in IC and the generalized independent posture coordinates q .

$$\begin{cases} M_b \ddot{q} = F + h_e \\ I_r \dot{\Omega}_r = \tau - r F_r \end{cases} \quad (5)$$

$$\begin{cases} M_b \ddot{q} = F + h_e \\ I_r \dot{\Omega}_r = \tau - r F_r \end{cases} \quad (5)$$

$$\dot{\Omega}_r = G(q)^T \ddot{q} + \dot{G}(q)^T \dot{q} \quad (6)$$
$$F = G(q) \left(\tau - I_r \dot{\Omega}_r \right)$$
$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} = G(q)\tau + h_e \quad (7)$$
$$M(q) = \begin{bmatrix} \frac{3J_r}{2r^2} + m_b & 0 & 0 \\ 0 & \frac{3J_r}{2r^2} + m_b & 0 \\ 0 & 0 & \frac{3J_r L^2}{2r^2} + I_b \end{bmatrix} \quad (8)$$

The diagram illustrates a power conserving conversion system for a vehicle, divided into three main functional blocks: Omniwheels, Power conserving conversion, and Chassis.

- Omniwheels:** This section represents the vehicle's wheels. It consists of three parallel paths, each corresponding to a wheel. Each path has an input force F_{r1} , F_{r2} , or F_{r3} and an input moment M_{r1} , M_{r2} , or M_{r3} . These inputs are connected to a junction with impedances Ω_{r1} , Ω_{r2} , and Ω_{r3} respectively.
- Power conserving conversion:** This central block performs the conversion between mechanical and electrical domains. It contains a block labeled **MTF** (Mechanical to Electrical Transducer). The output of the MTF block is a signal F that is fed into the Chassis section. A feedback loop is also present, where the output signal F is integrated (represented by the $\int$ block) and then passed through a block $G(q)$ before being fed back into the MTF block.
- Chassis:** This section represents the vehicle's chassis. It receives the converted force signal F from the MTF block. The chassis is modeled with three parallel paths for output forces F_x , F_y and output moments M_b , M_ϕ . These outputs are connected to a junction with impedances Ω_x , Ω_y , and Ω_ϕ respectively.

3) *Euler Lagrange*: The Lagrangian $\mathcal{L}$ of the mobile base is

$$\begin{aligned}\mathcal{L}(\psi, \dot{\psi}) &= \frac{1}{2} \dot{q}^T M_b \dot{q} + \frac{1}{2} \Omega_r^T I_r \Omega_r \\ &= \frac{1}{2} \dot{\psi}^T B \dot{\psi}\end{aligned}\tag{9}$$

Remark 3. From expression (9), it can be seen that the configuration coordinates ψ are cyclic (or ignorable) since they do not appear in the Lagrangian expression.

$$\frac{d}{dt} \left(\frac{\partial \mathcal{L}}{\partial \dot{\psi}} \right) - \frac{\partial \mathcal{L}}{\partial \psi} = S(\psi)u + A(\psi)\lambda, \quad u \in \mathbb{R}^6, \lambda \in \mathbb{R}^3 \quad (10)$$
$$S(\psi) = \begin{bmatrix} \mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{I}_3 \end{bmatrix} \quad u = \begin{bmatrix} h_e \\ \tau \end{bmatrix} \quad A(\psi) = \begin{bmatrix} G(q) \\ -\mathbf{I}_3 \end{bmatrix} \quad (11)$$
$$\begin{aligned} B\ddot{\psi} &= S(\psi)u + A(\psi)\lambda \\ A(\psi)^T\dot{\psi} &= \mathbf{0}_{3 \times 1} \end{aligned} \quad (12)$$
$$\dot{\psi} = J(\psi)\dot{q} \quad (13)$$
$$J(\psi)^T B \ddot{\psi} = J(\psi)^T u \quad (14)$$
$$M = J(\psi)^T B J(\psi) \quad C(q, \dot{q}) = J(\psi)^T B \dot{J}(\psi) \quad (15)$$
$$p := \frac{\partial \mathcal{L}}{\partial \dot{\psi}}$$
$$\mathcal{H}(\psi, p) := p^T \dot{\psi} - \mathcal{L}(\psi, \dot{\psi})$$

Thus, the Hamiltonian equations of a general system with constraints are

$$\begin{aligned}\dot{\psi} &= \frac{\partial H}{\partial p}(\psi, p) \\ \dot{p} &= -\frac{\partial H}{\partial \psi}(\psi, p) + S(\psi)u + A(\psi)\lambda \\ y &= S(\psi)^T \frac{\partial H}{\partial p}(\psi, p) \\ \mathbf{0}_{3 \times 1} &= A(\psi)^T \frac{\partial H}{\partial p}(\psi, p)\end{aligned}\quad (16)$$

where y is the *co-located output* (i.e., it has relative degree 1 with respect to the control input u). In this particular case, the momentum and the Hamiltonian are:

$$p = B\dot{\psi} \quad H(\psi, p) = \frac{1}{2}p^T B^{-1}p \quad (17)$$

and the values of the matrices $S(\psi)$ and $A(\psi)$ are as expressed in (11). Taking into account that the Lagrange multipliers can be eliminated by multiplying both sides of the second line of (16) by $J(\psi)^T$, we obtain

$$J(\psi)^T \dot{p} = -J(\psi)^T \frac{\partial H}{\partial \psi}(\psi, p) + J(\psi)^T S(\psi)u \quad (18)$$

Let's now define $\tilde{p} \in \mathbb{R}^3$ through the following change of coordinates (see [4])

$$\tilde{p} := J(\psi)^T p$$

where considering both, the equation (15) and that the momentum can be rewritten as $p = BJ(\psi)\dot{q}$, the Hamiltonian (17) can be expressed in terms of the new coordinates $\tilde{p}$ as

$$\tilde{H}(\psi, \tilde{p}) = \frac{1}{2}\tilde{p}^T M^{-1}\tilde{p} \quad (19)$$

Then, differentiating $\tilde{p}$ with respect to time, the equation (18) can be rewritten as

$$\dot{\tilde{p}} = -J(\psi)^T \frac{\partial \tilde{H}}{\partial \psi}(\psi, \tilde{p}) + \dot{J}(\psi)^T p + J(\psi)^T S(\psi)u$$

As it holds that

$$\frac{\partial H}{\partial p} = \frac{\partial \tilde{H}}{\partial \tilde{p}} = \dot{q} \quad \frac{\partial H}{\partial \psi} = \frac{\partial \tilde{H}}{\partial \psi} = \mathbf{0}_{6 \times 1} \quad (20)$$

the constrained system (16) takes the form

$$\begin{aligned}\begin{bmatrix} \dot{\psi} \\ \dot{\tilde{p}} \end{bmatrix} &= \begin{bmatrix} \mathbf{0}_6 & J(\psi) \\ -J(\psi)^T & J(\psi)^T B J(\psi) \end{bmatrix} \begin{bmatrix} \frac{\partial \tilde{H}}{\partial \psi} \\ \frac{\partial \tilde{H}}{\partial \tilde{p}} \end{bmatrix} + \begin{bmatrix} \mathbf{0}_6 \\ J(\psi)^T \end{bmatrix} u \\ y &= [\mathbf{0}_6 \quad J(\psi)] \begin{bmatrix} \frac{\partial \tilde{H}}{\partial \psi} \\ \frac{\partial \tilde{H}}{\partial \tilde{p}} \end{bmatrix}\end{aligned}$$

where in this case, the output are the velocities $y = \dot{\psi}$.

Now, since the Hamiltonian $\tilde{H}$ depends on the inertia matrix M which, in the general case, depends on the vector of posture

coordinates q , it is possible to express the Hamiltonian $\tilde{H}$ in terms of q and $\tilde{p}$. Thus, considering the equivalences of (20) and defining the state vector as $\mathbf{x} := [q^T, \tilde{p}^T]^T$, the simplified dynamic system is²:

$$\begin{aligned}\begin{bmatrix} \dot{q} \\ \dot{\tilde{p}} \end{bmatrix} &= \underbrace{\begin{bmatrix} \mathbf{0}_3 & \mathbf{I}_3 \\ -\mathbf{I}_3 & -C(q, \dot{q}) \end{bmatrix}}_{J(\mathbf{x})} \begin{bmatrix} \frac{\partial \tilde{H}}{\partial q} \\ \frac{\partial \tilde{H}}{\partial \tilde{p}} \end{bmatrix} + \underbrace{\begin{bmatrix} \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{I}_3 & G(q) \end{bmatrix}}_{g(\mathbf{x})} u \\ y &= \begin{bmatrix} \mathbf{0}_3 & \mathbf{I}_3 \\ \mathbf{0}_3 & G(q)^T \end{bmatrix} \begin{bmatrix} \frac{\partial \tilde{H}}{\partial q} \\ \frac{\partial \tilde{H}}{\partial \tilde{p}} \end{bmatrix}\end{aligned}\quad (21)$$

where the second equality of equation (15) together with the antisymmetry property $C(q, \dot{q})^T = -C(q, \dot{q})$ have been used. The skew-symmetric matrix $J(\mathbf{x}) = -J(\mathbf{x})^T$, of dimension (6×6) reveals the power conservation of the interconnection structure. Finally, the compact pHS model of the mobile base is:

$$\begin{aligned}\dot{\mathbf{x}} &= J(\mathbf{x}) \nabla \tilde{H}(\mathbf{x}) + g(\mathbf{x})u \\ y &= g(\mathbf{x})^T \nabla \tilde{H}(\mathbf{x})\end{aligned}$$

III. MODEL ANALYSIS

In this section, we will use modeling in the BG domain as a basis to establish connections and interpretations between the different developed models.

Recurring to the assignment of computational causality, BGs are mainly used for computational purposes, both numerical, i.e., digital simulation, and analytical, i.e., differential equations derivation, mainly state equations [5] but also EL [12] and pHS [13]. The formal procedures to do this start from either BGs in full IC and without algebraic loops, or from BGs including derivative causality (DC), like the one in Fig. 2, and algebraic loops.

In both latter cases differential-algebraic equations (DAE-systems) are implied with index equal or greater than one [17]. To avoid dealing with them, different approaches are available. When possible, the conversion of the DAE-system to a pure ODE (an index zero system) through formal manipulation is preferred. When not, the addition to the BG of parasitic dynamic components or residual sinks [18] are other options on hand allowing to break the causal couplings producing the DAE-system.

In the sequel some of these techniques are applied to the BG of Fig. 2 and some implications related to the models in the different formalisms are discussed.

A. DAE to ODE conversion and BG-EL-Force-balance correspondences

Eliminating the presence of energy storage elements in DC in a BG via regrouping the interdependent storage elements is equivalent to converting a DAE-system to an index zero system.

²Notice that $\dot{q}$ can be expressed in terms of $\tilde{p}$, so the interconnection matrix depends on the state $\mathbf{x}$.

In the case of the BG shown in Fig. 2, this regrouping involves moving the rotational inertias of the wheels that are in DC to the right side, so that all energy storage elements are in IC, as shown in the BG schematic in Fig. 3.

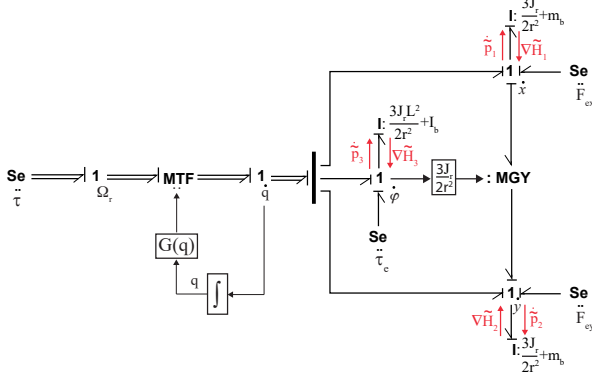


Fig. 3: Mobile base BG scheme with inertia regrouping.

The correspondence of this BG with the results of force-balance and EL modeling is clearly suggested considering equation (7) and comparing its main components given in (8), the inertia matrix M with the parameters of the I-elements of the BG, and the entries of the matrix $C(q, \dot{q})$ with the coefficient modulating the gyrator MGY.

The BG-EL equivalence can be more formally established deriving the EL equations from the BG of Fig. 2 following the procedure given in [12]. This is not presented here due to lack of space.

B. Residual Sinks and Lagrange multipliers

Another way to avoid DC is to enforce IC in all energy storage elements by using *residual sinks*. Starting from the BG in Fig. 2, the procedure consists on incorporating effort sources through **0**-junctions into the storage elements in DC, as observed in Fig. 4.

The *residual sinks* inject the necessary effort into the **0**-junctions to ensure that the net flow through them is zero. Thus, defining $f_{vir} := [f_{vir1}, f_{vir2}, f_{vir3}]^T$, the flow balance equations in the **0**-junctions are

$$\begin{aligned} \mathbf{0}_{3 \times 1} &= f_{vir} \\ &= G(q)^T \dot{q} - \Omega_r \\ &= A(\psi)^T \dot{\psi} \end{aligned}$$

Thus, we can see that the constraint matrix $A(\psi)$, of equation (12), can be obtained by enforcing IC for all energy storage elements in DC through the incorporation of *residual sinks*.

Remark 4. The efforts introduced by the *residual sinks* are equivalent to the Lagrange multipliers vector λ defined in (10), so they can be interpreted physically as the force that the ground must exert to fulfill the perfect rolling condition defined in (2). This force is hidden in the final model obtained through the BG modeling and both, the EL and pHS modeling.

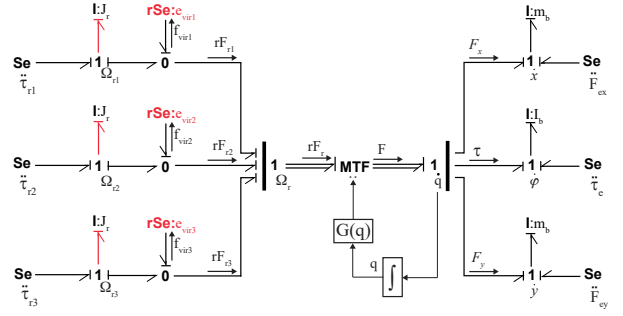


Fig. 4: Mobile base BG scheme with *residual sinks*.

C. Bond Graph - Hamiltonian System

The pHS model can be obtained through the manipulation of the BG using the method developed in [13]. Alternatively, in this simple case, the pHS is obtained applying the state-equations derivation procedure writing the derivatives of the state variables, i.e. the inputs of the storage elements, as functions of their outputs, as these are the components of the gradient of the stored energy (please refer to the red signals marked in Fig. 3).

D. Passivity

The power delivered to the system (21) is

$$\begin{aligned} u^T y &= u^T g(x)^T \nabla \tilde{H}(x) \\ &= \left(\dot{x} - J(x) \nabla \tilde{H}(x) \right)^T \nabla \tilde{H}(x) \\ &= \dot{x}^T \nabla \tilde{H}(x) \\ &= \dot{\tilde{H}}(x) \end{aligned}$$

where it was taken into account that

$$\frac{\partial \tilde{H}^T}{\partial \tilde{p}} C(q, \dot{q}) \frac{\partial \tilde{H}}{\partial \tilde{p}} = 0$$

Since the following inequality is satisfied

$$\underbrace{\tilde{H}(x(t)) - \tilde{H}(x(0))}_{\text{stored energy}} \leq \underbrace{\int_0^t u(s)^T y(s) ds}_{\text{delivered energy}}$$

the system is cyclo-passive [4]. Moreover, since $\tilde{H}(x) \geq 0$, the system is also passive.

Remark 5. In both the EL and pHS modeling, we started with the generalized configuration coordinates ψ of dimension 6, and after some algebraic work, the model ended up being expressed in terms of the posture coordinates vector q of dimension 3. This conclusion can be quickly reached by analyzing the causalized BG model in Fig. 2. Even if all the configurations coordinates ψ appear in the model, the number of energy storage elements in IC is 3.

IV. CONTROLLER DESIGN

Elaborating on the compact version of the BG scheme shown in Fig. 3 in this section we propose a PID-velocity

control scheme that can be an objective per-se or an inner-loop of a position controller. The PID can be represented in the energy domain as a combination of dissipative and (kinetic and potential) storage elements, as shown in Fig. 5. The external passivity between its input (the velocity error) and its output (the torque vector commanding the wheels) can be easily verified. The velocity input references Ω_r^* can be expressed as function of the desired mobile base velocities $\dot{q}^*$ using the relation of equation (3), i.e., $\Omega_r^* = G(q)^T \dot{q}^*$. Then, the resulting control law is

$$\tau = -K_P \tilde{\Omega}_r - K_I \tilde{\theta}_r - K_D \dot{\tilde{\Omega}}_r$$

where $\tilde{\Omega}_r = \Omega_r - \Omega_r^*$ is the velocity error, $\tilde{\theta}_r = \int \tilde{\Omega}_r$ and K_P , K_I and K_D are positive definite diagonal matrices, each (3×3) dimensional.

Exploiting the BG-pHS correspondence, we study the stability of the closed-loop equilibrium point ($\tilde{\theta}_{r*} = 0, \tilde{\Omega}_{r*} = 0, \tilde{p}_* = 0$) proposing the sum of the PID-associated energy function and the Hamiltonian (19) as the candidate Lyapunov function

$$V(\tilde{\theta}_r, \tilde{\Omega}_r, \tilde{p}) = \frac{1}{2} \tilde{\theta}_r^T K_I \tilde{\theta}_r + \frac{1}{2} \tilde{\Omega}_r^T K_D \tilde{\Omega}_r + \frac{1}{2} \tilde{p}^T M^{-1} \tilde{p}$$

From (20), considering that $\tilde{p} = M\dot{q}$ and considering $h_e = 0_{3 \times 1}$, the time derivative of V is

$$\begin{aligned} \dot{V} &= \tilde{\theta}_r^T K_I \dot{\tilde{\theta}}_r + \tilde{\Omega}_r^T K_D \dot{\tilde{\Omega}}_r + \tilde{p}^T M^{-1} \dot{\tilde{p}} \\ &= -\tilde{\Omega}_r^T \tau - \tilde{\Omega}_r^T K_P \tilde{\Omega}_r + \tilde{p}^T M^{-1} \left(G(q)\tau - C(q, \dot{q}) \frac{\partial \tilde{H}}{\partial \tilde{p}} \right) \\ &= -\tilde{\Omega}_r^T K_P \tilde{\Omega}_r \end{aligned}$$

As $\dot{V}$ is negative semi-definite, it cannot be guaranteed that the equilibrium point is asymptotically stable.

The latter can be easily proved invoking La Salle's invariance principle and analyzing the closed-loop dynamics when the input $\Omega_r^* = 0$. As the system continues evolving, the equilibrium point is asymptotically stable.

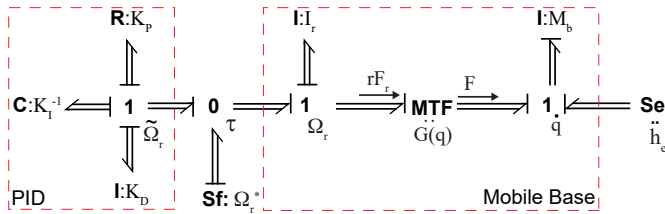


Fig. 5: PID controller BG scheme.

V. CONCLUSIONS

In this work, the kinematic and dynamic models of an omnidirectional base with three omni wheels were presented. The dynamic model was developed using four different techniques: BG, Force Balance, EL, and pHS. The first one was used as a *lingua franca* for the generalization of results. The manipulation of the BG model to avoid derivative causalities and its linking with the EL model allowed to explicitly express

and give a mathematical meaning to the Lagrange multipliers, demonstrating that these were equivalent to the forces introduced by the *residual sinks*. With both methodologies, these forces disappeared in the final model. At the same time, it was demonstrated that both can be interpreted physically as the force that the ground must exert to fulfill the perfect rolling condition. Finally, a PID-velocity control design was presented.

ACKNOWLEDGMENT

The authors wish to thank the National University of Rosario for funding this work through the projects PPCT-80020220600083UR and PID-80020190300098UR.

REFERENCES

- [1] T. INCOSE, "Systems engineering vision 2020," *INCOSE, San Diego, CA*, accessed Jan, vol. 26, no. 2019, p. 2, 2007.
- [2] G. Barbieri, K. Kernschmidt, C. Fantuzzi, and B. Vogel-Heuser, "A sysml based design pattern for the high-level development of mechatronic systems to enhance re-usability," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 3431–3437, 2014.
- [3] R. Ortega, J. A. L. Perez, P. J. Nicklasson, and H. J. Sira-Ramirez, "Introduction," in *Passivity-based control of Euler-Lagrange systems: mechanical, electrical and electromechanical applications*. Springer Science & Business Media, 2013, ch. 1, pp. 1–15.
- [4] A. Van der Schaft, *L2-gain and passivity techniques in nonlinear control*. Springer, 2000, vol. 2.
- [5] D. C. Karnopp, D. L. Margolis, and R. C. Rosenberg, *System dynamics: modeling, simulation, and control of mechatronic systems*. John Wiley & Sons, 2012.
- [6] A. Donaire and S. Junco, "Energy shaping, interconnection and damping assignment, and integral control in the bond graph domain," *Simulation Modelling Practice and Theory*, vol. 17, no. 1, pp. 152–174, 2009.
- [7] A. Achir, S. Junco, A. Donaire, and C. Sueur, "A bond-graph method for flatness-based dynamic feedback linearization controller synthesis: Application to a current-fed induction motor," in *ECMS '06*, 2006.
- [8] C. C. de Wit, B. Siciliano, and G. Bastin, *Theory of robot control*. Springer Science & Business Media, 2012.
- [9] R. Ram, P. Pathak, and S. Junco, "Trajectory control of a mobile manipulator in the presence of base disturbance," *SIMULATION*, vol. 95, no. 6, pp. 529–543, 2019.
- [10] M. M. Gor, P. M. Pathak, A. K. Samantaray, J. M. Yang, and S. W. Kwak, *Bond graph Modeling and Control of Compliant Legged Quadruped Robot*. Cham: Springer International Publishing, 2017, pp. 497–546.
- [11] M. Nacusse, M. Crespo, S. Junco, V. Rayankula, and P. M. Pathak, "Bond graph model conditioning for analysis, simulation and control system design: Application to a planar mobile robotic manipulator," in *IMAACA*, 2017.
- [12] D. Karnopp, "Lagrange's Equations for Complex Bond Graph Systems," *Journal of Dynamic Systems, Measurement, and Control*, vol. 99, no. 4, pp. 300–306, 12 1977.
- [13] A. Donaire and S. Junco, "Derivation of input-state-output port-hamiltonian systems from bond graphs," *Simulation Modelling Practice and Theory*, vol. 17, no. 1, pp. 137–151, 2009.
- [14] V. Duindam, A. Macchelli, S. Stramigioli, and H. Bruyninckx, *Modeling and Control of Complex Physical Systems: The Port-Hamiltonian Approach*. Springer Berlin Heidelberg, 2009.
- [15] B. Siciliano, L. Sciacivco, L. Villani, and G. Oriolo, "Motion control," in *Robotics: modelling, planning and control*. Springer Science & Business Media, 2009, ch. 11, pp. 369–521.
- [16] R. Fierro and F. L. Lewis, "Control of a nonholonomic mobile robot: Backstepping kinematics into dynamics," *Journal of robotic systems*, vol. 14, no. 3, pp. 149–163, 1997.
- [17] G. Wanner and E. Hairer, *Solving ordinary differential equations II*. Springer Berlin Heidelberg New York, 1996, vol. 375.
- [18] W. Borutsky and F. E. Cellier, "Tearing algebraic loops in bond graphs," *Transactions of the Society for Computer Simulation International*, vol. 13, no. 2, p. 102–115, Dec 1996.

Arquitectura de una computadora de navegación, guiado y control tolerante a fallas

Federico Nuñez Frau¹, Leonardo Garberoglio², Claudio Pose^{1,3}, Juan Giribet³

¹Laboratorio de Automática y Robótica, Facultad de Ingeniería, Universidad de Buenos Aires

²Grupo de Estudio de Sistemas de Control, Facultad Regional San Nicolás, Universidad Tecnológica Nacional y CONICET

³Laboratorio de Inteligencia Artificial y Robótica - Universidad de San Andrés y CONICET, Buenos Aires, Argentina.

Resumen—Este trabajo detalla el diseño y construcción de una computadora de Navegación, Guiado y Control para pequeños vehículos autónomos y/o teleoperados. Con el objetivo de lograr la tolerancia a fallas en la computadora de abordo de un pequeño UAV, se toman las consideraciones necesarias para la conexión y gestión de la información entre tres unidades idénticas denominadas nodos. Se plantea una arquitectura distribuida, donde cada nodo recibe los datos adquiridos por las demás y ejecuta un algoritmo para corroborar la consistencia de los mismos. Para validar el funcionamiento, se realiza una prueba experimental donde se detecta la falla de una unidad inercial.

Index Terms—Controladora de vuelo, Tolerancia a fallas, Vehículos aéreos no tripulados

Sesión 5: Diseño, arquitecturas, modelado y control II

Control en cascada de un vehículo aéreo utilizando compensación dinámica robusta y adaptable

Claudio Rosales
Instituto de Automática (INAUT)
CONICET-UNSJ
San Juan, Argentina
crosales@inaut.unsj.edu.ar

Daniel Khede Dourado Villa
Departamento de Engenharia Elétrica
Universidade Federal do Espírito Santo
Vitória, Brasil
daniel.villa@ufes.br

Mário Sarcinelli-Filho
Departamento de Engenharia Elétrica
Universidade Federal do Espírito Santo
Vitória, Brasil
mario.sarcinelli@ufes.br

Kimón Valavanis
Department of Electrical & Computer Engineering
University of Denver
Denver, Estados Unidos
Kimón.Valavanis@du.edu

Ricardo Carelli
Instituto de Automática (INAUT)
CONICET-UNSJ
San Juan, Argentina
rcarelli@inaut.unsj.edu.ar

Abstract—Este artículo propone un novedoso controlador de modo deslizante adaptativo para rechazar perturbaciones relacionadas con el control de alto nivel de un vehículo aéreo no tripulado (UAV). El controlador propuesto está diseñado como una arquitectura de control de bucle interior-externo, también conocida como controlador en cascada. Un controlador cinemático implementa el bucle exterior, cuya función es guiar al UAV en el seguimiento de una trayectoria determinada (también válido para alcanzar un punto estático deseado). En cuanto al bucle interior, está diseñado como un controlador de modo deslizante adaptativo de súper torsión, basado totalmente en las velocidades del cuerpo. La novedad de esta propuesta está justamente en el diseño de este circuito de control interno, en el que se considera el enfoque de súper torsión y a la superficie de deslizamiento se le agrega un término integral destinado a reducir el efecto de vibración asociado con el deslizamiento de primer orden más común superficie. También se demuestra la estabilidad de todo el sistema de control y se muestran resultados de simulaciones numéricas que permiten afirmar que el sistema de control propuesto es eficaz para guiar al UAV a seguir una trayectoria determinada.

Index Terms—Súper-Twisting, Control Adaptable, UAV

I. INTRODUCCIÓN

La investigación sobre vehículos aéreos no tripulados (UAVs), comúnmente conocidos como drones, ha recibido una atención creciente en los últimos años debido a su gran cantidad de aplicaciones, tales como vigilancia, operaciones de búsqueda y rescate, y monitoreo agrícola y ambiental, entre varias otras. Sin embargo, controlar UAVs es particularmente desafiante debido a su dinámica no lineal, sensores internos ruidosos y la influencia de perturbaciones como el desgaste mecánico de las partes del vehículo o ráfagas repentinas de viento.

El control por modo deslizante (SMC, por sus siglas en inglés) es una herramienta simple y eficiente para contrarrestar las perturbaciones que actúan sobre los UAVs. En su

diseño convencional, el controlador de modo deslizante de primer orden (FOSMC, por sus siglas en inglés) teóricamente garantiza una compensación exacta de perturbaciones si la ganancia de su acción de control discontinua es mayor que los límites superiores de las perturbaciones. Sin embargo, esta propiedad se logra debido al control discontinuo, lo que produce oscilaciones de alta frecuencia, conocidas como el efecto de "chattering". Una variación del FOSMC clásico es el llamado controlador de modo deslizante de super-twisting (STSMC), diseñado para evitar el problema de chattering. La idea principal que explora es ocultar la acción de control discontinua detrás de un integrador, generando así una acción de control continua, asegurando teóricamente una compensación exacta de perturbaciones Lipschitz.

Para discutir estos temas, el documento se divide en varias secciones, comenzando con la Sección II, que presenta el diseño del controlador, incluyendo el controlador de cinemática inversa, el compensador dinámico y las pruebas de estabilidad correspondientes, lo que permite concluir que el error de seguimiento de trayectoria converge a cero. A continuación, la Sección III describe el la plataforma de simulación y los experimentos realizados para validar la estrategia de control propuesta y discuten los resultados obtenidos. Finalmente, la Sección IV destaca las conclusiones finales del trabajo.

II. SISTEMA DE CONTROL

El sistema de control propuesto en este trabajo consta de dos lazos cerrados, formando lo que se conoce como un sistema de control en cascada o un sistema de control de bucle interno-externo. El lazo externo, el primero, es un controlador cinemático, seguido por un compensador dinámico en cascada. El controlador cinemático genera las referencias de velocidad necesarias para que el vehículo aéreo alcance los objetivos de seguimiento de trayectoria. Sin embargo, tradicionalmente los controladores cinemáticos asumen un

El autor Claudio Rosales agradece a la Comisión Fulbright Argentina por el soporte en la estadía de investigación en la University of Denver, EEUU

seguimiento de velocidad perfecto, descuidando las dinámicas del vehículo durante el desarrollo, lo que puede resultar en errores de posición significativos. Por lo tanto, se propone un compensador dinámico como el lazo interno para minimizar los errores de velocidad. El diagrama de bloques del sistema de lazo cerrado completo se muestra en la Figura 1. El enfoque de esta propuesta es la adopción de un controlador de modo deslizante adaptativo de super-twisting para la compensación dinámica. Esto incluye la integración de la superficie de deslizamiento del error para reducir el chattering, un problema común en los controladores de modo deslizante.

El controlador cinemático tiene como objetivo el seguimiento de una referencia variable en el tiempo. En otras palabras, dada una trayectoria de referencia $\mathbf{x}_d(t) \in \mathbb{R}^4$ que sea suave, diferenciable y continua a lo largo de la trayectoria deseada, el objetivo es alcanzarla y garantizar que los errores de control tiendan a cero. Las referencias definen la posición en el espacio y el ángulo de guiñada ($\mathbf{x}_d^w(t) = [x_d^w(t) \ y_d^w(t) \ z_d^w(t) \ \psi_d^w(t)]^T$). El modelo cinemático establece la relación entre las velocidades del UAV en el marco de referencia del cuerpo ($\dot{\mathbf{x}}^b$) y el marco de referencia fijo ($\dot{\mathbf{x}}^w$). En una condición cercana a la suspensión (consulte [2] para más detalles), lo que corresponde a ángulos de cabeceo y alabeo cercanos a cero [3], dicho modelo se da simplemente por

$$\dot{\mathbf{x}}^w = \underbrace{\begin{bmatrix} \cos \psi^b & -\sin \psi^b & 0 & 0 \\ -\sin \psi^b & \cos \psi^b & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\mathbf{R}(\psi^b)} \underbrace{\begin{bmatrix} \dot{x}^b \\ \dot{y}^b \\ \dot{z}^b \\ \dot{\psi}^b \end{bmatrix}}_{\dot{\mathbf{x}}^b}, \quad (1)$$

donde ψ^b es el ángulo de guiñada actual del UAV. Para lograr el objetivo de control, se propone la ley de control cinemático

$$\dot{\mathbf{x}}_k^b = \mathbf{R}(\psi^b)^{-1} (\dot{\mathbf{x}}_d^w + \mathbf{K}_k \tilde{\mathbf{x}}^w), \quad (2)$$

basada en cinemática inversa [1], para la tarea de seguimiento de la trayectoria de referencia. En esta ley de control, $\mathbf{K}_k$, un parámetro de ganancia de diseño, es una matriz diagonal definida positiva, mientras que $\tilde{\mathbf{x}}^w = \mathbf{x}_d^w(t) - \mathbf{x}^w(t)$ y $\dot{\mathbf{x}}_d^w$ es la velocidad de la trayectoria que se está siguiendo, obtenida derivando la trayectoria deseada $\mathbf{x}_d^w(t)$.

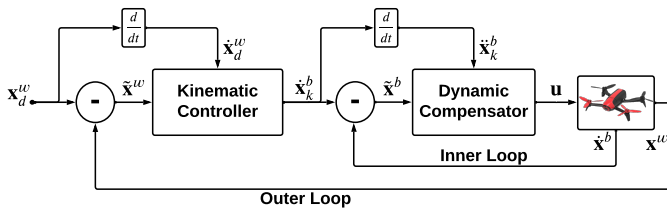


Fig. 1. Diagrama de bloques del sistema de control propuesto. El controlador cinemático está conectado en cascada con el compensador dinámico ya que genera las referencias de velocidad adoptadas por dicho compensador. Adaptado de [1]

A. Análisis de estabilidad

En la siguiente sub-sección, se diseña un compensador dinámico novedoso, cuyo objetivo es mejorar el seguimiento de las velocidades de referencia calculadas por el controlador cinemático de (2). Dado que un seguimiento perfecto no es posible considerando únicamente dicho controlador cinemático, persiste un error de seguimiento de velocidad, que es

$$\dot{\tilde{\mathbf{x}}}^b = \dot{\mathbf{x}}_k^b - \dot{\mathbf{x}}^b, \quad (3)$$

donde $\dot{\mathbf{x}}^b$ es el vector de velocidades actual del UAV y $\dot{\mathbf{x}}_k^b$ es el vector de velocidades de referencia proporcionadas por el controlador cinemático. La ecuación de lazo cerrado se obtiene reemplazando (2) y (3) en (1), lo que resulta en

$$\dot{\tilde{\mathbf{x}}}^w = \dot{\mathbf{x}}_d^w + \mathbf{K}_k \tilde{\mathbf{x}}^w - \mathbf{R}(\psi^b) \dot{\tilde{\mathbf{x}}}^b, \quad (4a)$$

o

$$\dot{\tilde{\mathbf{x}}}^w - \dot{\mathbf{x}}^w + \mathbf{K}_k \tilde{\mathbf{x}}^w - \mathbf{R}(\psi^b) \dot{\tilde{\mathbf{x}}}^b, \quad (4b)$$

o

$$\dot{\tilde{\mathbf{x}}}^w + \mathbf{K}_k \tilde{\mathbf{x}}^w = \rho, \quad (5)$$

considerando $\rho = \mathbf{R}(\psi^b) \dot{\tilde{\mathbf{x}}}^b$. Se utiliza la teoría de Lyapunov para analizar la estabilidad del sistema de lazo cerrado. Se propone la función candidata de Lyapunov $P = \frac{1}{2} \tilde{\mathbf{x}}^{wT} \tilde{\mathbf{x}}^w > 0$, cuya derivada temporal es

$$\dot{P} = \tilde{\mathbf{x}}^{wT} \dot{\tilde{\mathbf{x}}}^w. \quad (6)$$

Al introducir (5) en (6), se obtiene

$$\dot{P} = \tilde{\mathbf{x}}^{wT} \rho - \tilde{\mathbf{x}}^{wT} \mathbf{K}_k \tilde{\mathbf{x}}^w, \quad (7)$$

de donde se puede verificar que una condición suficiente para que $\dot{P}$ sea una función definida negativa es $|\tilde{\mathbf{x}}^{wT} \mathbf{K}_k \tilde{\mathbf{x}}^w| > |\tilde{\mathbf{x}}^{wT} \rho|$. Dado que $\mathbf{K}_k$ es simétrica, se puede aplicar la propiedad de la forma cuadrática a la ecuación anterior, lo que resulta en que

$$|\tilde{\mathbf{x}}^w| > \frac{|\rho|}{\lambda_{\min}(\mathbf{K}_k)}, \quad (8)$$

es la condición necesaria para garantizar que $\dot{P}$ sea negativa, donde $\lambda_{\min}(\mathbf{K}_k) > 0$ es el menor valor propio de la matriz $\mathbf{K}_k$. Tal ecuación representa la condición para garantizar la estabilidad práctica del sistema. A partir de ella se puede percibir que si $|\tilde{\mathbf{x}}^w|$ aumenta, provoca que el valor de $\dot{P}$ se vuelva más negativo, lo que hace que el error de control disminuya. Como resultado, el error de control $\tilde{\mathbf{x}}^w$ está acotado, de modo que

$$|\tilde{\mathbf{x}}^w| \leq \frac{|\rho|}{\lambda_{\min}(\mathbf{K}_k)}. \quad (9)$$

Esta condición implica que los errores de control están acotados en última instancia, y por lo tanto, el sistema no es asintóticamente estable. Además, los errores de control están acotados a valores que son función de los errores de velocidad. Por lo tanto, si $\rho \rightarrow 0$, dichos límites convergerán asintóticamente a cero, mostrando así que $\tilde{\mathbf{x}}^w(t) \rightarrow 0$, lo que permite concluir que la compensación dinámica, mediante la

reducción del error de seguimiento de velocidad ρ , contribuye efectivamente a lograr el objetivo de control de reducir el error de posición $\tilde{\mathbf{x}}^w$.

B. Compensación Dinámica

A continuación presentamos un compensador dinámico novedoso diseñado para mejorar el rendimiento de seguimiento de velocidades de referencia calculadas por el controlador cinemático según se define en la ecuación (2). Es crucial reconocer que la suposición de un seguimiento de velocidad perfecto es ficticia en escenarios prácticos. Los sistemas físicos del mundo real poseen inherentemente inercia, lo que hace que la consecución instantánea de una referencia de velocidad sea inalcanzable. Además, el análisis de estabilidad realizado en la subsección anterior revela que los errores de control están directamente influenciados por discrepancias de velocidad. En consecuencia, minimizar los errores de velocidad se vuelve imperativo para garantizar un seguimiento de trayectoria más preciso. Dentro de esta subsección, proponemos un compensador dinámico diseñado para lograr la convergencia de los errores de seguimiento de velocidad hacia cero.

La dinámica del vehículo aéreo, siempre considerando la condición de vuelo estacionario cercano, se expresa mediante el modelo

$$\ddot{\mathbf{x}}^b = \kappa_u \mathbf{u} - \kappa_v \dot{\mathbf{x}}^b - \delta(t), \quad (10)$$

inicialmente propuesto en [4] y también discutido en [3], [5], que representa la relación entre los comandos normalizados enviados a un dron comercial y sus aceleraciones. El vector $u_i \in [-1, +1]$ $i = 1, \dots, 4$ representa los comandos normalizados enviados al vehículo, que son la velocidad lineal a lo largo y la velocidad angular alrededor del eje z^b , y dos ángulos de referencia (cabecceo y alabeo). Además, el segundo término contempla todos los efectos aerodinámicos que afectan al vehículo, que están relacionados con su velocidad. Finalmente, las perturbaciones externas y los errores de modelado están representados por el término $\delta(t)$. Por hipótesis, se considera que las derivadas de esta perturbación están limitadas, $\|\dot{\delta}(t)\| \leq \delta_o$. En cuanto a las matrices κ_u y κ_v , son las matrices diagonales

$$\kappa_u = \begin{bmatrix} \kappa_{u1} & 0 & 0 & 0 \\ 0 & \kappa_{u3} & 0 & 0 \\ 0 & 0 & \kappa_{u5} & 0 \\ 0 & 0 & 0 & \kappa_{u7} \end{bmatrix} \quad (11a)$$

y

$$\kappa_v = \begin{bmatrix} \kappa_{v2} & 0 & 0 & 0 \\ 0 & \kappa_{v4} & 0 & 0 \\ 0 & 0 & \kappa_{v6} & 0 \\ 0 & 0 & 0 & \kappa_{v8} \end{bmatrix}, \quad (11b)$$

cuyas entradas fueron identificadas, obteniendo así las estimaciones $\hat{\kappa}_{u_i}$, $i = 1, 3, 5, 7$, y $\hat{\kappa}_{v_j}$, $j = 2, 4, 6, 8$.

Los comandos proporcionados por el controlador cinemático son las referencias de velocidad $\dot{\mathbf{x}}_k^b$, mientras que las velocidades reales desarrolladas por el UAV se expresan como $\dot{\mathbf{x}}_k^b$. Para trabajar en el error $\tilde{\mathbf{x}}^b = \dot{\mathbf{x}}_k^b - \dot{\mathbf{x}}^b$ al mismo

tiempo que se rechaza la perturbación $\delta(t)$, se propone la superficie de deslizamiento

$$\mathbf{s} = \dot{\mathbf{x}}^b + \lambda \int \dot{\mathbf{x}}^b dt \quad (12)$$

donde λ es una matriz diagonal de ganancia definida positiva. Derivando tal expresión, se obtiene que

$$\dot{\mathbf{s}} = \ddot{\mathbf{x}}^b + \lambda \dot{\mathbf{x}}^b = (\ddot{\mathbf{x}}_k^b - \ddot{\mathbf{x}}^b) + \lambda(\dot{\mathbf{x}}_k^b - \dot{\mathbf{x}}^b). \quad (13)$$

La ley de control equivalente en modo deslizante se puede encontrar resolviendo $\dot{\mathbf{s}} = 0$. Por lo tanto, introduciendo (10) en la ecuación anterior resulta en

$$\dot{\mathbf{s}} = \ddot{\mathbf{x}}_k^b - (\kappa_u \mathbf{u} - \kappa_v \dot{\mathbf{x}}^b - \delta(t)) + \lambda \dot{\mathbf{x}}^b = 0. \quad (14)$$

La ley de control tipo super-twisting

$$\begin{aligned} \mathbf{u} &= \hat{\kappa}_u^{-1}(\ddot{\mathbf{x}}_k^b + \hat{\kappa}_v \dot{\mathbf{x}}^b + \lambda \dot{\mathbf{x}}^b) + \mathbf{w} = \phi \hat{\boldsymbol{\theta}} + \mathbf{w} \\ \mathbf{w} &= \mathbf{K}_1 \sqrt{\|\mathbf{s}\|} \text{sign}(\mathbf{s}) + \int \mathbf{K}_2 \text{sign}(\mathbf{s}) dt \end{aligned} \quad (15)$$

Se propone entonces, para lograr el objetivo de control previamente presentado. En dicha ecuación, $\mathbf{K}_1$ y $\mathbf{K}_2$ representan matrices de ganancia diagonal positiva definida, mientras que el término $\phi \hat{\boldsymbol{\theta}}$ parametriza una parte de la ley de control, con

$$\phi = [(\ddot{\mathbf{x}}_d^w + \lambda \dot{\mathbf{x}}^b)^T \quad \dot{\mathbf{x}}^b]^T \quad (16a)$$

y,

$$\hat{\boldsymbol{\theta}}^T = \left[\frac{1}{\hat{\kappa}_{u1}} \quad \frac{1}{\hat{\kappa}_{u3}} \quad \frac{1}{\hat{\kappa}_{u5}} \quad \frac{1}{\hat{\kappa}_{u7}} \quad \frac{\hat{\kappa}_{v2}}{\hat{\kappa}_{u1}} \quad \frac{\hat{\kappa}_{v4}}{\hat{\kappa}_{u3}} \quad \frac{\hat{\kappa}_{v6}}{\hat{\kappa}_{u5}} \quad \frac{\hat{\kappa}_{v8}}{\hat{\kappa}_{u7}} \right]. \quad (16b)$$

donde

$$\hat{\kappa}_u^{-1} = \begin{bmatrix} \frac{1}{\hat{\kappa}_{u1}} & 0 & 0 & 0 \\ 0 & \frac{1}{\hat{\kappa}_{u3}} & 0 & 0 \\ 0 & 0 & \frac{1}{\hat{\kappa}_{u5}} & 0 \\ 0 & 0 & 0 & \frac{1}{\hat{\kappa}_{u7}} \end{bmatrix} \quad (17)$$

y

$$\hat{\kappa}_u^{-1} \hat{\kappa}_v = \begin{bmatrix} \frac{\hat{\kappa}_{v2}}{\hat{\kappa}_{u1}} & 0 & 0 & 0 \\ 0 & \frac{\hat{\kappa}_{v4}}{\hat{\kappa}_{u3}} & 0 & 0 \\ 0 & 0 & \frac{\hat{\kappa}_{v6}}{\hat{\kappa}_{u5}} & 0 \\ 0 & 0 & 0 & \frac{\hat{\kappa}_{v8}}{\hat{\kappa}_{u7}} \end{bmatrix}. \quad (18)$$

Consecuentemente, sustituyendo (15) en (14) se obtiene

$$\begin{aligned} \dot{\mathbf{s}} &= -\kappa_u \left(\phi \hat{\boldsymbol{\theta}} + \mathbf{K}_1 \sqrt{\|\mathbf{s}\|} \text{sign}(\mathbf{s}) + \int \mathbf{K}_2 \text{sign}(\mathbf{s}) dt \right) \\ &\quad + \kappa_v \dot{\mathbf{x}}^b + \delta(t) + \ddot{\mathbf{x}}_k^b + \lambda \dot{\mathbf{x}}^b. \end{aligned} \quad (19)$$

Ahora, consideremos $\mathbf{a} = \ddot{\mathbf{x}}_k^b + \lambda \dot{\mathbf{x}}^b$ para mejorar el desarrollo matemático. Sustituyendo esto y manipulando la ecuación anterior se obtiene

$$\begin{aligned} \dot{\mathbf{s}} &= -\kappa_u \left(\phi \hat{\boldsymbol{\theta}} + \mathbf{K}_1 \sqrt{\|\mathbf{s}\|} \text{sign}(\mathbf{s}) + \int \mathbf{K}_2 \text{sign}(\mathbf{s}) dt \right) \\ &\quad + \kappa_v \dot{\mathbf{x}}^b + \delta(t) + \ddot{\mathbf{x}}_k^b + \lambda \dot{\mathbf{x}}^b. \end{aligned} \quad (20)$$

Ahora, considerando $\tilde{\theta} = \theta - \hat{\theta}$, se obtiene que

$$\dot{s} = \kappa_u \phi \tilde{\theta} - \kappa_u \left(\mathbf{K}_1 \sqrt{\|s\|} \text{sign}(s) + \int \mathbf{K}_2 \text{sign}(s) dt \right) + \delta(t) \quad (21)$$

Para llevar a cabo el análisis de estabilidad, se utiliza nuevamente la teoría de Lyapunov, ahora considerando la función candidata de Lyapunov

$$V = \frac{1}{2} s^T \kappa_u^{-1} s + \frac{1}{2} \tilde{\theta}^T \gamma^{-1} \tilde{\theta},$$

donde γ es una matriz diagonal positiva definida con entradas pequeñas. La derivada temporal de dicha función es

$$\dot{V} = s^T \kappa_u^{-1} \dot{s} + \tilde{\theta}^T \gamma^{-1} \dot{\tilde{\theta}}. \quad (22)$$

Sustituyendo (21) en la ecuación anterior se obtiene

$$\dot{V} = s^T \kappa_u^{-1} \{ \kappa_u \phi \tilde{\theta} - \kappa_u [\mathbf{K}_1 \sqrt{\|s\|} \text{sign}(s) + \int \mathbf{K}_2 \text{sign}(s) dt] + \delta(t) \} + \tilde{\theta}^T \gamma^{-1} \dot{\tilde{\theta}}, \quad (23a)$$

o

$$\dot{V} = s^T \phi \tilde{\theta} - s^T [\mathbf{K}_1 \sqrt{\|s\|} \text{sign}(s) + \int \mathbf{K}_2 \text{sign}(s) dt - \kappa_u^{-1} \delta(t)] + \tilde{\theta}^T \gamma^{-1} \dot{\tilde{\theta}}. \quad (23b)$$

De la propiedad de transposición de matriz,

$$s^T \phi \tilde{\theta} = (s^T \phi \tilde{\theta})^T = \tilde{\theta}^T \phi^T s. \quad (24)$$

Al presentar esta propiedad se obtiene

$$\dot{V} = \tilde{\theta}^T \phi^T s + \tilde{\theta}^T \gamma^{-1} \dot{\tilde{\theta}} - s^T [\mathbf{K}_1 \sqrt{\|s\|} \text{sign}(s) + \int \mathbf{K}_2 \text{sign}(s) dt - \kappa_u^{-1} \delta(t)], \quad (25a)$$

o

$$\dot{V} = \tilde{\theta}^T (\phi^T s + \gamma^{-1} \dot{\tilde{\theta}}) - s^T [\mathbf{K}_1 \sqrt{\|s\|} \text{sign}(s) + \int \mathbf{K}_2 \text{sign}(s) dt - \kappa_u^{-1} \delta(t)]. \quad (25b)$$

Respecto a la ley de actualización de parámetros del sistema, se adopta

$$\dot{\tilde{\theta}} = \gamma \phi^T s. \quad (26)$$

Sin embargo, como $\tilde{\theta} = \theta - \hat{\theta}$, se obtiene que $\dot{\tilde{\theta}} = \dot{\theta} - \dot{\hat{\theta}}$, con $\dot{\hat{\theta}} = 0$, ya que θ corresponde a los valores exactos de los parámetros, por lo que son valores constantes. Esto resulta en

$$\dot{\tilde{\theta}} = -\dot{\hat{\theta}} = -\gamma \phi^T s, \quad (27)$$

cuya introducción en (25b) da como resultado

$$\dot{V} = -s^T \mathbf{K}_1 \sqrt{\|s\|} \text{sign}(s) - s^T \left\{ \int [\mathbf{K}_2 \text{sign}(s) - \kappa_u^{-1} \dot{\delta}(t)] dt \right\}. \quad (28)$$

En función de las normas

$$\dot{V} \leq -\|\mathbf{K}_1\| \|s\| \sqrt{\|s\|} - \|s\| \left\| \int \mathbf{K}_2 \text{sign}(s) dt \right\| + \|s\| \left\| \int \kappa_u^{-1} \dot{\delta}(t) dt \right\|, \quad (29)$$

o

$$\dot{V} \leq -\|\mathbf{K}_1\| \|s\| \sqrt{\|s\|} - \|s\| \int \|\mathbf{K}_2\| dt + \|s\| \int \|\kappa_u^{-1} \dot{\delta}(t)\| dt \quad (30)$$

La estabilidad está asegurada tomando $\|\mathbf{K}_2\| \geq \|\kappa_u^{-1} \dot{\delta}(t)\|$. Por hipótesis, se asumió que $\dot{\delta}(t)$ está acotado. A su vez, κ_u^{-1} también está acotado. De esta manera es perfectamente posible seleccionar una matriz $\mathbf{K}_2$ adecuada. Como consecuencia de tal elección de $\mathbf{K}_2$ se obtiene

$$\dot{V} \leq -\|\mathbf{K}_1\| \|s\| \sqrt{\|s\|} \leq 0, \quad (31)$$

y, por lo tanto, el controlador en (15) es asintóticamente estable, lo que significa que el error de seguimiento de velocidad tiende a cero cuando $t \rightarrow \infty$. Ahora, volviendo al controlador cinemático, el límite del error de seguimiento en (9) va a cero porque $\rho \rightarrow 0$, y por lo tanto el error de control resulta ser asintóticamente convergente a cero. Por lo tanto, el controlador completo se muestra en la Figura 2, donde también se muestran las variables involucradas.

III. SIMULACIONES

A continuación se presentan simulaciones numéricas que validan la propuesta de control presentada en este artículo. Las mismas se han desarrollado en MatLab y el modelo dinámico utilizado para describir el comportamiento del vehículo aéreo fue tomado de [6], en donde se considera la parte física del modelo y de los actuadores. Se describen los parámetros dinámicos y aerodinámicos. Este modelo fue implementado en un código de C++. Este modelo se comunica con MatLab a través de una aplicación bidireccional de memoria compartida donde se envían comandos y se recibe la telemetría del vehículo. El modelo recibe como comandos velocidades de referencia para cada uno de los grados de libertad y si bien se hicieron la identificación de los parámetros del modelo 10 el algoritmo de control se iniciaron con valores arbitrarios $\kappa_u = \text{diag}(0, 5, 0.5, 0.5, 0.5)$ y $\kappa_v = \text{diag}(0, 5, 0.5, 0.5, 0.5)$ de forma de poder evaluar el comportamiento del controlador frente a errores paramétricos considerables. El objetivo fue el seguimiento de una trayectoria circular con una variación de altura sinusoidal, $\mathbf{x}_d^w(t) = [r \sin(\frac{2\pi}{5}), r \cos(\frac{2\pi}{5}), 1 + 0.3 \sin(\frac{2\pi}{5}), \arctan(\frac{\dot{x}}{\dot{y}})]^T [m]$. La figura 3 muestra la gráfica de los errores de control de todo el sistema de control de lazo cerrado. Claramente se comprueba como a medida que la simulación evoluciona los errores disminuyen. Esto es consecuencia del algoritmo adaptable que ajusta el valor de los parámetros del controlador para mejorar su desempeño como se observa en la figura 4 desde los valores iniciales a valores que permiten reducir los errores de seguimiento de velocidad, y de manera solidaria disminuir los errores de control. Además, en la figura 5 muestra los errores de seguimiento de velocidad entre los comandos cinemáticos y la velocidad real del vehículo. De igual manera se observa como los valores mejoran conforme la simulación avanza y se

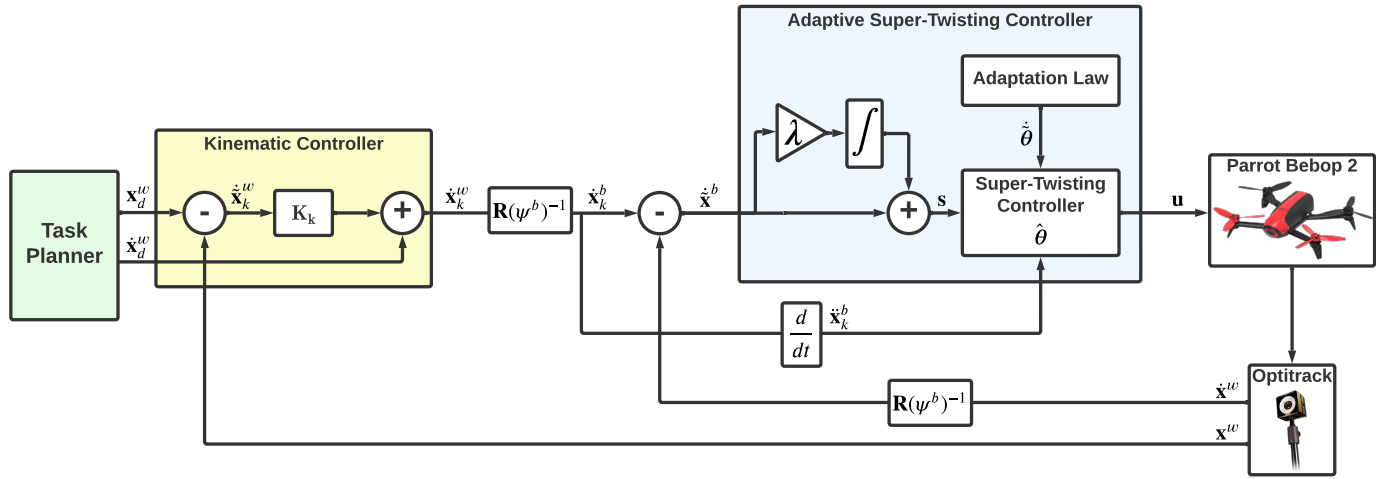


Fig. 2. Diagrama de bloques detallado del controlador en cascada y el controlador adaptativo Super-Twisting.

valida el compensador dinámico presentado en este trabajo. Finalmente, la figura 6 muestra la evolución de los cuatro grados de libertad y la trayectoria de referencia. En base a lo expuesto en las gráficas se valida que el algoritmo de control en cascadas es exitoso en su objetivo de controlar un vehículo aéreo en tareas de seguimiento de trayectoria.

IV. CONCLUSIONES

En este trabajo se presentó un esquema novedoso de control de trayectoria de un vehículo aéreo del tipo multirotor. La estructura utilizada es un controlador en cascada. La primera etapa consta de un controlador cinemático que genera comandos de velocidades de referencia, y posteriormente, un compensador dinámico garantiza el seguimiento de velocidad de estos comandos. La demostración de estabilidad desarrollada

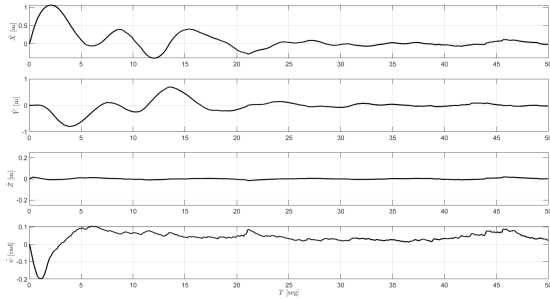


Fig. 3. Errores cinemáticos de cada uno de los cuatro grados de libertad.

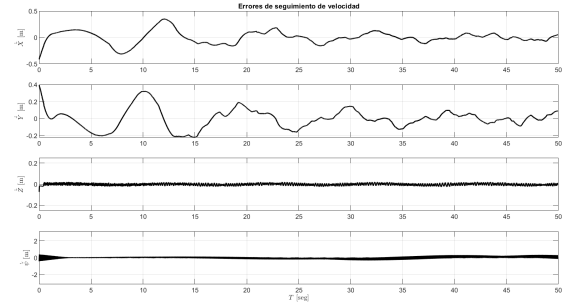


Fig. 5. Error de seguimiento de velocidad entre la referencia del controlador cinemático y la velocidad real del vehículo.

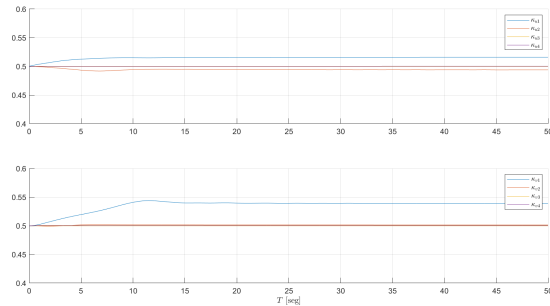


Fig. 4. Evolución de los parámetros del controlador. Se observa como los mismos convergen.

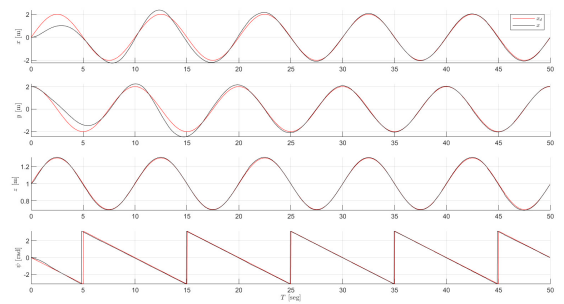


Fig. 6. Valores de la referencia y la posición real del vehículo.

valida que los errores de control son asintóticamente estable con el compensador dinámico desarrollado. Finalmente, simulaciones numéricas validaron la propuesta desarrollada

REFERENCES

- [1] M. Sarcinelli-Filho and R. Carelli, "Motion control," in *Control of Ground and Aerial Robots, chapter 4*. Cham, Switzerland: Springer International Publishing, 2023, pp. 41–98.
- [2] —, "Kinematic models," in *Control of Ground and Aerial Robots, chapter 2*. Cham, Switzerland: Springer International Publishing, 2023, pp. 5–22.
- [3] L. V. Santana, A. S. Brandão, and M. Sarcinelli-Filho, "Navigation and cooperative control using the AR.Drone quadrotor," *Journal of Intelligent & Robotic Systems*, pp. 1–24, 2016.
- [4] L. V. Santana, A. S. Brandao, M. Sarcinelli-Filho, and R. Carelli, "A trajectory tracking and 3D positioning controller for the AR.Drone quadrotor," in *International Conference on Unmanned Aircraft Systems (ICUAS)*, Orlando, FL, USA, June 2014, pp. 756–767.
- [5] M. Sarcinelli-Filho and R. Carelli, "Dynamic models," in *Control of Ground and Aerial Robots, chapter 3*. Cham, Switzerland: Springer International Publishing, 2023, pp. 23–39.
- [6] C. Powers, D. Mellinger, and V. Kumar, "Quadrotor kinematics and dynamics," *Handbook of Unmanned Aerial Vehicles*, pp. 307–328, 2015.

Catamarán Guazú: Embarcación Autónoma para Monitoreo de Espejos de Agua y Aplicaciones de Robótica Acuática

Santiago Pierini¹, Claus Rosito¹, Gabriel Torre^{1,2}, Ivan Jourdan¹, Pedro Segura³, Ignacio Mas^{1,4}, Juan Giribet^{1,4}

¹Laboratorio de Inteligencia Artificial y Robótica - Universidad de San Andrés, Buenos Aires, Argentina.

²Instituto de Ingeniería Biomédica, Facultad de Ingeniería, Universidad de Buenos Aires

³Nordelta S.A., Buenos Aires, Argentina

⁴CONICET - Argentina

Resumen—En este trabajo se presenta el diseño y construcción de una embarcación con el objetivo de servir como plataforma para el desarrollo e implementación de algoritmos de navegación autónoma. Se realiza un diseño buscando obtener una autonomía extendida utilizando un sistema eléctrico con capacidad de carga solar. Finalmente se valida el diseño naval, mecánico y eléctrico realizando ensayos de flotación y de conducción comandando la embarcación por radio control.

I. INTRODUCCIÓN

Los vehículos de superficie no tripulados (USV) han ganado relevancia en el ámbito náutico debido a su eficiencia y capacidad para operar en condiciones desafiantes sin tripulación a bordo. Su importancia se ve impulsada por los avances en tecnologías de baterías y motores eléctricos de la última década y la creciente demanda en sectores como la defensa, investigación y conservación de flora y fauna. Estos vehículos ofrecen una alternativa segura y económica para misiones que podrían ser muy riesgosas o directamente no factibles para vehículos tripulados.

Gran parte de los desarrollos recientes en USVs se centran en misiones de corta duración y de autonomía limitada [1]. BlueRobotics [2] provee un USV para fines de investigación y desarrollo, pero de dimensión reducida y con poca área para paneles solares, limitando su autonomía. Liquid Robotics Wavewalker [3] provee un USV autopropulsado que utiliza energía de las olas para generar electricidad, ofreciendo una autonomía prácticamente ilimitada y la capacidad de operar en condiciones de alto oleaje, pero requiere una gran profundidad mínima para operar. Del mismo modo, el Saildrone Surveyor [4] es un USV propulsado por viento y energía solar, ideal para estudios a largo plazo en áreas remotas pero tampoco es apto para operar en áreas con aguas poco profundas y vientos moderados a altos.

En nuestro país, se han desarrollado este tipo de vehículos, con diseños que demuestran capacidades de navegación y evasión autónoma de obstáculos [5], [6] o coordinación multirobot [7], y sigue siendo un tema actual de investigación y desarrollo. El presente desarrollo se enmarca en las actividades del Laboratorio de Inteligencia Artificial y Robótica de la Universidad de San Andrés en conjunto con Nordelta S.A. En este

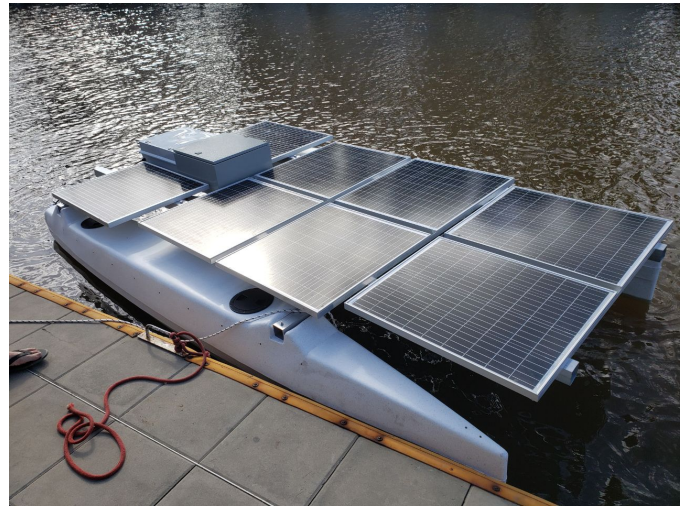


Figura 1. Vista panorámica de la embarcación

proyecto se puso especial énfasis en crear una plataforma que sirva como base para futuros ensayos de sensores, algoritmos y conceptos de misión como por ejemplo navegación autónoma, batimetría o control coordinado [8] y que a la vez posea una gran autonomía, posibilitando misiones de larga duración.

Para cumplir con este objetivo se seleccionó la configuración catamarán, ya que para naves de eslora y manga similares es la configuración que minimiza la masa mientras maximiza la eficiencia hidrodinámica y la estabilidad, permitiendo optimizar la autonomía y proveer una plataforma estable frente al oleaje [9]. Adicionalmente, la configuración catamarán permite minimizar el perfil, reduciendo la resistencia al viento en comparación con configuraciones monocasco, ayudando a optimizar la estabilidad y la autonomía. Bajo estas consideraciones se avanzó en un diseño que cuenta con dos pontones en paralelo unidos por una estructura central. Se tienen dos pares de motores, cada uno en la parte posterior de cada pontón, permitiendo realizar un control diferencial de la embarcación. La superficie superior del vehículo está cubierta por paneles solares, que en conjunto con la batería

y su sistema de regulación y carga asociados con el fin de proveer una elevada capacidad de autonomía para la nave.

II. DISEÑO

El sistema eléctrico presenta decisiones de compromiso al momento de realizar un diseño de un vehículo autónomo con paneles solares [10]. En primer lugar, cuanto mayor sea el área disponible de celdas solares, mayor será la potencia disponible de estos pero también mayor será el tamaño y el peso del vehículo. En segundo lugar, cuanto mayor sea la capacidad de la batería, mayor será la autonomía pero también será mayor el peso. Teniendo en cuenta estos compromisos se inicia la etapa de diseño con dos componentes fijados, los pontones y los motores eléctricos por cuestiones de disponibilidad en el mercado y de presupuesto.

Se utiliza el pontón Fun Cats de SIT ON TOP [11] de fabricación local.

Datos por cada pontón:

- Fabricante: SIT ON TOP
- Modelo: Fun Cats
- Eslora (largo): 3,4 M
- Manga (ancho): 0,35 M
- Alto: 0,50 m
- Peso: 25 kg
- Capacidad de carga máxima: 100 kg

Los motores eléctricos corresponden al modelo T500 de Blue Robotics [2]. Estos motores se adquieren con su controlador electrónico de velocidad (ESC) propietario. Se colocan dos motores por pontón, obteniendo un total de cuatro motores.

Datos de cada motor:

- Fabricante: Blue Robotics
- Modelo: T500
- Potencia máxima: 1044 W
- Empuje máximo: 16,1 kgf
- Corriente máxima: 43,5 A
- Tensión de alimentación (ESC): 7-26 V

Se estima el consumo de los motores ya que estos representan la parte mas significativa de la carga sobre el sistema eléctrico. Se considera que el funcionamiento promedio de los motores va a ser cuando estos generen la mitad del empuje máximo posible. Según los datos provistos por el fabricante [2], el valor de corriente para este valor de empuje es de 13,88 A por motor, por lo que el consumo promedio sera de 55,52 A para cuatro motores. Además se tiene en consideración que el valor máximo de corriente que puede solicitar cada motor es de 43,5 A a máxima potencia por hasta un minuto, por lo que la corriente pico podrá ser de 174 A. La tensión de alimentación del ESC (controlador del motor) debe estar en el rango de 7 V a 26 V [2]. Cuanto mayor sea esta tensión, mayor potencia podrá transmitirse a los motores. Por estas condiciones, se decide utilizar una batería con una tensión nominal de 24 V.

Debido a la dificultad de acceso rápido al vehículo durante el uso, ya que este estará navegando, se opta por una batería con celdas LiFePO4 debido su perfil de mayor seguridad

comparado con otras baterías basadas en Litio [12]. De esta manera se busca reducir el riesgo de incendio por problemas con la batería. Se encarga la construcción de una batería con celdas LiFePO4 de 24V y 200Ah a un fabricante local [13], esta estará compuesta principalmente por 16 celdas que se conectaran como dos paralelos de 8 celdas en serie cada uno.

Datos de la batería:

- Tecnología: LiFePO4
- Fabricante: Electroconversiones
- Capacidad nominal: 200 Ah
- Corriente de descarga nominal: 1 C
- Corriente de carga: 0,5 C
- Tensión nominal por celda: 3.2 V
- Peso por celda: 1,8 kg

Teniendo en consideración que la descarga máxima recomendada (DOD por sus siglas en inglés) de la batería es del 80 % de su capacidad y dejando 5 % de margen por una posible pérdida de capacidad por degradación [12], se tienen disponibles para consumo 150 Ah, con un consumo constante al valor estimado se obtiene una duración de 2 horas y 42 minutos de uso si solo se tiene en cuenta la capacidad de la batería como fuente de energía.

Para el diseño del sistema de paneles solares, se utiliza un regulador de carga del tipo seguidor de punto de máxima potencia (MMPT por sus siglas en inglés). Esta tecnología de reguladores adaptan su impedancia de carga para que el conjunto de paneles trabaje en el punto de máxima transferencia de potencia. Este componente regula la tensión que recibe del conjunto de paneles solares al valor necesario para la carga de la batería. Es importante destacar que el voltaje de carga solo puede ser un valor menor que la tensión que generan los paneles solares.

En la figura 2 se muestra de manera ilustrativa una curva de tensión contra corriente de una celda solar, aquí se puede observar el punto de máxima transferencia de potencia.

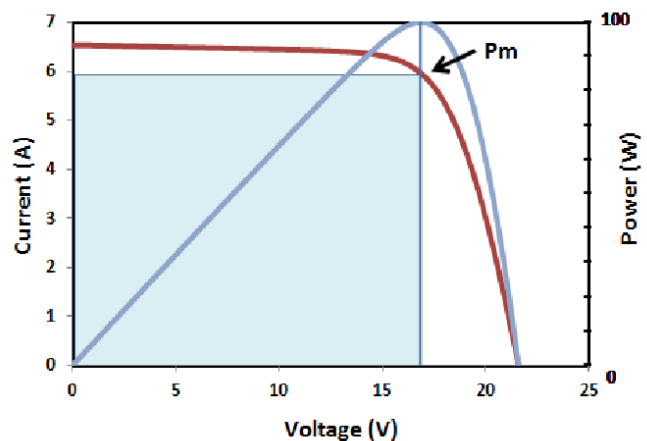


Figura 2. En esta figura se muestra en rojo la curva de corriente contra tensión y en celeste la curva de potencia contra tensión de una celda solar, donde PM es el punto de máxima potencia [14].

Esta curva se modifica según las condiciones de trabajo en las que se encuentre la celda solar. Las variables que alteran significativamente el funcionamiento de estas celdas son la temperatura de trabajo y el valor de irradiancia solar (potencia incidente por unidad de superficie). Las figuras 3 y 4 muestran los efectos de la temperatura y la irradiancia sobre la curva de la figura 2.

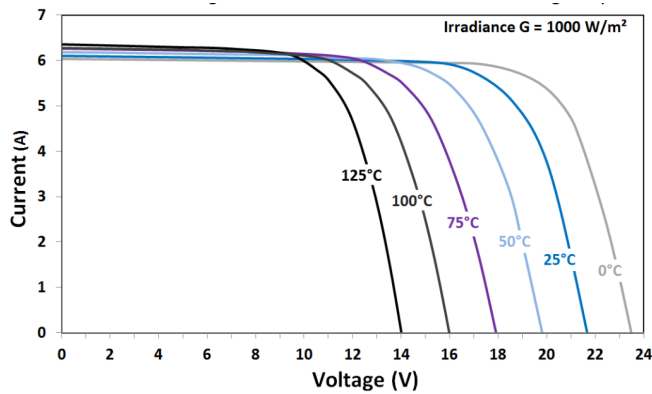


Figura 3. Efecto del cambio de temperatura en las condiciones de trabajo de una celda solar [14].

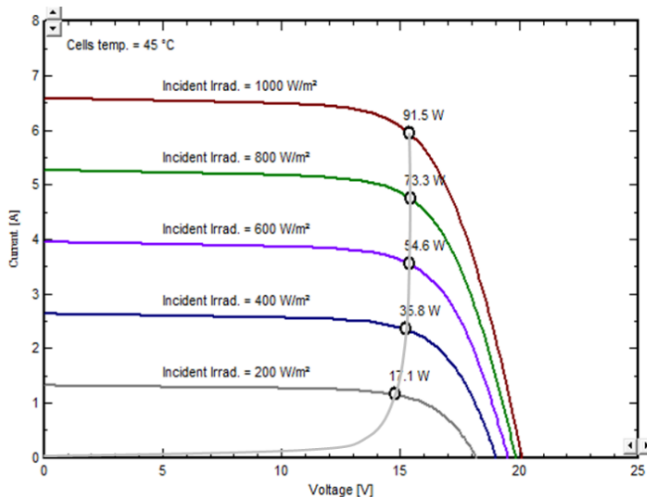


Figura 4. Efecto del cambio de irradiancia en las condiciones de trabajo de una celda solar [14].

Para determinar la tensión máxima del conjunto de paneles se tiene en cuenta dos factores. En primer lugar los efectos del cambio de temperatura e irradiancia que generan una reducción en el valor de tensión del punto de óptima transferencia de potencia. En segundo lugar que el regulador de carga solo pueda reducir la tensión de entrada. Por estos motivos se decide fijar un valor de tensión máxima para todo el conjunto de paneles solares, que permita tener un amplio margen respecto del valor de carga de la batería y de esta forma garantizar carga aún en condiciones no ideales. Con estos lineamientos se buscan opciones de paneles solares en el mercado local, evaluando no solo la tensión máxima del conjunto sino que también su tamaño, su peso y la potencia generada.

Se utiliza el software de dibujo en 3D Onshape [15] para representar los componentes y realizar pruebas de ensamble dentro de este entorno. Para la construcción de la estructura principal, donde irán montados los componentes, se decide utilizar caños cuadrados de aluminio de 60 mm y 1,5 mm de espesor. Se elige aluminio debido a su menor densidad, comparado con otros materiales estándares como acero, por lo que agregará menor peso al vehículo. Las dimensiones de los caños son medidas estándares que pueden soportar el peso máximo especificado de los pontones. Estos perfiles se sujetan a los pontones y entre si utilizando tornillería de acero inoxidable. En la figura 5 se muestra una vista del diseño de la estructura con los pontones.

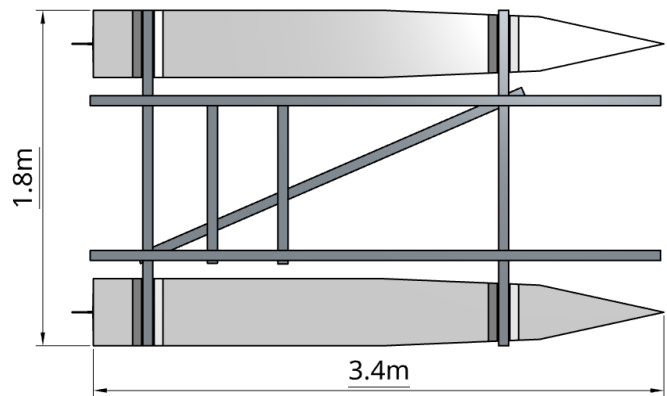


Figura 5. Vista superior del diseño 3D donde se puede observar la estructura mecánica.

Finalmente, teniendo en cuenta el área disponible del diseño, las dimensiones de la batería y las dimensiones estimadas de los gabinetes que alojan los componentes eléctricos, se colocan ocho unidades de los paneles SN-M-KIT200W de SolarLine [16], este modelo trae dos paneles de 100W por unidad.

Datos del panel solar:

- Fabricante: Solarline
- Modelo: SN-M-KIT200W
- Potencia máxima: 100 W
- corriente de cortocircuito: 6,12 A
- Tensión a circuito abierto: 21,6 V
- Peso: 7,5 kg
- Tamaño: 78x69x3 cm

Para poder definir las características del regulador de carga necesario se evalúan las condiciones de trabajo de este. Cada panel solar tiene una tensión de circuito abierto de 21,6 V y los ocho paneles se conectaran como dos conjuntos en paralelo de cuatro paneles en serie, con esta configuración se tiene una tensión máxima de circuito abierto de 86,4 V. De esta manera se obtiene una diferencia de tensión que es 3,6 veces mayor que la tensión nominal de la batería.

Por otro lado se evalúa la corriente máxima de carga teniendo en cuenta que la potencia máxima del conjunto de paneles es de 800 W. Con el fin de obtener la peor condición para la corriente de salida se considera que el regulador de

carga tiene una eficiencia del 100 % en la conversión y que la tensión de carga será 20 V (tensión aproximada del fin de descarga para 8 celdas de LiFePO4 en serie), de modo que el valor de corriente en su peor condición será de 40 A.

Por estas condiciones se necesita un regulador de carga que pueda tolerar al menos 86,4 V del conjunto de paneles solares y que pueda entregar más de 40 A de corriente de carga.

Se utiliza el regulador MPPT 100/50 de Victron Energy [17].

Datos del regulador MPPT:

- Corriente de carga nominal: 50 A
- Tensión máxima del conjunto de paneles: 100 V
- Corriente máxima del conjunto de paneles: 60 A
- Eficiencia: 98 %

Por ultimo se colocan dos componentes más en el circuito eléctrico, en primer lugar se agregan diodos de bloqueo a cada conjunto serie de paneles solares, de esta manera se busca evitar que cuando alguno de los conjuntos en paralelo opere en condiciones desfavorables respecto del otro, este le genere una carga al otro conjunto. En segundo lugar se coloca un fusible de 250A a la salida de la batería a modo de protección en caso de un cortocircuito. En la figura 6 se puede ver el diseño del circuito eléctrico.

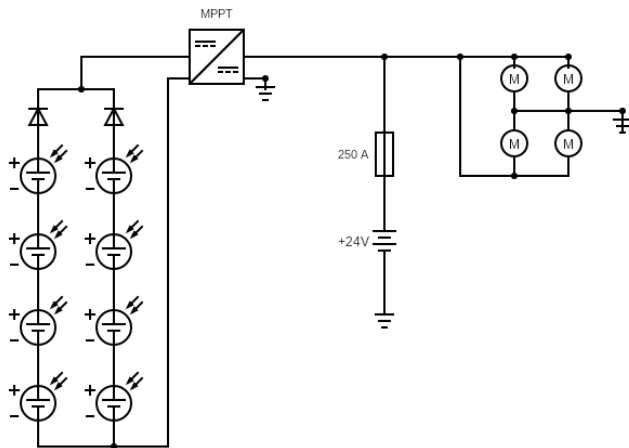


Figura 6. Esquemático de referencia del sistema eléctrico de la embarcación.

Se decide montar los motores sobre la superficie trasera de los pontones, el soporte de estos es construido a medida utilizando herramientas manuales. Los motores se montan a una placa de aluminio de 5 mm de espesor utilizando tornillos M4.

Con el sistema eléctrico y la posición de los motores definida, se puede realizar un ajuste fino del diseño mecánico, teniendo en cuenta los pesos de cada componente y evaluando la posición del centro de masa de diferentes configuraciones.

Cabe destacar que en el posicionamiento de los paneles se tiene en cuenta que en el caso de una colisión sean los pontones los que absorban el golpe. La construcción final del vehículo se puede ver en la figura 1.

III. RESULTADOS EXPERIMENTALES

Se presentan dos pruebas de la embarcación que se detallan a continuación

III-1. Primera prueba de flotación: La primera prueba es la verificación de la flotación de la embarcación solo con la estructura principal y los motores, es decir sin el peso que agregan los paneles solares y la batería. En esta prueba también se coloca un lastre de 70 kg para evaluar el impacto que tiene el peso que agregan los componentes restantes. Para tener control sobre la embarcación durante la prueba se coloca un gabinete con dos baterías de LiPo de 22,8 V y 10 Ah, cuyo peso es de 1,2 kg, además dentro de este gabinete se colocan los controladores de los motores y un receptor de radio Spektrum AR6210 [18] para utilizar un radio control Spektrum Dx7s [18]. Se verifica que el dispositivo flota con y sin el lastre, que no ingresa agua dentro de los pontones y que el montaje de los motores es lo suficientemente robusto para soportar la fuerza que estos ejercen. En la parte superior de la figura 7 se puede observar una imagen de esta prueba.



Figura 7. Prueba de flotación de la embarcación, en la figura superior se muestra sin el sistema eléctrico montado y en la figura inferior con el sistema eléctrico montado.

III-2. Segunda prueba de flotación: La segunda prueba de flotación se realiza con el sistema eléctrico montado (paneles solares, batería y gabinetes). Estos elementos agregan un peso de 120 kg a la embarcación. Las baterías pequeñas ya no son necesarias ya que la batería especificada para la embarcación está montada. Se verifica que el dispositivo flota con un peso aproximado al total, que no ingresa agua dentro de los pontones y que los motores tienen la potencia suficiente para maniobrar la embarcación con el peso agregado. En la parte inferior de la figura 7 se puede observar una imagen de esta prueba.

IV. CONCLUSIONES Y TRABAJO FUTURO

Se presentaron el diseño, construcción y ensayos de una embarcación no tripulada con capacidad de carga solar, permitiendo su operación prolongada sin necesidad de carga manual de las baterías. Combinado con las características hidrodinámicas de la configuración catamarán, esto la convierte en una plataforma ideal para misiones autónomas de larga duración, como por ejemplo el monitoreo de espejos de agua, batimetría de alta revisita y otros estudios de flora, fauna y conservación medioambiental, a la vez que también se presenta como una base para investigación y desarrollo en robótica acuática, algoritmos de navegación y control, evasión autónoma de obstáculos y coordinación multirobot, solo por nombrar algunos.

A partir de aquí se abren varias líneas de trabajo, incluyendo:

- Medición del sistema de carga solar.
- Ensayo de la autonomía de la embarcación en condiciones controladas.
- Estimación de la autonomía considerando diferentes escenarios de misión y casos de uso.
- Integración de sensores que aumenten la capacidad de navegación autónoma, tales como GPS, Corredera (Velocidad relativa al agua) y Lidar, así como instrumentos que permitan realizar mediciones meteorológicas.
- Integración con Sistema de Control de Misión para Misiones Robóticas de Largo Plazo.

Finalmente, se pasará a la etapa de desarrollo de los algoritmos de navegación y control para dotar a este robot acuático de las funcionalidades específicas de un robot autónomo.

AGRADECIMIENTOS

Se agradece a la Universidad de San Andrés por su apoyo y principalmente al Centro Educativo Área Beta de Nordelta. También se agradece al Dr. Hernán Socolovsky por su asesoramiento en lo referente a Energía Solar, y al personal de la carrera de Diseño de la Universidad de San Andrés.

REFERENCIAS

- [1] V. Bolbot, A. Sandru, T. Saarniniemi, O. Puolakka, P. Kujala, and O. A. Valdez Banda, "Small unmanned surface vessels—a review and critical analysis of relations to safety and safety assurance of larger autonomous ships," *Journal of Marine Science and Engineering*, vol. 11, no. 12, 2023. [Online]. Available: <https://www.mdpi.com/2077-1312/11/12/2387>
- [2] Blue Robotics, "Underwater rovs, usvs, thrusters, and sonars!" 2024, accessed: date-of-access. [Online]. Available: <https://bluerobotics.com/>
- [3] L. Robotics, "Liquid robotics." [Online]. Available: <https://www.liquid-robotics.com/wave-glider/customer-stories/>
- [4] Saildrone, "Saildrone." [Online]. Available: <https://www.saildrone.com/technology/vehicles>
- [5] R. C. Leegstra, M. De Paula, I. Carlucho, F. J. Solari, A. F. Rozenfeld, G. G. Acosta, B. V. Menna, R. J. de la Vega, L. M. Arrien, H. J. Curti, and S. A. Villar, "MACÁBOT: Prototipo de vehículo autónomo de superficie (asv)," *Revista Tecnología y Ciencia*, no. 36, p. 142–154, oct. 2019. [Online]. Available: <https://rtyc.utn.edu.ar/index.php/rtyc/article/view/344>
- [6] J. L. Rosendo and F. Garelli, "Proyecto genesas: Integración y puesta en marcha de robot autónomo de superficie," in *28 Congreso Argentino de Control Automático. Asociación Argentina de Control Automático (AADECA)*, 2023.

- [7] L. Garberoglio, P. Moreno, I. Mas, and J. I. Giribet, "Autonomous vehicles for outdoor multidomain mapping," in *2018 IEEE Biennial Congress of Argentina (ARGENCON)*, 2018, pp. 1–8.
- [8] L. Garberoglio, P. Moreno, I. Mas, and J. Giribet, "Control coordinado de vehículos aéreos y acuáticos para relevamiento batimétrico en aguas poco profundas," *Revista Electrón*, vol. 6, no. 2, pp. 86–95, 2022, accessed: 2024-03-26. [Online]. Available: <https://doi.org/10.37537/rev.elektron.6.2.164.2022>
- [9] E. Tupper, *Introduction to Naval Architecture*, 3rd ed. Oxford, England: Butterworth-Heinemann, Jun. 1996.
- [10] B. Allotta, R. Costanzi, F. Fanelli, N. Monni, L. Paolucci, and A. Ridolfi, "Sea currents estimation during auv navigation using unscented kalman filter," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 13 668–13 673, 2017, 20th IFAC World Congress. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S240589631733450X>
- [11] "Sitontop," 2024. [Online]. Available: <https://www.sitontop.com.ar/>
- [12] I. Buchmann, *Batteries in a portable world*. Cadex Electronics Inc, 2016.
- [13] Electroconversiones, 2024. [Online]. Available: <https://electroconversiones.com.ar/>
- [14] Victron Energy, "Which solar charge controller: Pwm or mppt?" 2020. [Online]. Available: <https://www.victronenergy.com/upload/documents/Technical-Information-Which-solar-charge-controller-PWM-or-MPPT.pdf>
- [15] "Onshape," 2024. [Online]. Available: <https://www.onshape.com/>
- [16] SolarLine, 2024. [Online]. Available: <https://www.solaryeolicasrl.com.ar/>
- [17] Victron Energy, 2024. [Online]. Available: <https://www.victronenergy.com/>
- [18] Spektrum, 2024. [Online]. Available: <https://www.spektrumrc.com/>

Dyna-1: Desarrollo, Construcción y Pruebas Experimentales de un Robot Cuadrúpedo de Diseño Abierto

Tadeo Casiraghi*, Gabriel Torre*[‡], Roberto Bunge*[†], Ignacio Mas*[†]

[†] CONICET, Argentina

* Universidad de San Andrés, Departamento de Ingeniería, LINAR, Buenos Aires, Argentina

[‡] Universidad de Buenos Aires, Facultad de Ingeniería, Instituto de Ingeniería Biomédica

Resumen

Presentamos un nuevo diseño de cuadrúpedo de 12 grados de libertad, centrado en reducir el costo y la complejidad de fabricación y maximizar el uso de componentes disponibles regionalmente. Para facilitar el ensamblado, las piernas son modulares, y los motores están lo mas cerca posible del cuerpo para reducir su inercia. Se utilizan motores brushless junto a controladores Odrive que permiten un control simple con su interfaz por CAN. El cuerpo contiene los motores de los hombros y toda la electrónica necesaria para su funcionamiento. Para controlar el cuadrúpedo se implementa el sistema operativo ROS2. Todos los procesos cruciales para el funcionamiento del robot se corren en la computadora a bordo, y los demás en una computadora externa. Se demuestra experimentalmente que cada pierna es capaz de saltar hasta 30 cm con un peso similar a un cuarto del cuerpo. Se ensayó el robot con peso agregado de hasta 4 kg y se observa buen rendimiento. Se creó un simulador en Pybullet para evaluar algoritmos de autonomía de manera eficiente. El costo total del robot es de aproximadamente 3000 USD. Todo el software y los modelos del hardware son de diseño abierto y pueden encontrarse en un repositorio público. <https://github.com/udesa-ai/dyna1-quadruped>

Sesión 6: Aprendizaje Automático en Robótica I

Simulador de escenas forestales para la segmentación de bosques utilizando VANTs

Francisco Raverta Capua
Instituto de Ciencias de la Computación
Universidad de Buenos Aires
C1428EGA, CABA, Argentina
Email: fraverta@icc.fcen.uba.ar

Juan Schandin
Departamento de Computación
Universidad de Buenos Aires
C1428EGA, CABA, Argentina
Email: jschandin@dc.uba.ar

Pablo De Cristóforis
Departamento de Computación
Universidad de Buenos Aires
C1428EGA, CABA, Argentina
Email: pdecris@dc.uba.ar

Abstract—El sensado remoto, mediante el uso de vehículos aéreos no tripulados (VANTs), ha crecido sostenidamente en los últimos años, junto con el uso del aprendizaje automático para el procesamiento de datos. La creación de *datasets* de ambientes naturales es costosa económica y temporalmente, ya que requiere de sensores y equipamiento de alta precisión, y de tareas de procesamiento tales como el etiquetado manual en imágenes o en nubes de puntos. Algunas áreas naturales, como los bosques, pueden resultar de difícil acceso o peligrosas para el ser humano. Para poder promover la investigación en tareas que requieran de *datasets* específicos de estos ambientes, en este trabajo desarrollamos un simulador que genera procedimentalmente escenas forestales sintéticas realistas. A partir de este simulador, buscamos responder la pregunta de si es posible utilizar datos sintéticos para el entrenamiento de redes de aprendizaje profundo basadas en nubes de puntos para la segmentación de escenas forestales sin la necesidad de contar con grandes volúmenes de datos reales de estos ambientes. Tanto el simulador como el dataset generado se encuentran disponibles públicamente on-line como parte de este trabajo.

Detección Automática de Ciervos de los Pantanos Utilizando Imágenes Aéreas de UAV

Agustín Roca¹, Gabriel Torre^{1,4}, Juan Giribet^{1,5}, Gastón Castro^{1,5}, Leonardo Colombo³, Ignacio Mas^{1,5}, Javier Pereira^{2,5}

¹Laboratorio de Inteligencia Artificial y Robótica - Universidad de San Andrés, Buenos Aires, Argentina.

²Museo Argentino de Ciencias Naturales "Bernardino Rivadavia", CABA, Argentina.

³Centro de Automática y Robótica - CSIC, Madrid, España

⁴Instituto de Ingeniería Biomédica, Facultad de Ingeniería, Universidad de Buenos Aires

⁵CONICET - Argentina

Resumen

La detección de fauna constituye un desafío esencial en la gestión de la vida silvestre y la conservación. Este trabajo explora cómo técnicas de visión por computadora y aprendizaje profundo pueden servir para la identificación de ciervos y venados en sus entornos naturales. Utilizando datos aéreos de alta resolución capturados por vehículos aéreos no tripulados (UAVs), se implementaron algoritmos de inteligencia artificial para la detección automática de ciervos de los pantanos (*Blastocerus dichotomus*) en el Delta del Paraná, Argentina. Se recopiló una base de datos con 39,798 imágenes aéreas, analizadas manualmente por científicos y voluntarios, identificando 231 ciervos. Se realizó un entrenamiento del modelo YOLOv8n-seg, obteniendo un mAP_{10} de 0.950, mostrando resultados prometedores en la detección automática de esta especie. Además, se evaluó el modelo en la detección de venados de las pampas (*Ozotoceros bezoarticus*) en el Parque Nacional Campos del Tuyú, aunque con un rendimiento inferior, destacando la necesidad de ampliar la base de datos con imágenes de esta especie. Este trabajo establece un marco para el uso de UAVs y técnicas de inteligencia artificial en la monitorización de fauna silvestre, con potencial para ser aplicado en la conservación de diversas especies. En futuras investigaciones, se planea recolectar más datos utilizando sensores térmicos para mejorar la detección en diferentes entornos.

Segmentación Semi-Supervisada de Malezas en Entornos Agrícola de Alta Complejidad

1st Fernando Blanco
Departamento de Ciencias de la Computación
FCEIA-UNR
Rosario, Argentina
ferblanco203@gmail.com

2nd Taihú Pire
CIFASIS
CONICET-UNR
Rosario, Argentina
pire@cifasis-conicet.gov.ar

3rd Claus Smitt
AgRobotics & Engineering
University of Bonn
Bonn, Alemania
csmitt@uni-bonn.de

Resumen—La robótica agrícola viene a facilitar el trabajo agropecuario, mejorándolo en eficiencia, eficacia y ahorro tanto de productos como de trabajo humano. Dentro de las áreas de robótica agrícola, una de suma importancia es la detección de maleza, la cual consiste en mediante sensores encontrar malezas en los campos. Nosotros presentamos un método de detección panóptico de malezas, el cual utilizando computer vision es capaz de identificar malezas, dándonos su forma y considerando cada una como una maleza individual. Esto es realmente útil para luego realizar estudios de los campos y/o actuar en consecuencia a la disposición de malezas. Logramos entrenar un modelo Mask2Former que es capaz de detectar malezas sobre cultivo, además lo probamos con un video para ver como se comportaba en movimiento obteniendo resultados aceptables y una buena velocidad de procesamiento.

Keywords—Visión Artificial, Aprendizaje Profundo, Segmentación Panóptica, Agricultura, Aprendizaje Semi-Supervisado

I. INTRODUCCIÓN

Hace ya varios años, el sector agrícola muestra gran interés en la utilización de robots autónomos para realizar diversas tareas como siembra, riego, remoción de maleza o cosecha. Por un lado, el uso de robots autónomos permite reducir la cantidad de horas hombre y por otro, realizar la tarea de una manera más sustentable y sostenible, haciendo un uso eficiente de los recursos al mismo tiempo que se reduce el impacto ambiental [1], [2].

La detección de malezas es de gran interés, ya que pueden darle al productor agrícola información sobre la distribución de las mismas en el lote, estimar la cantidad y así poder tomar decisiones al respecto como podría hacer un desmalezado preciso.

La existencia de malezas pueden perjudicar a los cultivos al generar competencia con ellos. Conocer dónde, cuáles y cuántas malezas tiene, puede ser útil a la hora de estudiar el lote, ya que beneficia a la hora de decidir como poder tratarlo. Para tratar esto se han desarrollado diversos enfoques, sin embargo, no se atacó desde el lado de segmentación panóptica, lo cual puede ser útil para contabilizar y separar una maleza de otra.

En este trabajo intentaremos detectar instancias de malezas en entornos de alta complejidad mediante segmentación panóptica utilizando métodos de inteligencia artificial, y de esta manera lograr obtener las distintas malezas que se ven en una imagen como así también la forma de cada una.

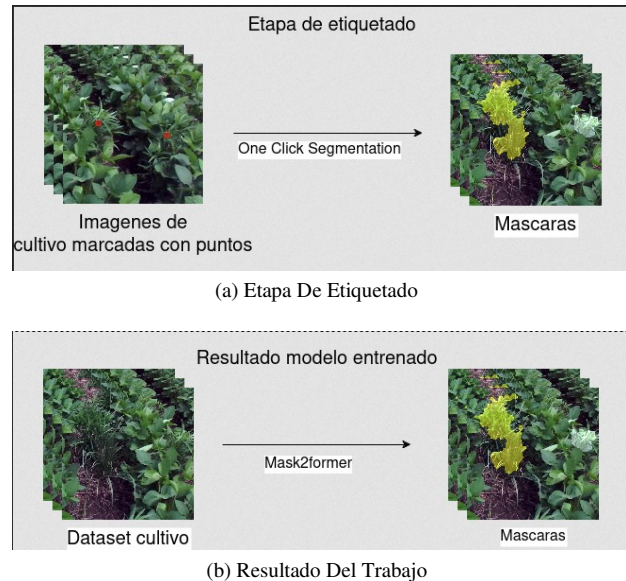


Figura 1. Representación del método usado en este trabajo. En 1a se ve la etapa de etiquetado donde se le pasó imágenes con puntos (clicks) a un One Click Segmentation y se obtuvo pseudo máscaras. En 1b se muestra el resultado del trabajo, el cual es un modelo entrenado con las pseudo máscaras que se lo alimenta con imágenes de cultivo y se obtiene las detecciones panópticas de las malezas.

El resto del trabajo está estructurado de la siguiente manera, en la sección II se muestra el estado del arte de trabajos relacionados con la detección en el sector agrícola. En la sección III se explica el proceso de como logramos realizar detecciones panópticas de malezas sobre cultivo. En la sección IV se muestran una serie de experimentos y resultados de los mismos realizados por nosotros. En la sección V concluimos el trabajo con un resumen de lo propuesto y un análisis de los resultados obtenidos, además se mencionan posibles trabajos a futuros para extender lo exployado en este artículo.

II. TRABAJO RELACIONADO

II-A. Monitoreo de cultivos

Dentro de la robótica para la agricultura tenemos lo que son robots de monitoreo de cultivo, estos inspeccionan cultivos de forma autónoma lo cual es útil para recolectar información de las plantaciones, como puede ser la contabilización de frutos,

estado de salud de las plantas y el suelo, entre otras. Entre los trabajos dentro de esta área se encuentran PATHoBot [3], un robot de monitoreo de pimientos y tomates en invernaderos desarrollado por la universidad de Bonn, Alemania. También, en la misma universidad, se ha desarrollado el prototipo BonnBot-I [4], un robot que realiza desmalezado y monitoreo de campo en diferentes ciclos de crecimiento. AgBot II [5], un robot desmalezador autónomo desarrollado por la universidad tecnológica de Queensland, Australia. En Argentina, el instituto CIFASIS (CONICET-UNR) ha desarrollado un prototipo de robot desmalezador. El mismo ha permitido la generación del Rosario dataset [6], utilizado para este trabajo.

El campo agrícola es un entorno desafiante para las tareas de monitoreo debido a que es altamente repetitivo, dinámico debido al movimiento de las hojas producido por el viento, presenta cambios de iluminación y movimientos bruscos producidos por el terreno irregular, entre otros [7]. Por lo tanto, el desarrollo de métodos de monitoreo robustos a estos problemas mencionados, es de gran utilidad para la industria agrícola.

Actualmente, los trabajos del estado del arte en monitoreo de cultivos agrícolas, hacen uso de técnicas de aprendizaje profundo (*deep learning*), cuyos métodos requieren de una gran cantidad de datos para ser entrenados. Para realizar estos monitoreos, se utilizan técnicas de entendimiento de escena (*scene understanding*) y segmentación semántica. En [8] se presenta una red profunda PAG-NeRF capaz de ubicar frutos en el espacio utilizando estas dos técnicas.

II-B. Segmentación Panóptica

La segmentación panóptica es una combinación de segmentación semántica y segmentación de instancias [9]. La segmentación semántica consisten en asignarle a cada píxel una etiqueta indicando a qué clase pertenece, a elementos incontables (ej: tierra, cielo, etc.) y contables (ej: planta, maleza, etc.). La segmentación de instancias consiste en diferenciar objetos contables de una misma clase, asignándoles un identificador único a cada una. Esta representación provee gran cantidad de información para una comprensión del entorno. Este tipo de predicción se ha empleado en diversas aplicaciones de agricultura, mapeo tales como contabilización de frutas [10], [11], detección de malezas en campo abierto [12].

Uno de los primeros trabajos en resolver esta tarea con redes neuronales convolucionales fue PanopticDeepLab [13], prediciendo segmentación semántica, centros de instancias y las distancias a los mismos por separado. Luego estas predicciones se combinan para obtener la predicción panóptica. Actualmente, el estado del arte para esta tarea está dominado por [14], [15], siendo Mask2Former [15] una de las arquitecturas más utilizada. Esta arquitectura emplea esquema de encoder-decoder en el espacio de imágenes y luego los mapas de activación de diferentes resoluciones son progresivamente destilados en predicciones de máscaras con clases y etiquetas individuales asociadas. Al igual que otros métodos de aprendizaje profundo, esta arquitectura requiere una gran cantidad de datos etiquetados con máscaras para ser entrenado.

Una forma de reducir la cantidad de anotaciones es el uso de métodos de aprendizaje semisupervisado.

II-C. Segmentación semisupervisada con etiquetas parciales

Anteriormente, se han desarrollado métodos para entrenar modelos de segmentación con etiquetas parciales en lugar de máscaras detalladas (ej: clicks [16], cajas 2D [17]). Estos métodos permite, a partir de un número reducido de etiquetas detalladas y muchas etiquetas parciales, entrenar un modelo para generar una gran cantidad de pseudo etiquetas de segmentación. Estas pueden ser empleadas para entrenar un modelo de segmentación estándar en forma supervisada.

Recientemente, Zimmer *et al.* [12] presentaron un método para obtener pseudo etiquetas panópticas de malezas en campo abierto a partir de clicks, entrenándose con escasas imágenes etiquetadas con máscaras. Usando una arquitectura similar a [13], este método extrae puntos aleatorios cercanos al centroide de las etiquetas. Estos son suavizados y agregados como un canal extra a la entrada, aprendiendo así la relación entre puntos/clicks y predicciones panópticas en la imagen. Este método solo fue evaluado en datasets de complejidad reducida con vista superior de cultivos en etapas tempranas de desarrollo.

En el presente trabajo evaluamos el uso de este método como método para reducir el esfuerzo de anotación en un dominio agrícola de gran complejidad [6]. A su vez, evaluamos la viabilidad de usar las pseudo etiquetas producidas para entrenar un modelo Mask2Former [15] para detección de malezas.

III. MÉTODO PROPUESTO

La segmentación panóptica de malezas nos permite diferenciar distintas instancias de malezas y ver sus formas. Para esto entrenamos una red Mask2Former [15].

Para entrenar el Mask2Former conseguimos del Rosario Dataset [6] imágenes de un cultivo de soja, donde las malezas no eran fáciles de ver, ya que algunas se encontraban creciendo entre los cultivos, lo que dificultaba dibujar las máscaras a mano.

Para evitarnos esa tarea optamos por entrenar un modelo de One-Click Segmentation [12], por lo que nosotros simplemente marcamos las malezas con puntos y este modelo se encargó de generar las máscaras para nuestro dataset. Las máscaras generadas por One Click Segmentation, eran un poco sucias, ya que eran des conexas y contaban con algunos píxeles sueltos. Esto a la hora de escribir las anotaciones se tornaba un poco molesto, ya que se marcaban como distintas a pesar de pertenecer a la misma máscara. Para evitar esto, sometimos a las máscaras a un procesamiento de imágenes utilizando operaciones de open y close, lo cual nos emprolijó las máscaras para poder anotarlas correctamente y pasarlas al entrenador del Mask2Former.

Una vez entrenado el modelo, lo utilizamos sobre un vídeo perteneciente al Rosario Dataset para probar que tal funciona. Para esto leímos frame por frame un vídeo, pasando cada uno por el modelo y pintando los resultados sobre estos.

Finalmente, generamos un vídeo nuevo con el resultado del procesamiento del anterior.

En la figura 2 se puede ver un diagrama del flujo de trabajo del método propuesto

IV. EXPERIMENTOS

En esta sección se presentan los experimentos que validan el método propuesto. La computadora utilizada para realizar los experimentos consta de una CPU Intel i7-4790 con 16 GB de RAM y una GPU ASUS GeForce 1070.

Como se comentó en la sección III, primero entrenamos un One Click Segmentation, el cual usamos para generar mascarar, como las que se ven en la Figura 3. Para este entrenamiento se utilizaron 10 imágenes. Luego con eso generamos 155 imágenes de mascarar, de las cuales usamos 125 de entrenamiento, 15 de validación y 15 de evaluación para entrenar un Mask2Former con el cual tuvimos resultados de validación bastante satisfactorios, mostrados en la Figura 4.

Se testeó el modelo resultando utilizando 12 imágenes con mascarar hechas a mano, para sacar distintas métricas la Tabla I, y obtuvimos la matriz de confusión mostrada en la Tabla II.

Para probar bajo un contexto similar al que se lo quiere usar, pasamos un vídeo grabado por el robot desmalezado de CIFASIS por el modelo, cuyo resultado se ve en la Figura 5. En las imágenes podemos ver que las malezas las detectas bien (Figura 5a), sin embargo, hay plantas de cultivos y otras cosas que detecta como falsos positivos (Figura 5b).

Tabla I
DESEMPEÑO EN TAREAS DE SEGMENTACIÓN PANÓPTICA

	Mask2Former
AP [%]	8.232311210158785
AP 50 [%]	25.182696841112684
InstanceW eedIoU [%]	24.41
SemanticW eedIoU [%]	47.96

		Predicted	
		Negative	Positive
True	Negative	0.9971	0.0029
	Positive	0.7306	0.2694

Tabla II
MATRIZ DE CONFUSIÓN MODELO ENTRENADO

V. CONCLUSIONES Y TRABAJO FUTURO

En este trabajo nosotros propusimos e intentamos hacer una detección panóptica de malezas utilizando una red Mask2Former, entrenándola con pseudo mascarar generadas por One Click Segmentation, lo cual nos permitiría contabilizar y conocer ubicación y forma de cada maleza. Con esto probaríamos, además, que tan bien funciona One Click Segmentation en contextos complejos y que tan útil son sus resultados para entrenar otra red.

Las métricas obtenidas no se ven muy buenas, pareciera no ser preciso y las predicciones de malezas no coincidir con los

datos reales. Mirando la matriz de confusión podemos deducir que nuestro modelo tiene bastante falsos negativos y bajos verdaderos positivos. Sin embargo, estos resultados no nos sorprenden, considerando la falta de refinamiento del trabajo, son valores esperables, probablemente mejorando el dataset y el etiquetado, las métricas mejoren, como así también la matriz de confusión se nos acerque más a una matriz identidad.

Al probar el resultado en un vídeo pudimos ver como se comporta el modelo en una situación de movimiento, donde las malezas se ven detectadas consistentemente, aunque cuenta con varios falsos positivos. Estimamos que los falsos positivos se deben al dataset, probablemente ampliando la cantidad de imágenes usadas para entrenar el Mask2Former, mejorando las etiquetas usadas para One Click Segmentation y usando imágenes más diversas, se podrían disminuir. También vimos que la misma maleza de un frame a otro la detecta como si fueran una nueva, esto se ve con el cambio de color en la máscara dibujada, lo cual se debe a que el modelo se ejecuta por frame ignorando que perteneces a un vídeo secuencial, por lo que ignora el contexto previo.

A futuro, nos interesa lograr que reconozca la misma maleza dentro de un contexto de movimiento, así poder generar un mapa de malezas dentro de un lote. Para esto nos gustaría agregar este modelo en un sistema de SLAM, el cual luego quisiéramos poder colocar en un robot que recorra de forma autónoma los lotes de cultivo. Otro posible trabajo a futuro, es reconocer y diferenciar distintas especies de malezas, para lo cual se debería anotar un dataset con las distintas clases para entrenar el modelo. Juntando ambos trabajos propuestos se podría generar un mapa de malezas mucho más rico en información del lote.

AGRADECIMIENTOS

Este trabajo fue apoyado por CONICET (PIBAA N° 0042), AGENCIA I+D+i (PICT 2021-570) y Universidad Nacional de Rosario (PCCT-UNR 80020220600072UR).

REFERENCIAS

- [1] F. A. Auat Cheein and R. Carelli, "Agricultural Robotics: Unmanned Robotic Service Units in Agricultural Tasks," *IEEE Industrial Electronics Magazine*, vol. 7, no. 3, pp. 48–58, 2013.
- [2] C. W. Bac, E. J. van Henten, J. Hemming, and Y. Edan, "Harvesting Robots for High-value Crops: State-of-the-art Review and Challenges Ahead," *Journal of Field Robotics*, vol. 31, no. 6, pp. 888–911, 2014.
- [3] C. Smitt, M. Halstead, T. Zaenker, M. Bennewitz, and C. McCool, "Pathobot: A robot for glasshouse crop phenotyping and intervention," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 2324–2330.
- [4] A. Ahmadi, M. Halstead, and C. McCool, "Bonnbot-i: A precise weed management and crop monitoring platform," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 9202–9209.
- [5] O. J. Bawden, "Design of a lightweight, modular robotic vehicle for the sustainable intensification of broadacre agriculture," Ph.D. dissertation, Queensland University of Technology, 2015.
- [6] T. Pire, M. Mujica, J. Civera, and E. Kofman, "The Rosario Dataset: Multisensor Data for Localization and Mapping in Agricultural Environments," *Intl. J. of Robotics Research*, vol. 38, no. 6, pp. 633–641, 2019.
- [7] J. Cremona, R. Comelli, and T. Pire, "Experimental evaluation of Visual-Inertial Odometry systems for arable farming," *Journal of Field Robotics*, vol. 39, no. 7, pp. 1123–1137, 2022.

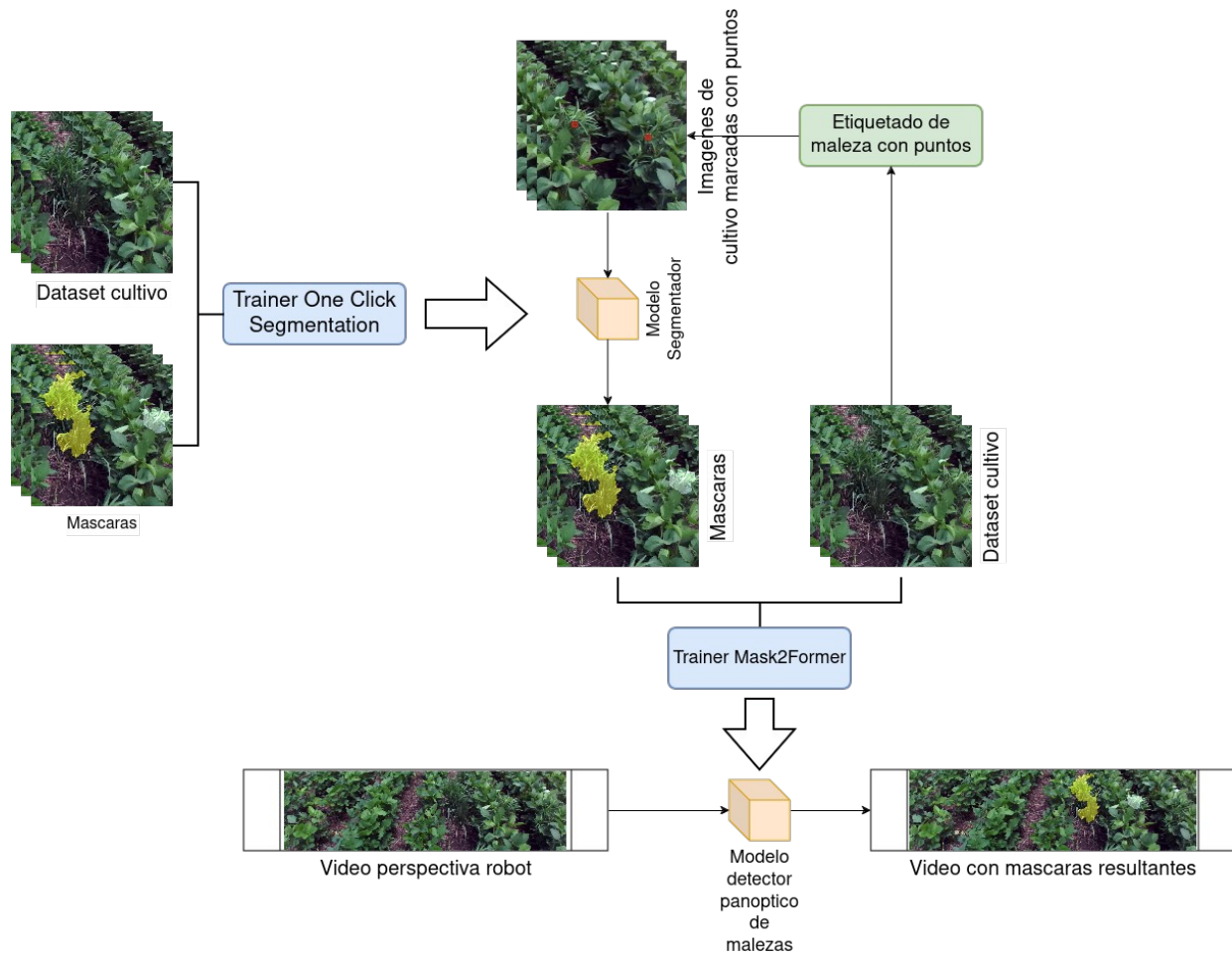
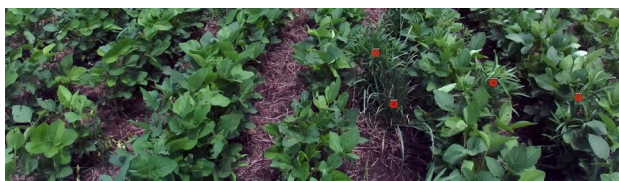
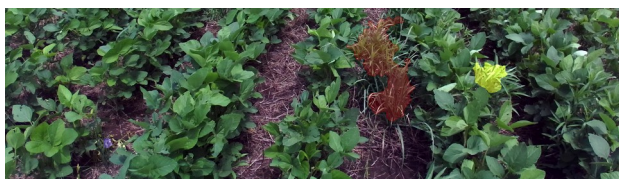


Figura 2. Flujo de trabajo del método implementado, los rectángulos celeste son entrenadores de IA, los cubos naranjas son modelos de IA entrenados, los rectángulos verdes representan procesos manuales



(a) Imagen anotada con puntos



(b) Mascarlas generadas por One Click Segmentation

Figura 3. Resultados de One Click Segmentation

- [8] C. Smitt, M. Halstead, P. Zimmer, T. Läbe, E. Guclu, C. Stachniss, and C. McCool, "Pag-nerf: Towards fast and efficient end-to-end panoptic 3d representations for agricultural robotics," *IEEE Robotics and Automation Letters*, vol. 9, no. 1, pp. 907–914, 2023.
- [9] A. Kirillov, K. He, R. Girshick, C. Rother, and P. Dollár, "Panoptic segmentation," in *Proceedings of the IEEE/CVF conference on computer*

- vision and pattern recognition*, 2019, pp. 9404–9413.
- [10] M. Halstead, A. Ahmadi, C. Smitt, O. Schmittmann, and C. McCool, "Crop agnostic monitoring driven by deep learning," *Frontiers in plant science*, vol. 12, 2021.
- [11] Y. Pan, F. Magistri, T. Läbe, E. Marks, C. Smitt, C. McCool, J. Behley, and C. Stachniss, "Panoptic mapping with fruit completion and pose estimation for horticultural robots," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 4226–4233.
- [12] P. Zimmer, M. Halstead, and C. McCool, "Panoptic one-click segmentation: applied to agricultural data," *IEEE Robotics and Automation Letters*, vol. 8, no. 5, pp. 2478–2485, 2023.
- [13] B. Cheng, M. D. Collins, Y. Zhu, T. Liu, T. S. Huang, H. Adam, and L.-C. Chen, "Panoptic-deeplab: A simple, strong, and fast baseline for bottom-up panoptic segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 12 475–12 485.
- [14] J. Jain, J. Li, M. T. Chiu, A. Hassani, N. Orlov, and H. Shi, "Oneformer: One transformer to rule universal image segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 2989–2998.
- [15] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar, "Masked-attention mask transformer for universal image segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 1290–1299.
- [16] N. Xu, B. Price, S. Cohen, J. Yang, and T. S. Huang, "Deep interactive object selection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 373–381.



(a) Mascaras de validación



(b) Resultado del modelo

Figura 4. Resultados de validación del modelo entrenado



(a) Malezas detectadas en vídeo



(b) Algunas plantas marcadas como falsos positivos

Figura 5. Frames del vídeo generado con el modelo entrenado

- [17] N. Xu, B. Price, S. Cohen, J. Yang, and T. Huang, "Deep grabcut for object selection," *arXiv preprint arXiv:1707.00243*, 2017.

Sesión 7: Aprendizaje Automático en Robótica II

Locomoción en Simulación de Robot Hexápodo sobre Terrenos Irregulares utilizando Aprendizaje por Refuerzo Profundo

Tania Sofía Ferreyra^{*†}, Nicolás Romero^{*†}, Matías Sandacz[‡], Emanuel Slawinski[‡] and Pablo De Cristóforis[†]

^{*}Estos autores contribuyeron en partes iguales

[†] Departamento de Ciencias de la Computación, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires, Argentina

[‡] Instituto de Automática, San Juan, Argentina

Resumen—El control de los robots con patas articuladas resulta complejo debido a la alta dimensionalidad del sistema. Este trabajo aborda el problema de locomoción en robots hexápodos ciegos mediante el uso de Aprendizaje por Refuerzo Profundo, como alternativa a las estrategias de control tradicional. Nos basamos en el enfoque de aprendizaje maestro-estudiante propuesto en otro trabajo para un robot cuadrúpedo de 12 grados de libertad y lo extendimos para un robot hexápodo de 16 grados de libertad. Implementamos y evaluamos el módulo del maestro para el robot PhantomX Mark II utilizando el simulador Gazebo, ROS para la comunicación entre los componentes de control y la librería StableBaselines para los algoritmos de aprendizaje por refuerzo. Los entrenamientos se realizaron con terrenos generados con ruido de Perlin y terrenos con bloques de altura aleatoria, adaptativamente seleccionados para aumentar la dificultad a medida que progresa el aprendizaje. Los resultados alcanzados muestran la factibilidad de utilizar Aprendizaje por Refuerzo Profundo para abordar el problema de la locomoción en robots hexápodos sobre terrenos irregulares.

Palabras clave—robot hexápodo, locomoción, control, aprendizaje por refuerzo profundo

I. INTRODUCCIÓN

Los robots con patas articuladas se destacan por su capacidad para realizar tareas en entornos desafiantes, hostiles o inaccesibles para otros robots terrestres con sistemas de locomoción basado en ruedas u orugas, al mismo tiempo que pueden transportar más carga y operar durante más tiempo que los vehículos aéreos no tripulados. Sus habilidades para realizar movimientos omnidireccionales y atravesar terrenos irregulares los convierten en una excelente opción para diversas aplicaciones, como las misiones de Búsqueda y Rescate Urbano (USAR por sus siglas en inglés) en construcciones colapsadas en situaciones de catástrofe, para operar en ambientes subterráneos como minas e incluso para la exploración extraplanetaria. Sin embargo, es sabido que el control de movimiento de este tipo de robots es sumamente desafiante debido a la cantidad de grados de libertad. Una alternativa a las estrategias de control tradicional es el uso de aprendizaje por refuerzo (RL del inglés Reinforcement Learning).

Existen algunos trabajos en el estado del arte que han explorado el uso de RL específicamente en robots hexápodos. Trabajos como [1] usan RL en simulación para que un hexápodo aprenda a caminar en terreno plano. En [2] se implementa RL para ajustar los parámetros de un generador de patrón central (CPG por sus siglas en inglés) en un robot real, pero los

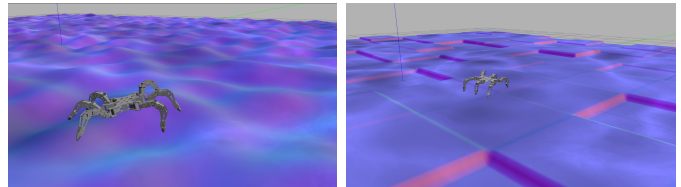


Figura 1: Ejemplo de los terrenos usados durante el aprendizaje: colinas (izquierda) y bloques (derecha).

terrenos utilizados son prácticamente planos. En [3] el robot es solo simulado y el terreno tiene irregularidades no muy pronunciadas ni diversas. En [4], sí se consideran terrenos más desafiantes y se propone un sistema de dos etapas: una que aprende por DRL (Deep Reinforcement Learning) una caminata óptima para cada terreno dentro de N posibles tipos y un clasificador que aprende a predecir el tipo de terreno para seleccionar la política experta correspondiente. El trabajo presenta resultados muy interesantes, pero únicamente en simulación y utilizando un modelo de robot simplificado a figuras geométricas sencillas, lo que impide asegurar que el resultado sea trasladable al mundo real.

Existe más desarrollo realizado para robots cuadrúpedos, donde hay múltiples ejemplos de sistemas llevados al mundo real, con resultados sumamente interesantes. En [5] se entrena con RL una política que luego se transfiere a un robot cuadrúpedo real y le permite moverse en escaleras, terrenos con agujeros y rocas. Aún más impresionante son los resultados obtenidos en [6], donde logran que un cuadrúpedo ANYmal pueda atravesar escaleras, terreno boscoso inclinado, vegetación, nieve entre otros terrenos. No obstante, ambos trabajos hacen uso de un sistema de visión, que escapa al alcance de este trabajo. Por otro lado, en [7] obtienen resultados similares para un robot ANYmal utilizando únicamente medidas propiocepticas, entrenando en simulación y pudiendo transferir la política aprendida al robot real sin ajuste adicional. Para ello proponen una estrategia maestro-estudiante, que consiste en entrenar en simulación un robot maestro con acceso a información privilegiada obtenida del simulador y luego usar la política obtenida para entrenar por aprendizaje supervisado un robot estudiante que solo tiene acceso a las mediciones que estarían disponibles en el robot real.

En este trabajo presentamos el módulo maestro de un sistema de locomoción para robots hexápodos utilizando únicamente mediciones propioceptivas y Aprendizaje por Refuerzo Profundo, basado en el enfoque maestro-alumno que se presenta en [7]. Específicamente, el sistema está diseñado para el robot PhantomX Mark II. A diferencia del trabajo antes mencionado, hacemos uso de herramientas ampliamente utilizadas por la comunidad: Gazebo [8] como plataforma de simulación, ROS [9] para comunicación entre los distintos módulos del sistema y la librería Stable Baselines 3 [10] para los algoritmos de Aprendizaje por Refuerzo Profundo. Los terrenos utilizados para el entrenamiento, mostrados en la Figura 1, fueron terrenos de colinas, generados con ruido de Perlin, y terrenos de bloques, de altura aleatoria, seleccionados adaptativamente con un filtro de partículas para aumentar progresivamente la dificultad del terreno con el que se entrena.

II. SISTEMA

En este trabajo nuestro objetivo es enseñar a un robot hexápodo a caminar en terrenos irregulares usando únicamente sensores propioceptivos. El robot utilizado para este trabajo es el PhantomX Mark II, cuyo modelo para el simulador en formato URDF fue sacado de [11]. La red neuronal que se entrena mediante DRL modifica la trayectoria de cada pata fijada *a priori* por el controlador de bajo nivel, que por sí solo mantiene al robot caminando en el lugar en un terreno plano, para que se adapte al terreno irregular y siga el comando de velocidad deseado. Para ello se modifican la frecuencia de la trayectoria de cada pata, f_i , y la posición final de referencia adicionando un diferencial, Δr_i , como se explica más adelante en la sección II-C. El sistema aprende a partir de un currículo de terrenos adaptativo. Se definen dos tipos de terreno, bloques y colinas, en los que se desea entrenar, y se utiliza un filtro de partículas para seleccionar terrenos con dificultad creciente para la etapa de entrenamiento en la que se encuentre.

II-A. Estructura General

La estructura general del control, mostrada en la Figura 2, se compone, en primer lugar, de una red neuronal, que recibe comandos de velocidad lineal y angular, y cuya salida son los valores de frecuencia f_i de cada pata y los diferenciales de movimiento Δr_i . Los valores de frecuencia sirven de entrada al bloque de generación de trayectoria que define el movimiento a realizar por cada pata. El valor de posición objetivo para cada pata es el obtenido por el generador de trayectoria más el diferencial de movimiento Δr_i determinado por la red neuronal. Este valor final se procesa con las ecuaciones de cinemática inversa para obtener las posiciones deseadas para cada articulación, que son enviadas a los controladores PID de cada articulación del robot. Finalmente, el robot y el simulador devuelven las observaciones necesarias para armar el vector de observación de la red neuronal.

II-B. Modelo Cinemático

Como podemos ver en la Figura 3, es posible establecer la posición de la pata a partir de los ángulos de coxa, femur y

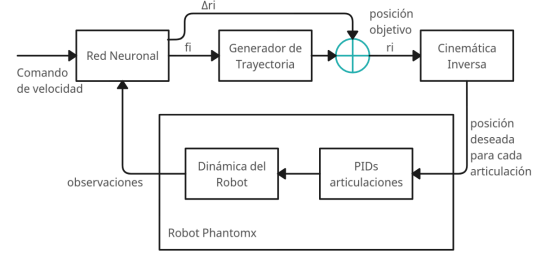


Figura 2: Diagrama en bloques del sistema de control del robot.

tibia (θ_C , θ_F y θ_T , respectivamente) conociendo las longitudes de cada parte (LC , LF y LT , respectivamente) haciendo uso del eje auxiliar r_{xy} :

$$r_{xy} = LC + LF \cos(\theta_F) + LT \cos(\theta_F + \theta_T)$$

$$z = LF \sin(\theta_F) + LT \sin(\theta_F + \theta_T)$$

$$x = r_{xy} \cos(\theta_C)$$

$$y = r_{xy} \sin(\theta_C)$$

A partir de estas relaciones podemos obtener las ecuaciones de cinemática inversa:

$$\theta_C = \text{atan2}(y, x)$$

$$u = \sqrt{x^2 + y^2} - LC$$

$$c2 = \cos(\theta_T) = \frac{u^2 + z^2 - LF^2 - LT^2}{2LF LT}$$

$$s2 = \sin(\theta_T) = -\sqrt{1 - c2^2}$$

$$c1 = \cos(\theta_F) = \frac{u(LF + LT c2) + z LT s2}{u^2 + z^2}$$

$$s1 = \sin(\theta_F) = \frac{z(LF + LT c2) - u LT s2}{u^2 + z^2}$$

$$\theta_F = \text{atan2}(s1, c1)$$

$$\theta_T = \text{atan2}(s2, c2)$$

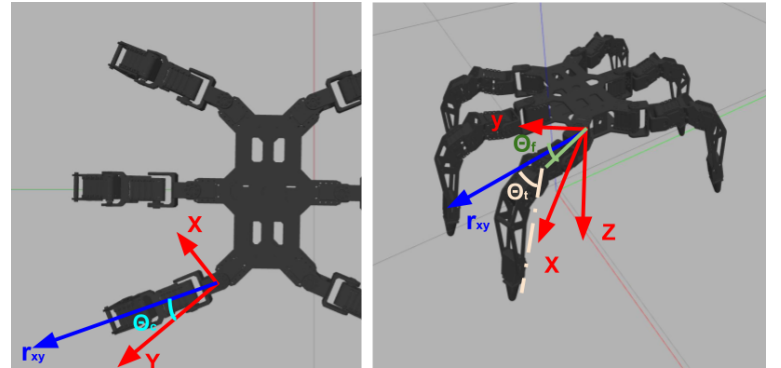


Figura 3: Sistema de coordenadas para obtener las ecuaciones cinemáticas para la pata derecha delantera.

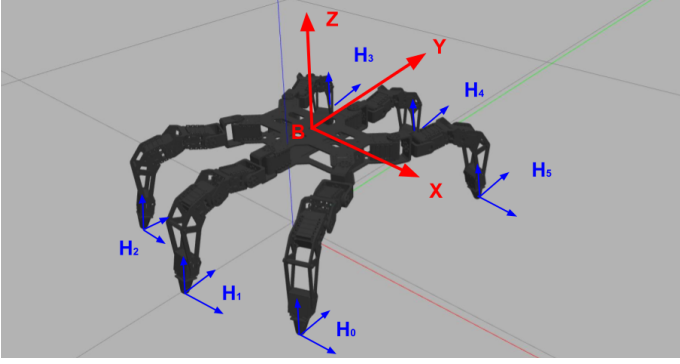


Figura 4: Robot PhantomX con los sistemas de coordenadas de referencia H_i (azul) y el sistema de coordenadas de la base B (rojo).

Como estas ecuaciones asumen cierta orientación del sistema de coordenadas de cada articulación y que todos los ángulos en cero corresponden a la pata estirada recta, los valores de comando real a enviar a Gazebo son: $\theta_{coxa} = \theta_C$, $\theta_{femur} = -\theta_F$ y $\theta_{tibia} = \theta_T + \pi/2$.

II-C. Control de Bajo Nivel

El control de bajo nivel consta de varias etapas. La primera es la generación de trayectoria de referencia para cada pata. En cada instante de tiempo la fase de la pata i está dada por:

$$\phi_i = (\phi_{i,0} + (f_0 + f_i)t) \pmod{2\pi}$$

donde $\phi_{i,0}$ es la fase inicial, f_0 es una frecuencia base común a todas las patas y f_i es la frecuencia de offset de la i -ésima pata. La fase indica etapa de contacto para $\phi_i \in [0, \pi)$ y movimiento para $\phi_i \in [\pi, 2\pi)$. Dada la fase ϕ_i , la posición objetivo de la pata será $r_i = F(\phi_i) + \Delta r_i$, donde Δr_i es la salida de la red neuronal, así como f_i , y la función F viene dada por:

$$F(\phi_i) = \begin{cases} (h(-2k^3 + 3k^2) - 0,5)^{H_{iz}} & k \in [0, 1] \\ (h(2k^3 - 9k^2 + 12k - 4) - 0,5)^{H_{iz}} & k \in [1, 2] \\ -0,5^{H_{iz}} & \text{sino} \end{cases}$$

donde $k = 2(\phi_i - \pi)/\pi$, h es un parámetro para la altura máxima del paso y H_i es un sistema de coordenadas de referencia colocado a una distancia del marco de coxa dada por la postura de parada de referencia, cuya orientación es siempre horizontal, con el eje x paralelo a la proyección ortogonal del eje x de la base del robot sobre el plano xy , como se puede ver en la Figura 4. De esta forma si $f_i = 0$ y $\Delta r_i = \vec{0}$ para todo i , el robot se mantiene caminando en el lugar.

Una vez obtenido los r_i en sus sistemas de coordenadas H_i , se convierte al sistema de coordenadas de coxa de cada pata y se usan las ecuaciones de cinemática inversa para obtener los valores de ángulos a usar como comandos de los controles de posición de cada articulación (los controladores PID ya están incluidos en el modelo URDF del robot PhantomX).

II-D. Currículum de Terrenos

Para el entrenamiento se utilizan dos tipos de terrenos que buscan cubrir variedad suficiente para permitir que la política aprendida pueda utilizarse en terrenos más generales. Éstos se describen por medio de un vector de parámetros $c_T = (c_{T,1}, c_{T,2}, c_{T,3})$.

- Terreno de colinas: basado en el ruido de Perlin, se genera a partir de 3 parámetros: rugosidad, frecuencia y amplitud del ruido. La altura de cada elemento del mapa de alturas se distribuye $\text{Perlin}(c_{T,2}, c_{T,3})[i, j] + \mathcal{U}(-c_{T,1}, c_{T,1})$.
- Terreno de bloques: consiste en bloques cuadrados de altura aleatoria. Todos los bloques tienen tamaño $c_{T,1} \times c_{T,1}$ con altura muestreada de $\mathcal{U}(0, c_{T,2})$.

Ambos terrenos se pueden apreciar en la Figura 1.

II-E. Filtro de Partículas

La generación de terrenos se realiza de forma adaptativa a través del uso de un filtro de partículas, siguiendo el enfoque presentado en [7]. En primer lugar veremos algunas definiciones necesarias.

La transitabilidad de un terreno generado mide la tasa de éxito al atravesarlo. La podemos utilizar para evaluar el progreso del aprendizaje de un agente. Se podría utilizar la función de recompensa, pero ésta métrica resulta ser mas intuitiva. En primer lugar, definimos la función v de etiquetado para la transición desde el estado s_t al estado s_{t+1} luego de aplicar la acción a_t :

$$v(s_t, a_t, s_{t+1}) = \begin{cases} 1 & \text{si } v_{pr}(s_{t+1}) > 0,2 \\ 0 & \text{si } v_{pr}(s_{t+1}) < 0,2 \vee \text{terminación} \end{cases}$$

donde $v_{pr}(s_{t+1})$ es el producto interno entre el vector velocidad de la base y la dirección del comando en el tiempo $t+1$, que indica la velocidad con la cual el agente logra seguir la velocidad indicada por el comando. Si la política π puede moverse en la dirección dada más rápido que 0,2 m/s, vamos a considerar al terreno transitable en esta dirección.

La transitabilidad de un terreno c_T bajo la política π se define como el siguiente valor esperado:

$$\text{Tr}(c_T, \pi) = \mathbb{E}_{\xi \sim \pi} \{v(s_t, a_t, s_{t+1} \mid c_T)\} \in [0.0, 1.0]$$

donde ξ son trayectorias generadas por π .

El objetivo del método de generación adaptativa de terrenos es obtener una distribución para muestrear parámetros c_T con transitabilidad en un rango medio $\text{Tr}(c_T, \pi) \in [0.5, 0.9]$. La razón es generar terrenos que no sean ni demasiado fáciles ni demasiado difíciles para el agente. Con esto en mente, dada una política π y un terreno c_T definimos la deseabilidad de un terreno $\text{Td}(c_T, \pi)$ como la probabilidad de que su transitabilidad esté en el rango deseado:

$$\begin{aligned} \text{Td}(c_T, \pi) &:= \Pr(\text{Tr}(c_T, \pi) \in [0.5, 0.9]) \\ &= \mathbb{E}_{\xi \sim \pi} \{\text{Tr}(c_T, \pi) \in [0.5, 0.9]\} \end{aligned}$$

Los terrenos con mayor probabilidad de tener una transitabilidad en el rango $[0.5, 0.9]$ serán los mas deseados.

Definimos la variable de medida y_j^k tal que $y_j^k = 1$ si $\text{Tr}(c_{T,j}^k, \pi) \in [0.5, 0.9]$. Podemos reescribir la deseabilidad del terreno en términos de la probabilidad de la variable de medida:

$$\Pr(y_j^k | c_{T,j}^k) = \Pr(\text{Tr}(c_{T,i}^k, \pi) \in [0.5, 0.9]) = \text{Td}(c_{T,j}^k, \pi)$$

En la implementación, dicha probabilidad de medida se computa reemplazando la esperanza matemática por un promedio de datos obtenidos durante el entrenamiento:

$$\Pr(y_j^k | c_{T,j}^k) \approx \sum_{N_{\text{traj}}} \frac{\chi_{[0.5, 0.9]}(\text{Tr}(c_{T,j}^k, \pi))}{N_{\text{traj}}}$$

en donde N_{traj} denota la cantidad de trayectorias obtenidas utilizando $c_{T,j}^k$ y χ es la función característica, que da 1 si $\text{Tr}(c_T, \pi) \in [0.5, 0.9]$ y 0 sino.

El filtro de partículas es un algoritmo utilizado para aproximar una distribución de probabilidad no paramétrica de forma iterativa. En este caso, nos interesa modelar la distribución de terrenos c_T tal que $\text{Tr}(c_T, \pi) \in [0.5, 0.9]$. Se comienza con un conjunto de terrenos, también llamados partículas, aleatorios e iterativamente se actualizan dichas partículas hasta converger a la distribución objetivo. Inicialmente, aproximamos la distribución con un conjunto finito de partículas, $c_T^k \in \mathcal{C}, k \in 1, \dots, N_{\text{particula}}$. Estas partículas se inicializan de manera uniforme.

El algoritmo se basa en las siguientes suposiciones:

- Los parámetros de terrenos con $\text{Tr}(\cdot, \pi)$ similares están cerca en Espacio Euclídeo.
- Una política π que fue entrenada en un área en $\mathcal{C}$ podrá interpolar a terrenos cercanos.

Basado en nuestras suposiciones, el filtro de partículas irá iterativamente incrementando la dificultad de los terrenos hasta converger a la distribución de c_T tal que $\text{Tr}(c_T, \pi) \in [0.5, 0.9]$ y nuestro agente se irá progresivamente adaptando a los nuevos terrenos generados asegurando un aprendizaje continuo.

II-F. Vector de Estado

El vector de estado es la entrada a la red neuronal, que se construye a partir del comando de velocidad, mediciones del simulador y variables internas del sistema. Se actualiza a 50Hz, que es la frecuencia de trabajo del sistema de control de alto nivel. Para el robot maestro implementado en este trabajo, siguiendo [7], el vector de estado está compuesto por:

- Dirección de velocidad lineal deseada $((\frac{B}{IB}\hat{v}_d)_{xy})$.
- Dirección de velocidad angular deseada $((\frac{B}{IB}\hat{\omega}_d)_z)$.
- Vector de gravedad.
- Velocidad angular de la base $(\frac{B}{IB}\omega)$.
- Velocidad lineal de la base $(\frac{B}{IB}v)$.
- Posición/velocidad de cada articulación $(\theta_i, \dot{\theta}_i)$.
- Fases para la generación de trayectorias $(\sin(\phi), \cos(\phi))$.
- Frecuencias para la generación de trayectorias (ϕ) .
- Frecuencia base (f_0) .
- Últimos dos valores de error de posición de cada articulación.
- Últimos dos valores de velocidad de cada articulación.

- Últimos dos valores de posición deseada de cada pata $((r_{i,d})_{t-1, t-2})$.
- Normal del terreno en cada pata.
- Altura de cada pata y de 8 puntos alrededor sobre un círculo de radio 10cm.
- Fuerzas de contacto de cada pata.
- Estados de contacto de cada pata.
- Estados de contacto de cada fémur.
- Estados de contacto de cada tibia.
- Coeficientes de fricción pata-suelo.

II-G. Función de Recompensa

La función de recompensa o refuerzo utilizada en este trabajo sigue lo propuesto en [7] y es:

$$R = 0,05r_{vl} + 0,05r_{va} + 0,05r_b + 0,01r_{ep} + 0,02r_c + 0,025r_s \quad (1)$$

donde los términos se explican a continuación, considerando que $(\frac{B}{IB}v)$ es la velocidad lineal de la base respecto del sistema global expresada en coordenadas de la base, $(\frac{B}{IB}\hat{v}_d)$ es el correspondiente comando de velocidad lineal normalizado, análogo para la velocidad lineal ω , y el subíndice xy indica la proyección ortogonal sobre este plano mientras que z indica la proyección ortogonal sobre esta dirección.

- Recompensa por velocidad lineal (r_{vl}):

$$r_{vl} = \begin{cases} e^{-\alpha_{vl}(v_p - 0,1)^2} & v_p < v_{threshold} \\ 1 & v_p \geq v_{threshold} \end{cases}$$

donde $v_p = (\frac{B}{IB}v)_{xy}(\frac{B}{IB}\hat{v}_d)_{xy}$ es la proyección ortogonal de la velocidad lineal de la base en xy sobre la dirección del comando de velocidad lineal.

- Recompensa por velocidad angular (r_{va}):

$$r_{va} = \begin{cases} e^{-\alpha_{va}(\omega_p - 0,1)^2} & \omega_p < \omega_{threshold} \\ 1 & \omega_p \geq \omega_{threshold} \end{cases}$$

donde $\omega_p = (\frac{B}{IB}\omega)_z(\frac{B}{IB}\hat{\omega}_d)_z$ es la proyección ortogonal de la velocidad angular de la base en z sobre la dirección del comando de velocidad angular.

- Recompensa por movimiento de la base (r_b):

$$r_b = e^{-\alpha_b v_0^2} + e^{-\alpha_b \|(\frac{B}{IB}\omega)_{xy}\|^2}$$

donde $v_0 = \|(\frac{B}{IB}v)_{xy} - v_p(\frac{B}{IB}\hat{v}_d)_{xy}\|$.

- Recompensa por elevación de patas (r_{ep}):

$$r_{ep} = \sum_{i \in I_{swing}} \chi_{\mathcal{F}_{clear}} / |I_{swing}|$$

donde I_{swing} es el conjunto de patas en movimiento (en inglés swing), χ es la función característica y $\mathcal{F}_{clear}$ es el conjunto de patas cuya posición es mayor o igual en altura que los 8 puntos medidos a su alrededor.

- Recompensa por colisión (r_c):

$$r_c = -|I_{c,cuerpo} \setminus I_{c,patas}|$$

es decir, sumamos las partes del robot que tienen una fuerza ejercida sobre ellas, exceptuando las patas.

- Recompensa por suavidad (r_s):

$$r_s = - \sum_i ||(r_{i,d})_t - 2(r_{i,d})_{t-1} + (r_{i,d})_{t-2}||$$

donde $(r_{i,d})_t$ es la posición deseada para la pata i -ésima en el instante t .

Los parámetros de la función de recompensa, incluyendo las constantes que acompañan cada término de la ecuación (1), se ajustaron manualmente para obtener mejores resultados. En función de los ensayos realizados, se definió usar $\alpha_{vl} = \alpha_{va} = 100$, $\alpha_b = 500$ y $v_{threshold} = \omega_{threshold} = 0,2$.

II-H. Entrenamiento mediante DRL y curriculum de terrenos

Para la red neuronal utilizamos la misma arquitectura propuesta en [7]. Sin embargo, el esquema de aprendizaje por refuerzo resultante no se entrenó utilizando el algoritmo Trust Region Policy Optimization (TRPO) [12] como sugiere el citado trabajo, sino que se utilizó el algoritmo Proximal Policy Optimization (PPO) [13]. PPO utiliza una aproximación de proximalidad para mejorar gradualmente la política sin hacer grandes cambios en cada paso de optimización mediante la aplicación de una restricción en la actualización de políticas para asegurar que no se desvíen demasiado de la política anterior. Por otro lado, TRPO utiliza un enfoque de región de confianza (*trust region*) para garantizar que las actualizaciones de políticas sean conservadoras y no introduzcan cambios drásticos que puedan desestabilizar el proceso de entrenamiento. Esto requiere calcular y resolver problemas de optimización más complejos y conlleva una mayor sensibilidad a los hiperparámetros. Por lo tanto, PPO tiende a ser más estable y computacionalmente menos costoso que TRPO, lo que lo hace que requiera tiempos de entrenamiento menor y resulte más escalable para entrenar en entornos de mayor tamaño y complejidad [13]. El pseudocódigo de entrenamiento completo, incluyendo el currículum de terrenos adaptativo, se muestra en el Algoritmo 1.

III. RESULTADOS

En primer lugar, se entrenó el sistema propuesto en terreno plano con comando de velocidad lineal estrictamente hacia adelante, es decir, sin giros. El resultado de dicho entrenamiento puede verse en la Figura 5, donde se muestra el refuerzo por paso de entrenamiento. Como material suplementario se presenta un video con un episodio de caminata con la política obtenida. La caminata resultante mantiene leves irregularidades en la forma de mover las patas, pero es aceptable y capaz de trasladarse en la dirección deseada.

Una vez alcanzada una política para caminar en plano, se procedió a probar con y sin el curriculum de terrenos adaptativo. En el primer caso se entrenó con terrenos aleatorios con parámetros uniformemente distribuidos en todo el rango y en el segundo caso con el Filtro de Partículas. En la Figura 6 se pueden ver ambas curvas de aprendizaje, donde el resultado para el Filtro de Partículas se finalizó ni bien contamos con información suficiente para realizar la comparación. Se puede observar que el resultado con curriculum de terrenos

Algoritmo 1 Entrenamiento con currículum automático de terrenos

Inicializar la memoria de replay, muestrear $N_{particle}$ $c_{T,0}$ s uniformemente de $\mathcal{C}$, $i, j = 0$.
while Todavía no hay convergencia **do**
 for $0 \leq k \leq N_{evaluate}$ **do**
 for $0 \leq l \leq N_{particle}$ **do**
 for $0 \leq m \leq N_{traj}$ **do**
 Generar un terreno usando $c_{T,j}^l$.
 Inicializar el robot.
 Correr la política π_i .
 Computar la traversibilidad para cada transición de estado.
 Guardar la trayectoria con sus valores de traversabilidad
 end for
 end for
 Actualizar la política con PPO.
 $i = i + 1$
 end for
 for $0 \leq l \leq N_{particle}$ **do**
 Computar la probabilidad de medición para cada parámetro $c_{T,j}^l$.
 end for
 for $0 \leq l \leq N_{particle}$ **do**
 Actualizar los pesos $\omega_j = \frac{P(y_i^l | c_{T,j}^l)}{\sum_m P(y_i^m | c_{T,j}^m)}$.
 end for
 Muestrear nuevamente $N_{particle}$ parámetros.
 Agregar los $c_{T,j}$ s a la memoria de replay.
 for $0 \leq l \leq N_{particle}$ **do**
 Muestrear de la memoria de replay con probabilidad *Preplay*.
 Mover $c_{T,j}^l$ a un valor adyacente en $\mathcal{C}$ con probabilidad $p_{transition}$.
 end for
 $j = j + 1$
end while

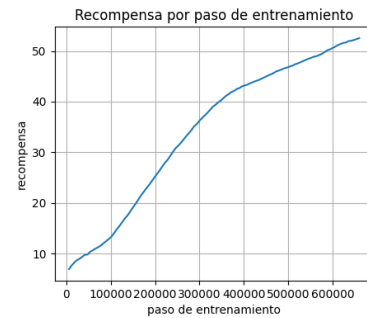


Figura 5: Curva de aprendizaje para terreno plano.

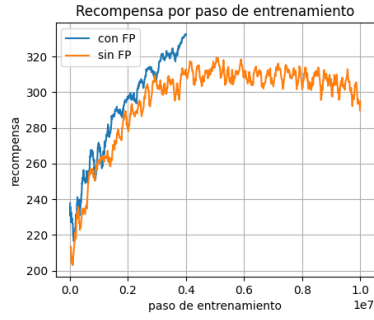


Figura 6: Curvas de aprendizaje para terreno plano utilizando el filtro de partículas para seleccionar los terrenos (celeste) y con terrenos aleatorios (azul).

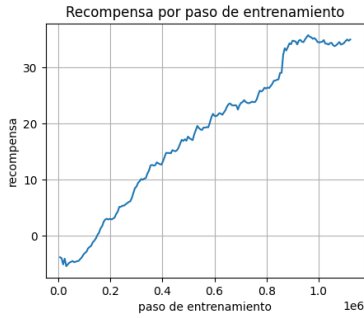


Figura 7: Curva de aprendizaje para terrenos irregulares.

adaptativo obtiene mayor recompensa final, y cualitativamente se observó que la política final obtenida es mejor.

Confirmada la utilidad del currículo de terrenos, se procedió a entrenar con éste al robot por un periodo de entrenamiento mayor. En la Figura 7 se puede ver la curva de aprendizaje obtenida. En el material suplementario se presentan videos con ejemplos de caminata con la política obtenida. Se observa que el robot logra aprender a caminar en los terrenos irregulares generados.

Tanto en terreno plano como en terrenos irregulares podemos observar que el robot alcanza una trayectoria subóptima, pero suficiente para poder transitar, con un movimiento de patas razonable, aunque no simétrico, ya que no hay nada en el esquema de aprendizaje por refuerzo ni en la función de recompensa utilizada que contemple la simetría del robot. Si observamos por varios segundos cómo el robot responde al comando de velocidad deseada, veremos que eventualmente se desvía de la dirección original, pero esto se debe a que el comando de velocidad se da a lazo abierto para el sistema global, es decir, ante una perturbación que corra al robot de su camino en una dirección éste continuará hacia adelante sin corregir que su postura se desvió de la orientación inicial. En un sistema que utilizara la política obtenida esto lo podría corregir el teleoperador o el sistema de planificación de trayectoria que sí tendrían disponible información de la posición global del robot.

IV. CONCLUSIONES Y TRABAJO FUTURO

En este trabajo se presentó un sistema de locomoción para robot hexápodo basado en aprendizaje por refuerzo profundo en simulación utilizando únicamente mediciones propioceptivas. El sistema obtenido pudo aprender a caminar en terrenos irregulares; la caminata final presenta ciertas limitaciones, pero logra transitar todo el terreno a pesar de ellas.

Una de las mayores dificultades encontradas durante la realización de este trabajo fue la demanda temporal del simulador para el entrenamiento. A pesar de las simplificaciones realizadas al sistema, el entrenamiento para un único comando de velocidad demanda alrededor de un día en una computadora con las siguientes prestaciones: Intel i7-5600U, 2.6 – 3.2GHz, dual-core 64-bit, 16GB de memoria RAM y una tarjeta gráfica NVIDIA GeForce RTX 3070 Ti. Esto se debe a que no se pudieron obtener velocidades de simulación mayores a real time en Gazebo Classic, y al requerirse miles de episodios para el entrenamiento el tiempo real escala a valores que impiden realizar pruebas eficientes, ajustar mejor los hiperparámetros y/o entrenar con más comandos de velocidad. Por lo tanto, una de las líneas de trabajo futuro consiste en explorar otras plataformas de simulación para el entrenamiento.

En cuanto al curriculum de terrenos adaptativo, se planea incorporar terrenos de tipo escalera, tal como en [7], así como ajustar más finamente los parámetros de los terrenos generados para obtener una variedad más acorde a los tipos de terrenos que se encontrará la política obtenida en su aplicación final. Por otro lado, como se mencionó previamente, la función de recompensa no contempla explícitamente la simetría del robot, ni tampoco la estabilidad estática y/o dinámica del mismo; estas también son cuestiones a abordar en un futuro.

El presente trabajo se enmarca en un objetivo más ambicioso que consiste en implementar un sistema de locomoción basado en aprendizaje maestro-estudiante en un robot hexápodo. Los resultados obtenidos presentan un primer paso en esta dirección. Una vez desarrollado el sistema completo maestro-estudiante, se evaluará la política obtenida con el estudiante en el robot real PhantomX Mark II.

V. AGRADECIMIENTOS

Este trabajo fue financiado parcialmente gracias al proyecto PIP 11220200101675CO otorgado por el Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET), Argentina.

REFERENCIAS

- [1] T. Li, N. Lambert, R. Calandra, F. Meier, and A. Rai, "Learning generalizable locomotion skills with hierarchical reinforcement learning," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 413–419.
- [2] W. Ouyang, H. Chi, J. Pang, W. Liang, and Q. Ren, "Adaptive locomotion control of a hexapod robot via bio-inspired learning," *Frontiers in Neurobotics*, vol. 15, p. 627157, 2021.
- [3] M. Schilling, K. Konen, F. W. Ohl, and T. Korthals, "Decentralized deep reinforcement learning for a distributed and adaptive locomotion controller of a hexapod robot," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 5335–5342.

- [4] T. Azayev and K. Zimmerman, “Blind hexapod locomotion in complex terrain with gait adaptation using deep reinforcement learning and classification,” *Journal of Intelligent & Robotic Systems*, vol. 99, no. 3, pp. 659–671, 2020.
- [5] A. Agarwal, A. Kumar, J. Malik, and D. Pathak, “Legged locomotion in challenging terrains using egocentric vision,” in *Conference on robot learning*. PMLR, 2023, pp. 403–415.
- [6] T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, “Learning robust perceptive locomotion for quadrupedal robots in the wild,” *Science Robotics*, vol. 7, no. 62, p. eabk2822, 2022.
- [7] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, “Learning quadrupedal locomotion over challenging terrain,” *Science robotics*, vol. 5, no. 47, p. eabc5986, 2020.
- [8] Gazebo. [Online]. Available: <https://gazebo-sim.org/>
- [9] Ros. [Online]. Available: <https://www.ros.org/>
- [10] Stable baselines 3. [Online]. Available: <https://stable-baselines3.readthedocs.io/en/master/>
- [11] Phantomx urdf. [Online]. Available: https://github.com/HumaRobotics/phantomx_gazebo.git
- [12] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, “Trust region policy optimization,” in *International conference on machine learning*. PMLR, 2015, pp. 1889–1897.
- [13] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” *arXiv preprint arXiv:1707.06347*, 2017.

Detección, clasificación y estimación de daños en hélices de vehículos multirrotor

Gabriel Torre^{1,2}, Claudio Pose^{1,3,4} y Juan Giribet^{1,4}

¹Laboratorio de Inteligencia Artificial y Robótica, Universidad de San Andrés

²Instituto de Ingeniería Biomédica, Facultad de Ingeniería, Universidad de Buenos Aires

³Laboratorio de Automática y Robótica, Facultad de Ingeniería, Universidad de Buenos Aires

⁴Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET)

Resumen—Este artículo presenta una arquitectura y metodología de entrenamiento para un procedimiento de aprendizaje automático, destinado a detectar, identificar y cuantificar daños en las hélices de vehículos aéreos no tripulados (UAV) multirrotor. Se recopilaban datos de vuelo de un vehículo multirrotor donde alguna de sus hélices fue reemplazada por una dañada, con tres tipos distintos de daños y de variada magnitud. Estos datos fueron luego utilizados para entrenar un modelo por partes, que incluye tanto clasificadores como redes neuronales, capaz de identificar con precisión el tipo de falla, estimar la gravedad del daño y localizar el rotor afectado. Los datos empleados para este análisis provienen exclusivamente de mediciones inerciales y comandos de control. Esta elección estratégica asegura la adaptabilidad de la metodología propuesta en diversas plataformas de vehículos multirrotor.

Index Terms—Vehículos aéreos no tripulados, detección de fallas, aprendizaje estadístico, redes neuronales.

A comparison between deep reinforcement learning methods applied to the control of a mobile robot in a Very Small Size Soccer environment

Natã Ismael Schmitt

Computer Science

Federal University of Santa Maria

Av. Roraima - Camobi, Santa Maria

email: nischmitt@inf.ufsm.br

Bruno Silva de Castro

Control and Automation Engineering

Federal University of Santa Maria

Av. Roraima - Camobi, Santa Maria

ORCID: 0009-0006-4696-435X

Anselmo Rafael Cukla

Control and Automation Engineering

Federal University of Santa Maria

Av. Roraima - Camobi, Santa Maria

email: anselmo.cukla@ufsm.br

Gabriel Tarnowski

Mechatronics Engineering department

Universidad Nacional de Misiones

Obera, Misiones, Argentina

email: gabriel.tarnowski@fio.unam.edu.ar

Leonardo R. Emmendorfer

Department of Electrical Energy Processing

Federal University of Santa Maria

Av. Roraima - Camobi, Santa Maria

email: leonardo.emmendorfer@ufsm.br

Thiago A. N. De Andrade

Department of Statistics

Federal University of Santa Maria

Av. Roraima - Camobi, Santa Maria

email: thiagoan.andrade@gmail.com

Abstract—Reinforcement Learning is a field of Artificial Intelligence that studies the development of algorithms that learn to make decisions optimally. In this research, we push the boundaries of knowledge by exploring deep learning techniques for robot control in soccer games. Our approach considers a controlled environment, with the IEEE Very Small Size Soccer robotic competition as a reference. The research's main contribution is comparing the Soft Actor-Critic (SAC) and Deep Deterministic Policy Gradient (DDPG) control algorithms evaluated in a virtual environment with dynamically altered game complexity. Specifically, those algorithms are used to model the actions of a player acting as an attacker in the robotic competition. Changes in the virtual environment consider the Curriculum Learning method, which enables a gradual and effective learning curve during training. The results underscore the SAC algorithm's superiority over DDPG in this specific context, highlighting its greater effectiveness and adaptability. From a broader perspective, the findings of this research contribute to a better understanding of how deep learning techniques can be optimized for applications in competitive robotics.

I. INTRODUCTION

Reinforcement Learning (RL) is probably the area of machine learning that attracts the most fascination among non-experts. This is because research in this area is often associated with robot training [1]. The peak popularization of this field of knowledge occurred in 2013 when the British company DeepMind built an algorithm capable of learning to play Atari games (<https://atari.com/>) without prior knowledge. It is a much broader field whose origins of study and research date back to the 50' [2].

In short, RL is the field of Artificial Intelligence that studies the development of algorithms that learn to make optimized decisions based on a process of trial and error under a mechanism of rewards and punishments. Through a virtual environment with controlled parameters, the algorithm (agent)

performs specific actions to achieve a specified objective. A reward function is used to assign a numerical value, which tells the agent whether its action contributed to or hindered the pursuit of the goal. In this way, the algorithm learns which actions, in which states, contribute to a positive reward, always seeking to maximize the gain of future rewards. Deep Reinforcement Learning (DRL) [3] is an essential variation of RL, which employs the use of deep neural networks to solve this problem.

Over the years, several works applying DRL techniques in different contexts have been published. A search on *Google Scholar* for the term *Deep Reinforcement Learning* returns more than a million articles and other academic contributions, highlighting the extent of research in this domain. One of the applications in emerging studies includes using DRL techniques to control robots in football matches within the scope of the *IEEE Very Small Size Soccer* (IEEE VSSS) competition.

IEEE VSSS robot football is a type of robotics competition that combines several technologies that cooperate so that the robots can score against the opponent, based on the rules of the Brazilian Robotics Competition (CBR) [4].

A game comprises two teams of 3 differential wheel robots, which are externally controlled. Its dimensions are a maximum of 8 cm wide, 8 cm long, and 8 cm high. The arena is 1.5 meters long and 1.3 meters wide. An orange golf ball with well-defined weight and characteristics is used for the match. The VSSS is widely used as a source of research, bringing scientific advances to robotics and bringing together groups that aim to achieve the best robots and control methods. In Brazil, the main attraction event is the *Latin American Robotics Competition* (LARC), with the VSSS and many other types of competitions taking place annually.

The control of the robots is done via a computer, which, connected to the camera (positioned above the field), monitors the movements of these agents in a match. The captured image is then processed with computer vision algorithms, extracting variables such as the ball's position and direction and speed of all robots on the field, among others. This information is used by the game control algorithm, which will make decisions based on the current state. Differential wheel robots can be controlled through the speeds of each of their wheels so that the algorithm can generate the final speed of the wheels of each robot and, consequently, control its movements.

This research aims to use DRL techniques to study robot control in football matches within the scope of the IEEE VSSS competition. More specifically, training *Soft Actor-Critic* (SAC) [5] and *Deep Deterministic Policy Gradient* (DDPG) [6] control algorithms, which were evaluated in a virtual environment with dynamically changing game complexity. For this, the open source simulator *RSoccer* [7] was used through its implementation as a library of the *Python* programming language.

In addition to this introduction, this paper is structured as outlined below. Section II lists some essential works related to the research topic. Section III presents methodological details that guided the achievement of the results of this research, including theoretical aspects about reinforcement learning, a computational framework centered on the open source simulator *RSoccer* [7], the definition of learning premises and rewards, among others. Detailed analysis of the scenarios under investigation is presented in Section IV. Section V addresses final considerations and future proposals.

II. RELATED WORKS

From both theoretical and applied perspectives, researchers have dedicated themselves to advancing the field of DRL. In this Section, mention is made of some exponent works, emphasizing those that have direct links with the findings presented in this study or are closely related to the lines of investigation currently under development. The main objective of the connections is to provide the fundamental theoretical framework for the scope of this research and to anticipate future directions within the scope of DRL applied to robotic control.

The research carried out by [8] presents a comprehensive overview of the use of RL in robotic manipulation tasks, mapping existing knowledge gaps and suggesting future directions for investigation. The authors undertook a systematic review, employing a broad sweep of scientific literature, focusing on electronic databases recognized for disseminating essential results, such as *IEEE Xplore* and *Google Scholar*. In this process, 42 articles were selected and analyzed, published since 2013, based on specific inclusion parameters.

In a similar perspective, [9] perform a comprehensive literature review, covering publications in the period 2015 and 2022, to understand and apply DRL algorithms. These authors direct their focus to significant advances in the use of RL in robotics, highlighting the challenges linked to the

transition from strategies learned in simulations to real-world applications. This study included the analysis of more than 150 academic works, using research platforms such as *Google Scholar*, *IEEE Xplore*, and *ArXiv*.

In a seminal paper, [10] introduced the reinforcement learning methodology known as *Curriculum Learning*. This approach, which prioritizes progressive training in steps of increasing complexity, is inspired by the effective learning processes observed in humans and animals. Our research uses the contribution of [10] to adapt the training environment in the scenarios considered.

Recently, in [11], they conducted a study in which two types of neural networks are compared to control the formation of robots in a VSSS environment considering the DDPG methods and *Double Deep Q-Network* (DDQN) [12]. The function of neural networks is to introduce a controller that acts as a type of technician who coordinates the role of the robots, be it an attacker, defender, or goalkeeper during the match.

The work presented by [13] explores the application of two neural network algorithms, specifically DDPG and SAC, to move differential robots in a virtual environment containing obstacles. The main focus of this study is to analyze the robot's movement on the playing field, where the objective is for it to move between specific points on the map without colliding with obstacles. Our investigation is heavily inspired by the contributions of [13], similarly exploring the effectiveness of these algorithms in the specific context of robotic navigation.

In addition to the previously mentioned articles, we would like to highlight other research at the frontier of knowledge in robotic computing. We refer to the works of [14], [15], [16], [17], [18] and [19]. These studies span a wide variety of subareas in robotics research, including multi-robot trajectory planning, path planning, and smoothing, as well as decentralized and probabilistic methods for avoiding collisions between multiple robots. They also explore using reinforcement learning in football competitions with small and very small robots. This work and its future developments have benefited significantly from the contributions of these researchers.

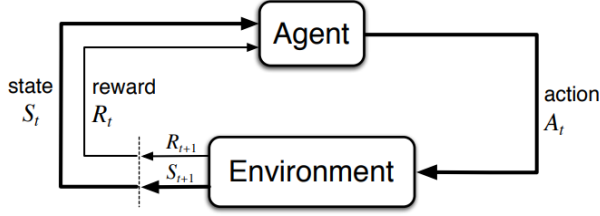
III. METHODOLOGY

A. Reinforcement Learning

This work follows the data flow generally applied in Reinforcement Learning, as shown in Figure 1 [20]. Here, the environment and the agent are presented. The agent receives initial data from the environment, such as state S_t and R_t , and generates an action A_t , which is executed in the environment, generating new states and rewards S_{t+1} and R_{t+1} , which are processed again by the agent, completing the cycle.

This process is aligned with the Markov Decision Process (MDP) properties, where each decision is based only on the current state, thus respecting Markov properties. In a simplified way, the MDP can be represented as $MDP(S, A, P, R)$, where S represents the set of states of the environment, A is the set of actions available to the agent, P is the transition probability function and R is the reward function ([21], [22] and [1]).

Figure 1. Execution of Reinforcement Learning algorithms



B. rSoccer Overview

To carry out the Reinforcement Learning tests used, the library `rSoccer` [7] was used, an environment with simulated physics aimed at training VSSS games, based in the library `GYM` [23], widely used in the literature to evaluate the performance of reinforcement learning algorithms. The library is available in the `Python` programming language, and its environment was modified to allow the player to be trained with progressive difficulties and personalized rewards. Furthermore, the `rSoccer` library supports the rendering of images for real-time visualization of environments through a GUI (Graphical User Interface), simplifying the observation of game progress in the environment and allowing an empirical evaluation of the behavior of agents.

The VSSS environment includes an out-of-the-box reward function that evaluates changes between the previous and current state of the environment and assigns a numerical value to indicate the importance of the agent's action in achieving the goal. This reward comprises three functions: a ball potential function, a movement function, and an energy expenditure function. The ball potential function compares the previous and current position of the ball to determine whether it has approached the enemy goal, returning a positive value proportional to the ball's displacement in the direction of the goal and calculated by Equation 1. The movement function provides feedback on the agent's action, considering the agent's approach to the ball, encouraging the robot's initial learning, and is defined by Equation 2. Lastly, the reward is given to agent's energy expenditure to avoid unnecessary movements and discard non-optimal paths to reach the ball. This reward is given by the Equation 3.

$$R_p = \frac{bp_t - bp_{t-dt}}{dt} \quad (1)$$

$$\vec{RB} = ||(B_x - R_x, B_y - R_y)|| \quad (2)$$

$$E_r = - \sum_{p=0}^E (|w_0| + |w_1|) \quad (3)$$

With the three functions described, the final reward function is achieved by including weights for each function so the user can balance and contribute more appropriate weights to the algorithms. In Equation 4 W_b , W_m and W_e are introduced, representing weights for ball potential, agent movement, and

energy spent, respectively. Note that if the goal indication for the reward function is given by adding or subtracting G_p , which is normally a higher value, the algorithm learns the importance of the goals.

$$R = \begin{cases} W_b \cdot R_p + W_m \cdot M_r + W_e \cdot E_r + G_p & \text{Goal} \\ W_b \cdot R_p + W_m \cdot M_r + W_e \cdot E_r & \text{Didn't goal} \\ W_b \cdot R_p + W_m \cdot M_r + W_e \cdot E_r - G_p & \text{Opponent goal} \end{cases} \quad (4)$$

C. Attacker Simulation Ambient

This subsection covers the theoretical aspects that makeup one of this research's main contributions: the modification of the learning environment, emphasizing progressive training and stages of increasing complexity.

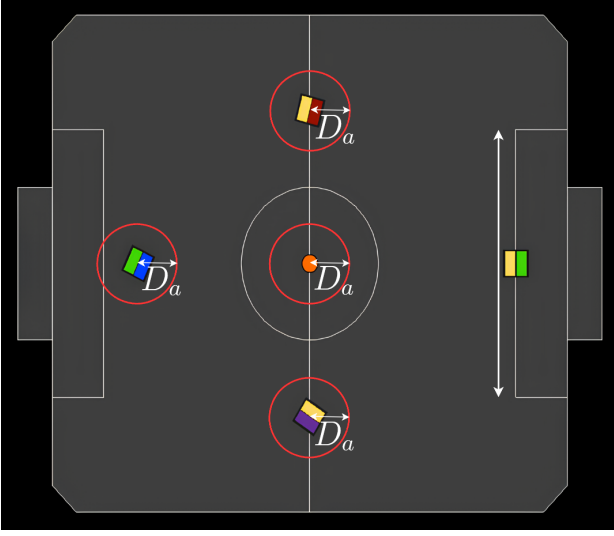
Aiming to take advantage of Curriculum Learning techniques [10], discrete environments were replaced by a single environment, which will have its difficulty dynamically changed, allowing the agent smooth training, as its learning curve will dictate the difficulty of the environment. Therefore, we adopted a number ranging between 0 and 1 for the difficulty criterion, with 0 being the least complexity and 1 being the most significant difficulty. A difficulty of 0 means the environment with the robots in a fixed starting position and immobile. Based on the average rewards from the previous 100 episodes, we define the current difficulty of the environment to calculate D_a , whose effect is described in Figure 2. From this, Equation 5 describes the calculation of D_a , representing the difficulty of the environment, and which values D_a admits. In this equation, $r_m = \frac{\max_r R}{\lambda}$, R is the set of averages among the agent's last 100 rewards, which we select the maximum value of R for use in this equation. λ is an arbitrary number representing the reward function's maximum value, which was empirically defined as 450 from previous training. Note that in 5, the minimum value of D_a is 0.1.

$$D_a = \begin{cases} 0.1 & r_m \leq 0.6 \\ 0.25 & 0.6 < r_m \leq 0.7 \\ 0.55 & 0.7 < r_m \leq 0.8 \\ 1 & r_m > 0.8 \end{cases} \quad (5)$$

D. Agent Observations

As inspiration from the work of [13], the standard `rSoccer` observations in this striker training environment have been replaced by those described in Table I. Observations that previously reported absolute coordinates of x and y now report distance and angle as a way of representing the observed positions always in relation to the agent so that the neural network has data that occurs more frequently because it is centered on the agent's position, and no longer globally in the field, thus facilitating the generalization of the network. It was also decided to reduce the amount of data reported about other robots, such as linear and angular velocities, since this information is not of great relevance to the robot's objective, in addition to being proof that few observations are needed for the agent satisfies the requirements of the environment. A visual

Figure 2. Dynamicity of the attacker's environment



description for a better understanding of the observations can be seen in Figure 3.

Observations
Distance between agent and ball
Agent to ball angle
Distance between agent and opponent goal
Agent angle to opponent goal
Distance between agent and M_i
Agent angle for M_i
Sine of the agent angle
Cosine of the agent's angle
Linear velocity in X of the ball
Linear velocity in Y of the ball
Linear speed in X of the agent
Linear velocity in Y of the agent
Agent angular velocity

Table I

OBSERVATIONS FOR THE ATTACKING AGENT USED IN THIS WORK

E. Hyperparameters

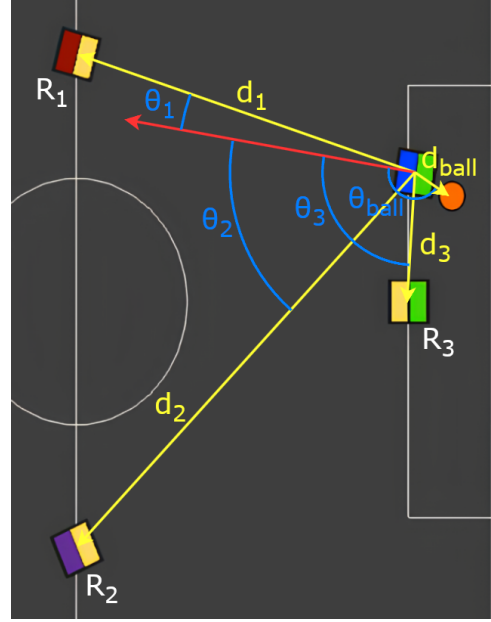
The hyperparameter scheme adopted in this research was based on the work of [13]. The authors made the source codes available in a repository on GitHub, facilitating their application in different contexts by other researchers. Although the main focus of the study is not robotic programming, its contributions were precious for developing the programming environment presented in this work. Table II below presents the hyperparameters considered in this work.

Hyperparameter	Description	Used Value
α	Learning rate	2.5e-5
Batch size	Number of samples per training batch	256
Experience memory	Replay buffer size	5e5
σ	Noise level	0.1
γ	Future discount factor	0.99
τ	Soft update rate for target networks	0.999

Table II

SOFT ACTOR CRITIC HYPERPARAMETERS

Figure 3. Agent Notes



F. Reward Function

This work considers a modification of the reward function to improve agent learning. The first change was in the weights so that the agent better prioritizes the occurrence of goals. An increase in the weight of the ball gradient was also made to make the robot learn more quickly that the ball must follow a particular path. The energy was increased to make it more obvious to the agent when choosing a shorter route. The weights used for training are found in Table III, whose values were explained previously.

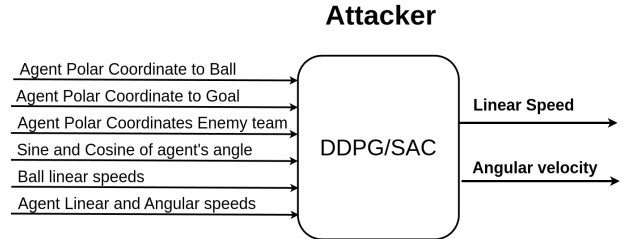
Observation	Quantity
W_b	2
W_m	0.25
W_e	1e-3
G_p	500

Table III

WEIGHTS USED IN THE REWARD FUNCTION OF THE ATTACKING ENVIRONMENT

In Figure 4, we see the inputs and outputs for the attacking agent, which come from the data presented in Table I. Each of the inputs is part of the state S of the environment at time t , and their outputs make up the action A at time t .

Figure 4. Attacking agent input and output diagram



IV. RESULTS AND DISCUSSIONS

In this section, the numerical evaluation results are presented for the study that compared the performance of two DRL algorithms, namely SAC and DDPG, applied to the control of a mobile robot in the context of IEEE VSSS.

It is observed that SAC surpassed DDPG in terms of obtaining rewards and effectiveness in carrying out the proposed tasks. As can be seen in Figures 5 and 6, comparing the rewards obtained by the SAC and DDPG methods, respectively.

The graph in Figure 5 highlights four regions with different colors, each with a different value for D_a , representing training steps with different difficulty levels. Such complexity levels are determined as described in Equation 5. Generally, it can be inferred that as D_a increases, the performance metric stabilizes, suggesting that a larger D_a could be associated with a more stable or effective learning policy in the long run.

Similarly, the image in Figure 6 shows the performance obtained by the DDPG model, considering different learning stages. Contrary to what was found for the SAC algorithm, performance is highly affected, decreasing as complexity increases. In the graphs presented in both Figure 5 and Figure 6, the X axis indicates the training steps. In contrast, the Y axis represents the Simple Moving Average (SMA) calculated from the last 100 episodes.

Figure 5. SAC algorithm training graph

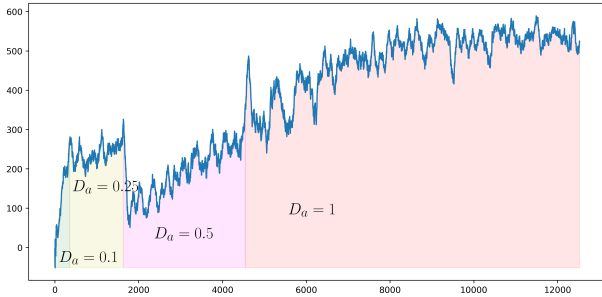
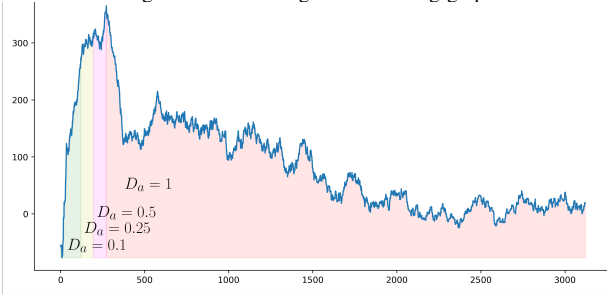


Figure 6. DDPG algorithm training graph



The experimental results showed that SAC achieved significantly superior performance in the attacker control task. Not only was SAC able to score more effectively against virtual opponents, but it also showed greater adaptability and generalization ability to different game configurations and obstacles. On the other hand, DDPG, although it showed good

initial performance, faced difficulties adapting to incremental changes in the complexity of the environment, resulting in less consistent performance.

The superiority of SAC can be attributed to its ability to explore the space of actions more efficiently and states thanks to its maximum entropy policy. This allows for a more comprehensive exploration of the game environment, leading to more robust and adaptive strategies. Furthermore, the SAC framework, which emphasizes reward-based learning and error minimization, has proven to be particularly suitable for the dynamic and unpredictable environment of IEEE VSSS.

The results suggest that applying *Curriculum Learning* techniques may have played a crucial role in improving the learning process, facilitating the progressive adaptation of robots to increasingly complex challenges. This hypothesis, however, requires a more detailed investigation for confirmation. This simulates a more natural and effective learning environment, where agents master simple tasks before progressing to more difficult ones.

The results also suggest the importance of a well-defined reward function in reinforcement learning. The functionality of dynamically adjusting the environment's difficulty and reward based on the agent's performance proved to be fundamental in encouraging the robot to learn effective control and navigation strategies.

V. CONCLUSION

This paper applies Deep Reinforcement Learning (DRL) algorithms to control robots in the *IEEE Very Small Size Soccer* (IEEE VSSS) robot soccer modality. Specifically, *Soft Actor-Critic* (SAC) and *Deep Deterministic Policy Gradient* (DDPG) control algorithms were considered and modified to act as attackers, with performances evaluated in a virtual environment with dynamically changed game complexity. The learning mechanism takes into account *Curriculum Learning* [10] methodology. This methodology emphasizes progressive training in stages of increasing complexity, taking inspiration from the efficient learning processes observed in humans and animals. In summary, the main findings of the research are listed below:

- DRL algorithms demonstrated effectiveness in controlling robots within the scope of the IEEE VSSS competition.
- As an attacker, the SAC algorithm not only outperformed DDPG, but it did so with a level of efficiency that demands respect, as evidenced by its superior accumulated rewards.
- The DDPG algorithm exhibited a learning curve that began with progressive improvement, followed by a decline as the complexity of the environment increased. In contrast, despite requiring a more extended training period, the SAC algorithm demonstrated superior performance under environmental conditions of varying complexity.
- The superiority of SAC can be primarily attributed to its maximum entropy policy, which promotes a more effective exploration of the action space.

- Considering that the primary success metric defined was the ability to score goals in all episodes with the fewest possible attempts, it was found that none of the algorithms achieved absolute success. Therefore, the need for future algorithm improvements is identified to achieve this goal.

The results listed above suggest a broad spectrum of future contributions. Among the main lines of investigation underway by the proponents of this research, the following stand out:

- Explore specific algorithms to improve goalkeeper training in robotics, considering different reinforcement learning techniques.
- Expand the scope of performance comparison between different algorithms, such as SAC and DDPG, to include new approaches, focusing on their capabilities as attacking players.
- Investigate *path planning* algorithms that integrate reinforcement learning knowledge for better real-time navigation between obstacles, fulfilling the objectives of robot football.
- Deepen the study of the maximum entropy policy used by SAC to understand how this approach can be explored or improved in different competitive robotics configurations.

REFERENCES

- [1] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, Second. The MIT Press, 2018. [Online]. Available: <http://incompleteideas.net/book/the-book-2nd.html>.
- [2] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and Tensorflow: Concepts, Tools, and Techniques to Build Intelligent Systems*, 3rd. O'Reilly Media, Nov. 2022, p. 861, ISBN: 978-1098125974.
- [3] V. Mnih, K. Kavukcuoglu, D. Silver, *et al.*, *Playing atari with deep reinforcement learning*, 2013. arXiv: 1312.5602 [cs.LG].
- [4] *Very small size soccer rules*, <https://www.cbrobotica.org/wp-content/uploads/2021/05/vssRules3x321.pdf>, Acesso: 18/12/2023.
- [5] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, *Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor*, 2018. arXiv: 1801.01290 [cs.LG].
- [6] T. P. Lillicrap, J. J. Hunt, A. Pritzel, *et al.*, *Continuous control with deep reinforcement learning*, 2019. arXiv: 1509.02971 [cs.LG].
- [7] F. B. Martins, M. G. Machado, H. F. Bassani, P. H. M. Braga, and E. S. Barros, “Rsoccer: A framework for studying reinforcement learning in small and very small size robot soccer”, in *RoboCup 2021: Robot World Cup XXIV*, R. Alami, J. Biswas, M. Cakmak, and O. Obst, Eds., Cham: Springer International Publishing, 2022, pp. 165–176, ISBN: 978-3-030-98682-7.
- [8] H. Nguyen and H. La, “Review of deep reinforcement learning for robot manipulation”, in *2019 Third IEEE International Conference on Robotic Computing (IRC)*, 2019, pp. 590–595. DOI: 10.1109/IRC.2019.00120.
- [9] D. Han, B. Mulyana, V. Stankovic, and S. Cheng, “A survey on deep reinforcement learning algorithms for robotic manipulation”, *Sensors*, vol. 23, no. 7, p. 3762, 2023. DOI: 10.3390/s23073762.
- [10] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, “Curriculum learning”, in *Proceedings of the 26th Annual International Conference on Machine Learning*, ser. ICML '09, Montreal, Quebec, Canada: Association for Computing Machinery, 2009, pp. 41–48, ISBN: 9781605585161. DOI: 10.1145/1553374.1553380. [Online]. Available: <https://doi.org/10.1145/1553374.1553380>.
- [11] C. H. C. Pena, M. G. Machado, M. S. Barros, *et al.*, “An analysis of reinforcement learning applied to coach task in ieee very small size soccer”, in *2020 Latin American Robotics Symposium (LARS), 2020 Brazilian Symposium on Robotics (SBR) and 2020 Workshop on Robotics in Education (WRE)*, 2020, pp. 1–6. DOI: 10.1109/LARS/SBR/WRE51543.2020.9307069.
- [12] M.-W. Han and P. Kopacek, “Neural networks for the control of soccer robots”, in *ISIE'2000. Proceedings of the 2000 IEEE International Symposium on Industrial Electronics (Cat. No.00TH8543)*, vol. 2, 2000, 571–575 vol.2. DOI: 10.1109/ISIE.2000.930360.
- [13] J. C. de Jesus, V. A. Kich, A. H. Kolling, R. B. Grando, M. A. d. S. L. Cuadros, and D. F. T. Gamarra, “Soft actor-critic for navigation of mobile robots”, in *Journal of Intelligent & Robotic Systems*, vol. 102, 2021. DOI: 10.1007/s10846-021-01367-5.
- [14] A. Zolanvari, M. Shirazi, and M. Menhaj, “A q-learning approach for controlling a robotic goalkeeper during penalty procedure”, in *2019 II International Congress on Science and Engineering*, Hamburg-Germany, 2019, pp. 1–12.
- [15] H. Ali, G. Xiong, H. Wu, B. Hu, Z. Shen, and H. Bai, “Multi-robot path planning and trajectory smoothing”, in *2020 IEEE 16th International Conference on Automation Science and Engineering (CASE)*, IEEE, 2020, pp. 685–690.
- [16] W. Luo, W. Sun, and A. Kapoor, “Multi-robot collision avoidance under uncertainty with probabilistic safety barrier certificates”, in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 372–383.
- [17] H. Zhu, B. Brito, and J. Alonso-Mora, “Decentralized probabilistic multi-robot collision avoidance using buffered uncertainty-aware voronoi cells”, *Autonomous Robots*, vol. 46, pp. 401–420, 2022. DOI: 10.1007/s10514-021-10029-2.
- [18] F. Martins, M. Machado, H. Bassani, P. Braga, and E. Barros, “Rsoccer: A framework for studying reinforcement learning in small and very small size robot soccer”, in *RoboCup 2021: Robot World Cup XXIV*,

- R. Alami, J. Biswas, M. Cakmak, and O. Obst, Eds., ser. Lecture Notes in Computer Science, vol. 13132, Cham: Springer, 2022. DOI: 10.1007/978-3-030-98682-7_14.
- [19] L. da Silva Costa and F. Tonidandel, “Multi-robot path planning with safety based control applied to the small size league robots”, in *RoboCup 2023: Robot World Cup XXVI*, C. Buche, A. Rossi, M. Simões, and U. Visser, Eds., ser. Lecture Notes in Computer Science, vol. 14140, Cham: Springer, 2024. DOI: 10.1007/978-3-031-55015-7_7.
 - [20] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: A Bradford Book, 2018, ISBN: 0262039249.
 - [21] M. Ororbia and G. Warn, “Structural design synthesis through a sequential decision process”, in *Proceedings of the ASME 2020 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, ser. 40th Computers and Information in Engineering Conference (CIE), Available: <https://doi.org/10.1115/DETC2020-22647>, ASME, Virtual, Online, 2020, V009T09A045. DOI: 10.1115/DETC2020-22647.
 - [22] R. BELLMAN, “A markovian decision process”, *Journal of Mathematics and Mechanics*, vol. 6, no. 5, pp. 679–684, 1957, ISSN: 00959057, 19435274. [Online]. Available: <http://www.jstor.org/stable/24900506> (visited on 05/15/2024).
 - [23] G. Brockman, V. Cheung, L. Pettersson, *et al.*, *Openai gym*, 2016. arXiv: 1606.01540 [cs.LG].

Sesión 8: Robótica Industrial y Manipuladores

Análisis de posición de eslabones de robot colaborativo

Estudio de las posiciones de los eslabones de un robot colaborativo mediante matrices de transformación homogénea

Resumen—El presente artículo tiene como objetivo determinar las posibles posiciones que pueden ocupar los eslabones de un robot colaborativo sin que se produzca una colisión entre ellos. Para ello se emplea una combinación de matrices de transformación homogéneas, ejecutadas en un script de MatLab.

Palabras Clave—trayectoria; robot; eslabón; posición; matriz de transformación homogénea

I. INTRODUCCIÓN

Con el objetivo de seleccionar la mejor combinación de longitudes de eslabones en el diseño de un robot colaborativo, se llevó a cabo un análisis de trayectoria, en el que se tienen en cuenta las posibles dimensiones y su influencia en la amplitud de alcance del robot. Es decir que la opción seleccionada será aquella que posibilite la mayor cantidad de puntos para el actuador final¹.

Para ello, es necesario realizar un análisis de trayectoria, en el cual se pueda identificar numéricamente la posición del efector final del robot, tomando como entrada la longitud de los eslabones, y el rango de giro de cada eje. Este estudio se basa en una herramienta matemática muy utilizada en la robótica: la matriz de transformación homogénea.

II. MATRIZ DE TRANSFORMACIÓN HOMOGÉNEA

Esta matriz contiene la información numérica necesaria para ubicar un sistema de ejes de coordenadas (en posición y orientación de sus ejes), partiendo desde otro sistema ubicado en algún punto del espacio [1].

En ella se representan los movimientos de rotación y traslación que se llevan a cabo en los 3 ejes.

Siendo M la matriz de transformación homogénea, de dimensión 4×4 , se compone de una parte que define la rotación del eje de coordenadas, y otra que afecta a la traslación.

Además, es posible concatenar distintos movimientos, representados por diferentes matrices, multiplicándolas y formando una nueva matriz que representa la totalidad de los movimientos.

Rotación alrededor del eje Z

Traslación en el eje Y

$$M = \begin{bmatrix} \cos \theta & -\sin \theta & 0 & 0 \\ \sin \theta & \cos \theta & 0 & \Delta_y \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Fig. 1. Explicación de las partes de la matriz de transformación homogénea.

A continuación, unos ejemplos de movimientos de rotación y traslación pura de sistemas de ejes de coordenadas, representados en sus respectivas matrices y de forma gráfica:

Ejemplo de matriz de transformación homogénea para un caso de traslación pura sobre el eje x:

$$M = \begin{bmatrix} 1 & 0 & 0 & \Delta_x \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Fig. 2. Representación del movimiento de traslación pura.

Ejemplo de matriz de transformación homogénea para un caso de rotación pura alrededor del eje x:

$$M = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \alpha & -\sin \alpha & 0 \\ 0 & \sin \alpha & \cos \alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Fig. 3. Representación del movimiento de rotación pura.

¹ Elemento ubicado en el extremo del último eslabón del robot en el que usualmente se acoplan herramientas.

III. APLICACIÓN DE LA MATRIZ EN EL ANÁLISIS DE TRAYECTORIA

En este caso, se tienen 6 sistemas de coordenadas, uno por eje. Por lo que también se tienen 6 matrices de transformación homogénea. Cada una de estas contiene el ángulo de rotación y traslación de cada eje. Es importante aclarar que, para este estudio, la distancia de traslación de cada eje es constante, ya que con ella se representa la longitud de cada eslabón.

Los valores de rotación y traslación de cada eje están expresados en función del sistema de coordenadas del eje anterior. Sin embargo, al multiplicar las matrices consecutivamente y en orden creciente, se puede obtener la posición de cualquier eje con respecto al primer sistema de coordenadas. De esta manera es posible conocer la posición de cada articulación según la longitud de los eslabones y los ángulos de giro.

En el siguiente esquema, se observa una simplificación de los eslabones de un robot, con los movimientos que puede realizar cada eje. En esta configuración, los tipos de movimiento de cada uno son:

- Eje 1: Rotación.
- Eje 2: Doblado.
- Eje 3: Doblado.
- Eje 4: Rotación.
- Eje 5: Doblado.

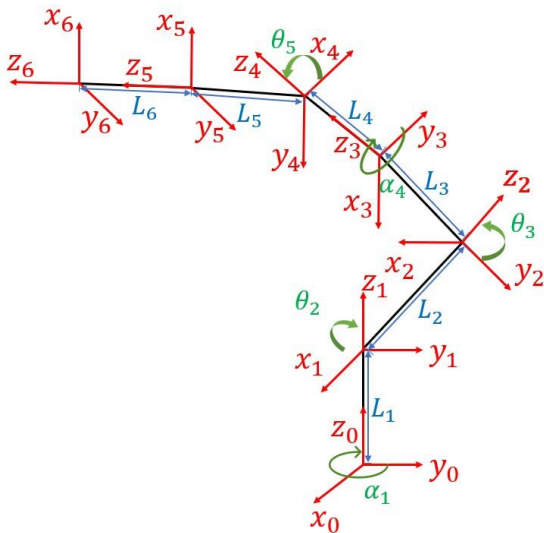


Fig. 4. Esquema simplificado del robot con sus sistemas de coordenadas.

Por lo tanto, las variables de entrada son:

- L_1 = Longitud del eslabón 1
- L_2 = Longitud del eslabón 2
- L_3 = Longitud del eslabón 3
- L_4 = Longitud del eslabón 4

- L_5 = Longitud del eslabón 5
- L_6 = Longitud del efector final
- α_1 = Ángulo de giro de la articulación 1
- θ_2 = Ángulo de giro de la articulación 2
- θ_3 = Ángulo de giro de la articulación 3
- α_4 = Ángulo de giro de la articulación 4
- θ_5 = Ángulo de giro de la articulación 5

Y la variable de salida:

- P_6 = Posición del extremo del efector final

Cabe aclarar que, si bien se pueden conocer las posiciones de todas las articulaciones, la única de interés es la posición del extremo del efector final.

Siguiendo esta lógica se realizó un script en Matlab, cuyo objetivo es poder conocer todas las posiciones posibles del efector final, en un rango de rotación de cada eje:

- Eje 1: de -180° a 180°
- Eje 2: de -140° a 140°
- Eje 3: de -140° a 140°
- Eje 4: de -180° a 180°
- Eje 5: de -140° a 140°

Este script tiene la capacidad de generar todas las alternativas de ángulos de giros posibles para cada articulación (con un paso determinado, por ejemplo: de -180° a 180° , con un incremento de 30° por vez).

Entonces, se ingresan los valores de las longitudes a analizar, y con las combinaciones de ángulos, se llenan las matrices de transformación homogénea, arrojando la posición del efector final para cada combinación. Sin embargo, no todos los puntos son válidos, ya que numerosas posiciones de los eslabones no son coherentes con el comportamiento real del robot, por lo que es necesario analizar cada combinación según las restricciones colocadas.

Las restricciones se establecieron de manera tal que los eslabones no colisionen entre ellos. Por lo tanto, se descartan todas las combinaciones que generen una proximidad inferior a 10 milímetros entre los diferentes eslabones, asegurando que no existirán contactos ni choques entre ellos.

Además, se descartan todas las combinaciones de ángulos que generen posiciones de cualquier eje en $-Z$, ya que esto en la realidad significaría una colisión con el suelo donde está apoyado el robot.

Finalmente, los valores que cumplan con las restricciones establecidas se contabilizan y se grafican.

El objetivo de este análisis es determinar qué combinación de longitudes posibilita la mayor cantidad de posiciones del efector final, es decir, la que otorga mayor rango de movimiento.

En el siguiente gráfico, se observan las posiciones que puede adoptar el efector final del robot colaborativo a analizar, siendo cada una de ellas un punto en el espacio.

Se acotó el gráfico al primer cuadrante, con el fin de facilitar la visualización.

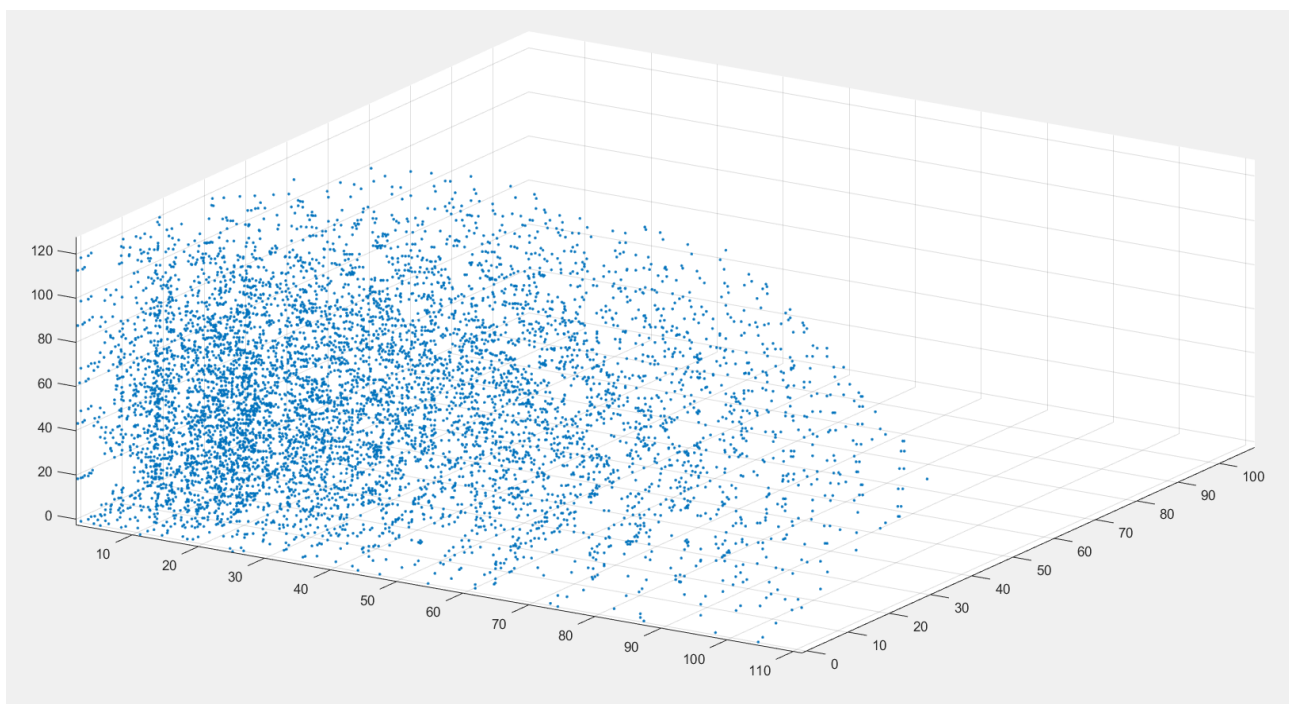


Fig. 5. Ejemplo de posiciones del actuador final.

IV. CÓDIGO

Cabe aclarar que, si bien se pueden conocer las posiciones de todas las articulaciones, la única de interés es la posición del extremo del efector final.

```
%definicion de longitudes%
long1=20;
long2=25;
long3=25;
long4=25;
long5=25;
long6=10;

%declaracion de variables%
a12=0;
a23=0;
a34=0;
a45=0;
a56=0;

%combinaciones de angulos%
paso = 30;
matAng=combvec(-180:paso:180,-
180:paso:180,-180:paso:180,-
180:paso:180,-180:paso:180);

%definicion de variables para el
bucle%
sz=size(matAng);
numfil=sz(1,2);
x=zeros(1,numfil);
y=zeros(1,numfil);
z=zeros(1,numfil);

e = 1;
i = 1;
for n=1:numfil
    %definicion de angulos%
    a12=matAng(1,n)*pi/180;
    a23=matAng(2,n)*pi/180;
    a34=matAng(3,n)*pi/180;
    a45=matAng(4,n)*pi/180;
    a56=matAng(5,n)*pi/180;

    %matrices homogeneas%
    mh1=[cos(a12) -sin(a12) 0
0;sin(a12) cos(a12) 0 0;0 0 1 long1; 0
0 0 1];
    mh2=[1 0 0 0; 0 cos(a23) -sin(a23)
long2*(-sin(a23));0 sin(a23) cos(a23)
long2*cos(a23);0 0 0 1];
    mh3=[1 0 0 0; 0 cos(a34) -sin(a34)
long3*(-sin(a34));0 sin(a34) cos(a34)
long3*cos(a34);0 0 0 1];
    mh4=[cos(a45) -sin(a45) 0
0;sin(a45) cos(a45) 0 0;0 0 1 long4; 0
0 0 1];
    mh5=[1 0 0 0; 0 cos(a56) -sin(a56)
long5*(-sin(a56));0 sin(a56) cos(a56)
long5*cos(a56);0 0 0 1];
    mh6=[1 0 0 0;0 1 0 0;0 0 1 long6;
0 0 0 1];
```

```
pos1=mh1;
pos2=mh1*mh2;
pos3=mh1*mh2*mh3;
pos4=mh1*mh2*mh3*mh4;
pos5=mh1*mh2*mh3*mh4*mh5;
mfin=mh1*mh2*mh3*mh4*mh5*mh6;

%Armado de la matriz con posiciones del
efector final%

x(1,n)=mfin(1,4);
y(1,n)=mfin(2,4);
z(1,n)=mfin(3,4);

%Eliminacion de las posiciones en las
que se intersectan eslabones%

P0 = [0,0,0];
P1 = [pos1(1,4),pos1(2,4),pos1(3,4)];
P2 = [pos2(1,4),pos2(2,4),pos2(3,4)];
P3 = [pos3(1,4),pos3(2,4),pos3(3,4)];
P4 = [pos4(1,4),pos4(2,4),pos4(3,4)];
P5 = [pos5(1,4),pos5(2,4),pos5(3,4)];
P6 = [mfin(1,4),mfin(2,4),mfin(3,4)];

cant_pint=20;

pint1=zeros(cant_pint,3);
pint2=zeros(cant_pint,3);
pint3=zeros(cant_pint,3);
pint4=zeros(cant_pint,3);
pint5=zeros(cant_pint,3);
pint6=zeros(cant_pint,3);

for i=1:cant_pint
    t=i/(cant_pint+1);
    pint1(i,:)=(1-t)*P0+t*P1;
    pint2(i,:)=(1-t)*P1+t*P2;
    pint3(i,:)=(1-t)*P2+t*P3;
    pint4(i,:)=(1-t)*P3+t*P4;
    pint5(i,:)=(1-t)*P4+t*P5;
    pint6(i,:)=(1-t)*P5+t*P6;
end

if pos2(3,4)>0 && pos3(3,4)>0 &&
pos4(3,4)>0 && pos5(3,4)>0 && mfin(3,4)>0
    for g = 1:length(pint1)
        for s = 1:length(pint3)
            d13=norm(pint1(g,:)-
pint3(s,:));
            d14=norm(pint1(g,:)-
pint4(s,:));
            d15=norm(pint1(g,:)-
pint5(s,:));
            d16=norm(pint1(g,:)-
pint6(s,:));
            if or(d13<e,d14<e)
                x(1,n) = 0;
                y(1,n) = 0;
                z(1,n) = 0;
            end
            if or(d15<e,d16<e)
                x(1,n) = 0;
                y(1,n) = 0;
                z(1,n) = 0;
            end
        end
    end
end
```

```

        for a = 1:length(pint2)
            for o = 1:length(pint4)
                d24=norm(pint2(a,:)-
pint4(o,:));
                d25=norm(pint2(a,:)-
pint5(o,:));
                d26=norm(pint2(a,:)-
pint6(o,:));
                if (d24<e||d25<e)
                    x(1,n) = 0;
                    y(1,n) = 0;
                    z(1,n) = 0;
                end
                if d26<e
                    x(1,n) = 0;
                    y(1,n) = 0;
                    z(1,n) = 0;
                end
            end
        end
        for u = 1:length(pint3)
            for r = 1:length(pint5)
                d35=norm(pint3(u,:)-
pint5(r,:));
                d36=norm(pint3(u,:)-
pint6(r,:));
                if (d35<e || d36<e)
                    x(1,n) = 0;
                    y(1,n) = 0;
                    z(1,n) = 0;
                end
            end
        end
        for w = 1:length(pint4)
            for q = 1:length(pint6)
                d46=norm(pint4(w,:)-
pint6(q,:));
                if d46<e
                    x(1,n) = 0;
                    y(1,n) = 0;
                    z(1,n) = 0;
                end
            end
        end
    end
end
end

plot3(x,y,z, '. ')
axis([inf inf inf inf 0 inf])
hold on
grid on

```

V. CONCLUSIONES

El código desarrollado ofrece una herramienta eficaz para determinar la mejor combinación de longitudes de eslabones para un robot colaborativo, considerando las posibles posiciones del efector final.

Sin embargo, cabe destacar que este no es el único criterio que debe tenerse en cuenta para la elección de las longitudes, por lo que se sugiere ponderar los resultados obtenidos del análisis con otros factores intervinientes.

Además, con modificaciones menores en el código, se puede llevar a cabo un análisis comparativo entre robots colaborativos con distinta cantidad de eslabones, o con una combinación diferente de movimientos de rotación y doblado.

No obstante, sin conocimientos básicos de programación puede resultar desafiante, por lo que se propone como mejora a futuro la modificación del código, para que adaptarlo a los diferentes casos resulte más accesible.

REFERENCIAS

- [1] A. Barrientos, L. P. Peñín, C. Balaguer, y R. Aracil, "Fundamentos de Robótica", 2da ed., Universidad Politécnica de Madrid, 1997.

Teleoperación de un Manipulador Dual Móvil con Torso

Fernando A. Chicaiza

Instituto de Automática

Universidad Nacional de San Juan

San Juan, Argentina

fachicaiza@inaut.unsj.edu.ar

Emanuel Slawiński

Instituto de Automática

Universidad Nacional de San Juan

San Juan, Argentina

slawinski@inaut.unsj.edu.ar

Vicente Mut

Instituto de Automática

Universidad Nacional de San Juan

San Juan, Argentina

vmut@inaut.unsj.edu.ar

Abstract—La teleoperación de sistemas robóticos ofrece una forma de mejorar las capacidades humanas de manipulación y navegación en entornos peligrosos o inaccesibles. Los manipuladores móviles, que combinan la manipulación de un brazo robótico con la movilidad de un robot móvil, enfrentan desafíos al realizar tareas que requieren manipulación simultánea en múltiples puntos. Integrar un manipulador adicional puede abordar este desafío, mejorando la versatilidad del robot para realizar diversas tareas. Para controlar estos manipuladores, se pueden diseñar enfoques que los consideren como una cadena cerrada o que los controlen de manera independiente. Este trabajo propone un tipo de mapeo para controlar un robot seguidor complementario en una configuración de teleoperación bilateral. Se ejecuta un conjunto de experimentos donde un operador humano controla ambos dispositivos hápticos, generando referencias para controlar los efectores finales del manipulador móvil dual con torso. La implementación se realiza con el robot manipulador móvil dual simulador en Gazebo-ROS2, mientras se considera dos dispositivos hápticos Falcon-Novint y operadores humanos en el lazo de control.

Index Terms—Teleoperación, manipulador dual móvil, retardos de tiempo.

Actualización integral basada en Arduino y ROS de un manipulador robótico Scorb什么 ER-VII

León Herrera

Laboratorio de Robótica y

Sistemas Embebidos

Departamento de Computación

Universidad de Buenos Aires, CABA

Email: lherrera@dc.uba.ar

Facundo Pessag

Laboratorio de Robótica y

Sistemas Embebidos

Departamento de Computación

Universidad de Buenos Aires, CABA

Email: fpessag@dc.uba.ar

Pablo De Cristóforis

Laboratorio de Robótica y

Sistemas Embebidos

Departamento de Computación

Universidad de Buenos Aires, CABA

Email: pdecris@dc.uba.ar

Resumen—En este trabajo presentamos el conjunto de acciones llevadas a cabo en el manipulador robótico Scorb什么 ER-VII para reemplazar tanto el hardware como el software original del mismo. Esta actualización integral está basada en microcontroladores Arduino y ROS (Robot Operating System). También se presenta un modelo cinemático para el control del brazo y un modelo en formato URDF obtenido a partir de Xacro que permiten la visualización en Rviz y la simulación en Gazebo. El código del proyecto fue publicado en: <https://github.com/lrse/scorb什么-vii> para beneficio de la comunidad científica.

I. INTRODUCCIÓN

Los manipuladores robóticos han ganado gran aceptación en la industria gracias a su capacidad de llevar a cabo un número variado de tareas de manera repetitiva con rapidez y precisión, y la posibilidad de adaptarse a diversos entornos de trabajo. En todo manipulador se pueden distinguir cuatro componentes principales que lo conforman: partes electromecánicas, hardware, firmware de bajo nivel y software de alto nivel. Es la suma de ellos, lo que hace que los manipuladores puedan operar como un sistema autónomo, versátil, y robusto para desarrollar múltiples propósitos.

Los manipuladores robóticos Scorb什么 fueron lanzados al mercado durante la década '90 por la empresa Eshed Robotec, llamada ahora Intelinek Inc. [6]. Estos robots fueron adquiridos en la Argentina por varias universidades para tareas de docencia e investigación. Hoy en día tanto el hardware como el software de control de estos manipuladores se encuentran completamente desactualizados respecto de las prestaciones que ofrecen los sistemas actuales. Sin embargo, las partes electromecánicas siguen funcionando correctamente en la mayoría de los casos, con lo cual tiene sentido plantearse la reingeniería de los mismos para volver a ponerlos operativos a bajo costo.

En este trabajo presentamos el conjunto de acciones que fueron llevadas a cabo para actualizar de manera integral el hardware y software de un Scorb什么 ER-VII, a partir del uso de placas Arduino [9] y ROS (Robot Operating System) [4]. La adopción de estas tecnologías permitió la incorporación de un entorno de simulación del manipulador en Gazebo [7] y visualización del estado en Rviz, a partir de un modelo hecho en Xacro [12] que luego se pasa a formato URDF.

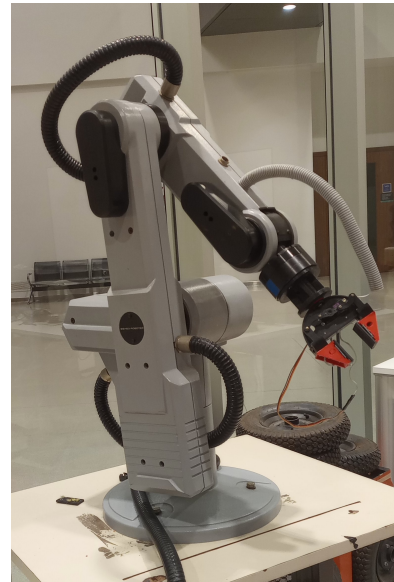


Figura 1. Foto del Scorb什么 ER-VII con el efector final original reemplazado por una pinza paralela actuada por un servo-motor.

El Scorb什么 ER-VII tiene 4 articulaciones aparentes (base, hombro, codo y muñeca) y 1 no aparente (la rotación en la muñeca) que le confieren sus grados de libertad. Como efector final posee originalmente una pinza neumática de apertura paralela. Cada articulación es actuada por un motor de corriente continua. La transmisión mecánica entre el giro del motor y la rotación de cada articulación se realiza mediante un sistema de polea con correa dentada y una caja reductora. Cada uno de los motores tiene acoplado a su eje un encoder óptico incremental. Sobre cada articulación se encuentran los micro-switches que definen los finales de carrera y un home-switch para cada articulación. En la Figura 1 podemos ver una foto del Scorb什么 ER-VII.

A continuación presentamos una explicación detallada de las modificaciones realizadas en cada uno de los componentes del manipulador: Hardware, Firmware, Software. Luego presentamos el modelo cinemático y por último el entorno de visualización y simulación en RViz y Gazebo respectivamente.

II. HARDWARE

El hardware que controla el movimiento del brazo fue reemplazado por un conjunto de microcontroladores Arduino [9] conectados en una placa de prototipado (ver Figura 2). Se siguen utilizando las fuentes de energía originales para alimentar tanto a las placas Arduinos como a los motores. Por otro lado, el efector final original fue reemplazado por una pinza servo-motor a la cual se le incorporaron dos sensores de tacto capacitivos.

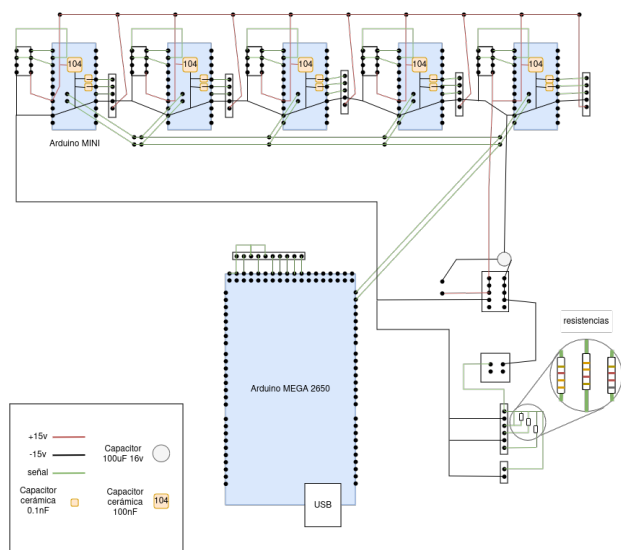


Figura 2. Diagrama del circuito de control del SCORBOT ER-VII renovado. Solo se incluyen las conexiones en la placa de prototipado.

Específicamente, se utilizaron 5 Arduino Pro MINIs, 1 Arduino MEGA 2560, 1 Arduino UNO, 2 sensores de tacto capacitivos TTP223B y 5 puentes H Arduino IBT-2 para reemplazar el hardware de control original del manipulador.

En la Figura 2 podemos ver un diagrama de la placa de prototipado con los Arduinos Pro MINIs y el Arduino MEGA 2560. Cada Arduino Pro MINI lee la señal del encoder correspondiente a cada una de las 5 articulaciones del manipulador a través de un conector de 5 pines y envía una señal PWM al puente H que controla el movimiento del motor de corriente continua mediante un conector de 8 pines. Como se puede observar en la Figura 2, la comunicación entre el Arduino MEGA y los MINIs se hace por un canal de dos pines y el protocolo de comunicación I^2C . En la Figura 3 también podemos ver como está hecha la conexión entre los Arduinos MINIs y sus respectivos puente-H y encoders de la articulación correspondiente.

La pinza neumática original fue reemplazada por una que se acciona mediante un servo-motor, a la cual se le incorporaron sensores de tacto capacitivos para la detección de objetos en el agarre. Para esto, diseñamos dos piezas en TinkerCad [10] y Ultimaker Cura [11] de manera de poder acomodar los sensores de tacto por detrás de una lamina de goma que sirve como material sensible al entrar en contacto con un objeto en el momento del agarre. El modelo 3D de las piezas puede

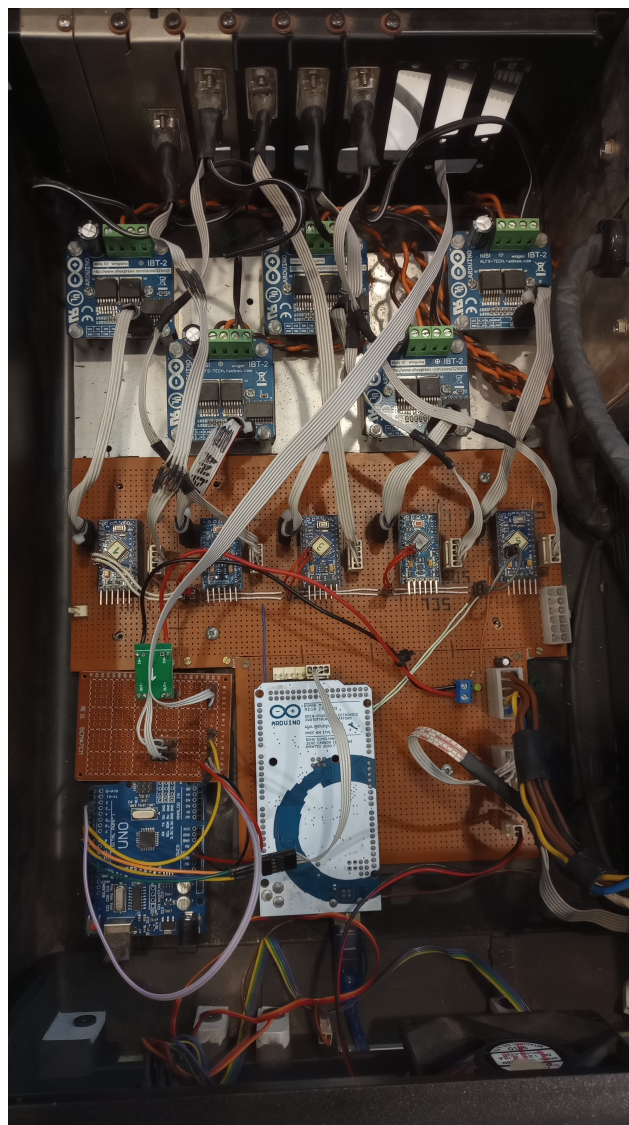


Figura 3. Foto de las placas Arduinos y puentes H que reemplazan al hardware original de control del Scrobot ER-VII. Pueden observarse las conexiones entre los Arduino Pro MINIs, los puentes H, las fuentes de energía y los conectores que van al manipulador.

observarse en la Figura 4b, donde se ven los peldaños que ayudan a separar el sensor de la lamina de goma y el agujero por donde salen los pines del sensor a la sección de atrás para conectarse a los cables de alimentación y señal. En la Figura 4a se puede ver la nueva pinza montada en el Scrobot ER-VII como efector final.

Los sensores de tacto y el servo-motor que actúa la pinza están conectados a la fuente de baja tensión y al Arduino UNO, que lee la señal de los sensores y envía la señal al servo-motor para controlar el ángulo de apertura de la pinza (ver Figura 3). El Arduino UNO se conecta a 4 pines al Arduino MEGA quien a su vez le indica cuando cerrar y abrir la pinza y reporta cuando sensa que un objeto fue agarrado. El Arduino UNO utiliza uno de sus pines de interrupción para cada sensor y otro para pasarle un ángulo de apertura al servo-motor. El Arduino

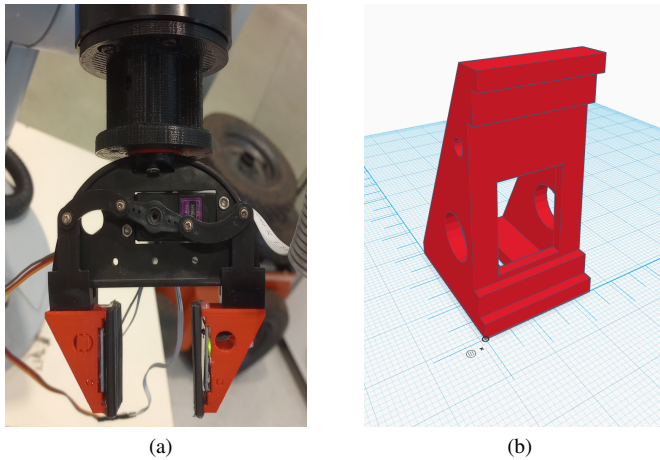


Figura 4. A la izquierda (a): la nueva pinza montada al SCORBOT ER-VII con los dos sensores capacitivos. A la derecha (b): el modelo 3D de la pieza diseñada para montar los sensores y unas laminas de goma que ayudan a sensar

MEGA se conecta a través de un cable USB con una PC que corre ROS y un nodo ROSserial para poder recibir comandos y reportar el estado de de cada articulación del manipulador.

III. FIRMWARE

En el Arduino UNO se ejecuta una maquina de estados simple que mantiene el estado de la pinza y dependiendo del mismo comanda el cierre o apertura, al llegarle la orden correspondientes del Arduino MEGA, a quien a su vez le reporta su estado cada vez que ocurre una variación. Los estados de la pinza son: abierta, cerrada, abriendo, cerrando, esperando señal de sensor de tacto izquierdo, esperando señal del sensor de tacto derecho, agarrando objeto.

Cuando se desea agarrar un objeto con la pinza, el Arduino UNO espera recibir la señal de alguno de los dos sensores de tacto y cuando esto ocurre, cambia a otro estado donde sigue cerrando mientras espera la señal contacto del otro sensor. Cuando ésta llega, asume que terminó de agarrar un objeto completamente y la pinza se cierra 5 grados más para luego parar y terminal el comportamiento de cierre.

En cada Arduino Pro MINI se ejecuta el código que se encarga de crear una señal PWM a partir de un control PID a lazo cerrado con las lecturas de los encoders de la articulación correspondiente. Además, cada Arduino Pro MINI debe comunicarse con el Arduino MEGA para informar el ángulo de la articulación que controla.

El Arduino MEGA ejecuta una biblioteca de ROS para Arduino que le permite recibir y publicar mensajes por su puerto serial estableciendo una conexión con el nodo *ROS-Serial* de la computadora que controla el manipulador. El código que se ejecuta en el Arduino MEGA informa cada determinado tiempo los ángulos de cada articulación del brazo y se comunica con el Arduino MINI correspondiente al recibir un mensaje para mover alguna articulación o con el Arduino UNO para cerrar o abrir la pinza. En caso de interactuar con el Arduino UNO, se usan 4 pines para la comunicación: uno

para saber si la pinza está abierta o cerrada, otro para saber si está agarrando un objeto o no, un tercero para setear si se quiere abrir o cerrar la pinza, y el último para ordenar cuando tiene que cerrarse o abrirse la pinza.

Para recibir y enviar mensajes entre el Arduino MEGA y los Arduinos MINIs, se usa una comunicación de dos pines y el protocolo *I²C*. Se utilizan varias funciones para cada interacción posible entre el Arduino MEGA y cada Arduino MINI, entre las que vale la pena mencionar:

- Definir una velocidad para una articulación durante un determinado lapso de tiempo.
- Definir un ángulo deseado para una articulación.
- Leer el valor del encoder de la articulación.
- Setear los parámetros del Arduino MINI (entre ellos los coeficientes del controlador PID)

Para la comunicación entre el Arduino MEGA y la computadora que se realiza por el puerto serie se definen los siguientes mensajes:

- Velocidad y dirección de movimiento para las articulaciones.
- Parada total de todos movimientos del brazo.
- Abrir o cerrar la pinza.
- Mover cada articulación a un ángulo especificado.
- Mover cada articulación a una posición de inicio (*home*).

IV. SOFTWARE DE ALTO NIVEL

En la computadora que controla el Scorbobot ER-VII se ejecuta un sistema que implementa un conjunto de nodos de ROS con los que se controla el brazo real y/o se realiza la simulación en Gazebo. En la figura 5 se pueden observar los nodos de ROS activos durante la ejecución del sistema.

Como puede observarse, para leer los comandos de entrada se utilizan los nodos *ros_joy* [14] y *ros_keyboard* [15] para el joystick y el teclado respectivamente. Luego se utiliza el nodo *universal_teleop* [16] que traduce los comandos a *eventos* y *controles* para el nodo *scorbobot*, el cual a su vez comunica estos eventos al brazo a través de nodo *Scorbobot_serial*.

Los eventos del nodo *universal_teleop* se usan para comunicar el input de los botones del joystick y las teclas del teclado, y se implementan mediante un *string* detallando la acción a realizar y un valor booleano que nos indica si hay que finalizar o iniciar la acción. Las acciones que están definidas hasta ahora son: activar el modo seguro del manipulador (que setea una velocidad baja para todas las articulaciones), ir a una posición inicial (*home*), cerrar o abrir la pinza, mover hacia un lado u otro el codo, muñeca o rotación del brazo o parar en seco todos los movimientos. Un control es como un evento pero en lugar de un valor booleano tiene un valor entero asociado, esto se usa para comunicar el input de los controles analógicos del joystick y en nuestra configuración del joystick lo usamos para mover las articulaciones de la base y el hombro del brazo. El nodo *scorbobot* toma estos eventos y controles y los interpreta para publicar en varios *topics* de

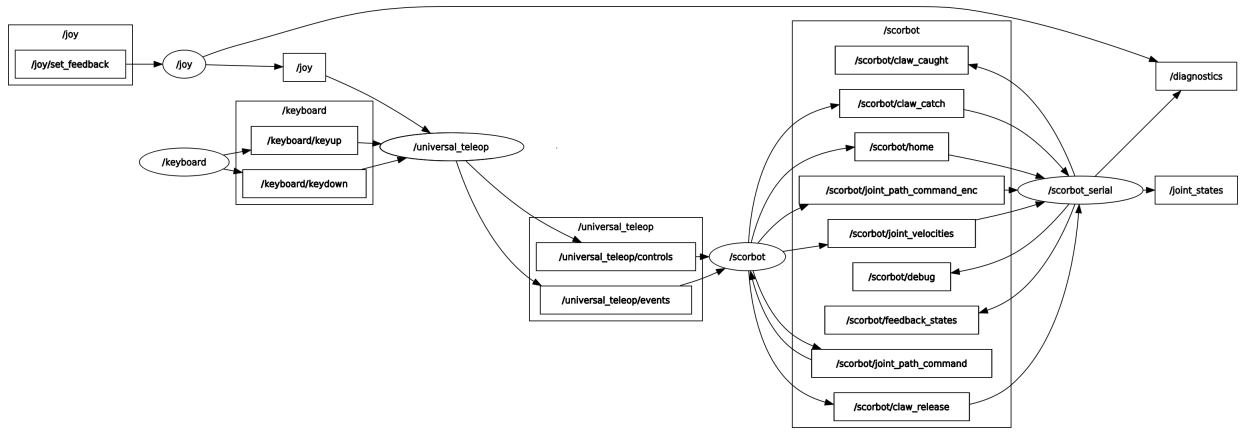


Figura 5. Esquema de los nodos de ROS activos al ejecutar el sistema de control del SCORBOT ER-VII en una PC.

manera que el nodo *ROSserial* pueda leerlos, entre los que podemos mencionar:

- */claw_catch* y */claw_release* se usan para cerrar y abrir la pinza respectivamente,
- */home* se usan para ir a la posición donde se inició el SCORBOT,
- *joint_path_command_enc* se usa para comunicar una posición deseada para cada articulación del brazo,
- *joint_path_command* es el conjunto de posiciones que se le comunican al nodo *scorbot* para pasarle al Arduino MEGA,
- */joint_velocities* es un arreglo de velocidades para cada motor del brazo,
- */claw_caught* se usa para comunicar si la pinza agarró un objeto,
- */debug* se usa para mensajes de control,
- *feedback_states* comunica cuando se llega a la posición deseada anterior a la que se debía mover el brazo,
- *joint_states* reporta el ángulo actual en radianes de las articulaciones.

V. MODELO CINEMÁTICO

El modelo cinemático que desarrollamos para el SCORBOT ER-VII está basado en los modelos presentados por Predescu y Predescu [11], y por Predescu y Stroe [12]. A este modelo se le hicieron algunas modificaciones para que funcionaran mejor con la simulación de Gazebo [13]:

- Usamos como origen de coordenadas la base del Scorbót en vez de su primer articulación.
- Usamos rotaciones en sentido horario en vez de anti-horario.
- No utilizamos un sistema paralelo para la rotación de la 5ta articulación (en la Figura 6 la rotación θ_5 , el *roll* del efector), si no que esa rotación se agrega a la última matriz de transformaciones del sistema.

Parametrizamos el brazo de la forma vista en la Figura 6, donde θ_i es la rotación de la articulación i , y a_i y d_i son la medida de la sección i del robot. De esta forma conseguimos el modelo que vemos en la figura 7 y la siguiente tabla de

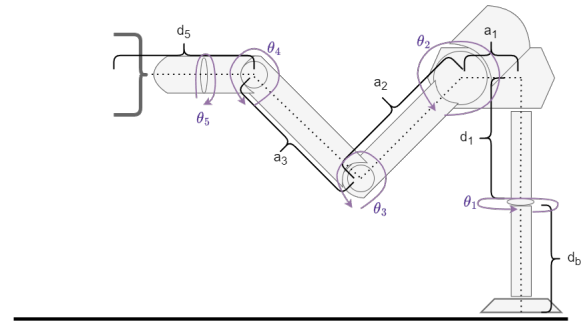


Figura 6. Parámetros y medidas del Scorbót ER-VII usadas para calcular la cinemática directa.

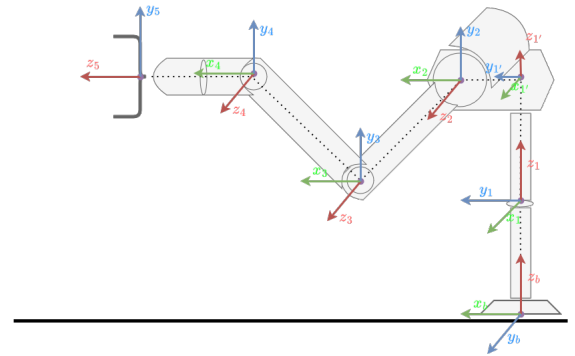


Figura 7. Modelo cinemático para el Scorbót ER-VII usado en este proyecto

parámetros Denavit Hartenberg (D-H), donde θ es la rotación en el eje z , β es la rotación del eje y y γ la rotación del eje x .

Luego, definimos las siguientes abreviaciones:

- $c_k = \cos(\sum_{i \in k} \theta_i)$ donde $k \subseteq \{1, 2, 3, 4, 5\}$
- $s_k = \sin(\sum_{i \in k} \theta_i)$ donde $k \subseteq \{1, 2, 3, 4, 5\}$
- $c_{\frac{\pi}{2}} = \cos(\frac{\pi}{2}) = 0$
- $s_{\frac{\pi}{2}} = \sin(\frac{\pi}{2}) = 1$

A partir de este modelo podemos calcular la cinemática directa con las matrices de rotación y traslación correspondientes:

Parámetros de D-H					
Sección i	$d_i(mm)$	$a_i(mm)$	$\theta_i(rad)$	$\beta_i(rad)$	$\gamma_i(rad)$
b	200	0	0	0	0
1	17	0	θ_1	0	0
1'	0	43	0	$\frac{\pi}{2}$	$\frac{\pi}{2}$
2	0	300	θ_2	0	0
3	0	250	θ_3	0	0
4	0	0	θ_4	$\frac{\pi}{2}$	0
5	190	0	θ_5	0	0

Cuadro I

TABLA DE PARÁMETROS DENAVIT HARTENBERG (D-H) DEL MANIPULADOR SCORBOT ER-VII.

T_1^b : rotación $\frac{\pi}{2}$ en z + negamos x para invertirlo + traslación d_b en z

$$T_1^b = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_b \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_1^b = \begin{bmatrix} 0 & -1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & db \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$T_{1'}^1$: rotación θ_1 en z + traslación d_1 en z

$$T_{1'}^1 = \begin{bmatrix} c_1 & -s_1 & 0 & 0 \\ s_1 & c_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_{1'}^1 = \begin{bmatrix} c_1 & -s_1 & 0 & 0 \\ s_1 & c_1 & 0 & 0 \\ 0 & 0 & 1 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$T_2^{1'}$: rotación $\frac{\pi}{2}$ en x + rotación $\frac{\pi}{2}$ en y + traslación a_1 en x

$$T_2^{1'} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_2^{1'} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & a_1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

T_3^2 : rotación θ_2 en z + traslación a_2 en x

$$T_3^2 = \begin{bmatrix} c_2 & -s_2 & 0 & 0 \\ s_2 & c_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_3^2 = \begin{bmatrix} c_2 & -s_2 & 0 & a_2 c_2 \\ s_2 & c_2 & 0 & a_2 s_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

T_4^3 : rotación θ_3 en z + traslación a_3 en x

$$T_4^3 = \begin{bmatrix} c_3 & -s_3 & 0 & 0 \\ s_3 & c_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_3 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_4^3 = \begin{bmatrix} c_3 & -s_3 & 0 & a_3 c_3 \\ s_3 & c_3 & 0 & a_3 s_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

T_5^4 : rotación θ_4 en z + rotación $\frac{\pi}{2}$ en y

$$T_5^4 = \begin{bmatrix} c_4 & -s_4 & 0 & 0 \\ s_4 & c_4 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_{\frac{\pi}{2}} & 0 & s_{\frac{\pi}{2}} & 0 \\ 0 & 1 & 0 & 0 \\ -s_{\frac{\pi}{2}} & 0 & c_{\frac{\pi}{2}} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_5^4 = \begin{bmatrix} 0 & -s_4 & c_4 & 0 \\ 0 & c_4 & s_4 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

T_e^5 : rotación θ_5 en z + traslación d_5 en z

$$T_e^5 = \begin{bmatrix} c_5 & -s_5 & 0 & 0 \\ s_5 & c_5 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_5 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_e^5 = \begin{bmatrix} c_5 & -s_5 & 0 & 0 \\ s_5 & c_5 & 0 & 0 \\ 0 & 0 & 1 & d_5 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Con éstas matrices podemos calcular la cinemática directa del manipulador:

$$T_e^b = T_1^b \times T_{1'}^1 \times T_2^{1'} \times T_3^2 \times T_4^3 \times T_5^4 \times T_e^5$$

REFERENCIAS

$$T_e^b = \begin{bmatrix} s_{234}c_1s_5 + s_1c_5 & s_{234}c_1c_5 + s_1s_5 & -c_{234}c_1 & x \\ -s_{234}s_1s_5 - c_1c_5 & -s_{234}s_1c_5 + c_1s_5 & c_{234}s_1 & y \\ c_{234}s_5 & c_{234}c_5 & s_{234} & z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

de la cual podemos derivar las siguientes ecuaciones para el punto $(x, y, z)_e$, donde se ubica el efector con los ángulos de articulaciones $(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5)$ con el origen del sistema de coordenadas en la base del robot:

- $x = -\cos(\theta_1) * (a_1 + a_2 * \cos(\theta_2) + a_3 * \cos(\theta_2 + \theta_3) + d_5 * \cos(\theta_2 + \theta_3 + \theta_4))$
- $y = \sin(\theta_1) * (a_1 + a_2 * \cos(\theta_2) + a_3 * \cos(\theta_2 + \theta_3) + d_5 * \cos(\theta_2 + \theta_3 + \theta_4))$
- $z = d_b + d_1 + a_2 * \sin(\theta_2) + a_3 * \sin(\theta_2 + \theta_3) + d_5 * \sin(\theta_2 + \theta_3 + \theta_4)$

Todos estos cálculos fueron realizados en MatLab Online [5].

VI. SIMULACIÓN EN GAZEBO Y VISUALIZACIÓN EN RVIZ

Uno de los primeros objetivos de la actualización integral del Scorbot ER-II fue poder contar con un entorno de simulación y visualización. Gracias a un [trabajo](#) del laboratorio de robótica en la Universidad de Sevilla [3], en el que presentan un modelo y una simulación de un Scorbot ER-VII pudimos desarrollar nuestra simulación en Gazebo y visualización en RViz como puede verse en la Figura 8 con algunas modificaciones menores para nuestra conveniencia, como el cambio de la longitud de la pinza para coincidir con la nuestra. Para esto se genera un modelo en formato URDF a partir de un modelo en Xacro que pudimos obtener del mencionado trabajo.

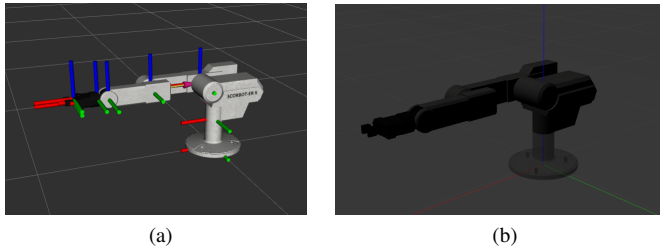


Figura 8. A la izquierda (a): visualización del Scorbot ER-VII en Rviz y a la derecha (b) simulación en Gazebo.

VII. CONCLUSIONES Y TRABAJO FUTURO

En este trabajo se presenta una re-ingeniería del manipulador Scorbot ER-VII para volver a ponerlo operativo. Para esto se realizó una actualización integral del hardware y software basada en Arduino y ROS. Podemos afirmar que el resultado de este trabajo fue positivo ya que volvimos a poder controlar el manipulador de manera completa. Como trabajo futuro nos proponemos desarrollar un planificador de trayectorias que se ejecute en la computadora del manipulador y comanden los movimientos que permitan que el efector final recorra varios puntos en el espacio de trabajo del robot.

- [1] L. Predescu, M. Predescu, *PLANNING THE TRAJECTORY OF THE SCORBOT-ER VII ROBOT*, 5th International Conference Computational Mechanics and Virtual Engineering COMEC 2013 24-25 October 2013, Braşov, Romania.
- [2] Laurentiu Predescu, Ion Stroe, *ON THE KINEMATIC OF THE SCORBOT ER-VII ROBOT*, U.P.B Sci. Bull., Series D, Vol. 77, Iss. 2, 2015.
- [3] RTC research group, *Event Based Scorbot ER-VII*.
- [4] Stanford Artificial Intelligence Laboratory et al., 2018. *Robotic Operating System*, Available at: <https://www.ros.org>.
- [5] The MathWorks, Inc. (2022). MATLAB version: 9.13.0 (R2022b). Accessed: January 01, 2023. Available: <https://www.mathworks.com>.
- [6] Intelitek incorporated. <https://intelitek.com/>
- [7] gazebo.org
- [8] <http://wiki.ros.org/rviz>
- [9] www.arduino.cc
- [10] www.tinkercad.com
- [11] <https://ultimaker.com/es/software/ultimaker-cura/>
- [12] <http://wiki.ros.org/xacro>
- [13] <http://wiki.ros.org/roscpp>
- [14] <http://wiki.ros.org/joy>
- [15] <http://wiki.ros.org/keyboad>
- [16] http://wiki.ros.org/universal_teleop

Sesión 9: Fusión de sensores y sistemas multi-robot

Álgebra de Clifford y aplicaciones a Sistemas de Navegación

Sebastian Incicco¹, Juan Ignacio Giribet², y Claudio Pose^{1,2}

¹*Facultad de Ingeniería, Universidad de Buenos Aires, Buenos Aires, Argentina*

²*Universidad de San Andrés y CONICET, Buenos Aires, Argentina*

{sincicco@fi.uba.ar, jgiribet@conicet.gov.ar, cldpose@fi.uba.ar}

Resumen—En este trabajo, se presenta un algoritmo de navegación integrada basado en cuaterniones tridentes, una extensión de los cuaterniones duales. La propuesta ofrece una metodología eficaz para lograr una navegación precisa y robusta, aprovechando las ventajas de los cuaterniones tridentes. Se validó el rendimiento del sistema de navegación mediante ensayos experimentales con un cuadricóptero equipado con dos computadoras de navegación: una ejecutando el algoritmo propuesto y la otra con un autopiloto comercial.

La implementación de este sistema integrado promete mejorar las capacidades de navegación en varios campos, incluyendo aeroespacial, robótica y vehículos autónomos. Este enfoque innovador tiene el potencial de impulsar el desarrollo de sistemas de navegación más avanzados y confiables para una variedad de aplicaciones críticas.

Teach and Repeat con calibración odométrica para robots omnidireccionales con sensor LiDAR

Sebastián Bedín
Instituto de Ciencias
de la Computación (ICC-CONICET)
Universidad de Buenos Aires,
Buenos Aires, Argentina
Email: sbedin@dc.uba.ar

Javier Civera
I3A,
Universidad de Zaragoza,
España
Email: jcivera@unizar.es

Matías Nitsche
Instituto de Ciencias
de la Computación (ICC-CONICET)
Universidad de Buenos Aires,
Buenos Aires, Argentina
Email: mnitsche@icc.fcen.uba.ar

Abstract—En este trabajo, presentamos un método novedoso de *teach-and-repeat* (T&R) para vehículos terrestres autónomos de tracción omnidireccional, basado en la fusión fuertemente acoplada de mediciones de LiDAR y odometría bajo una formulación relativa. Además de la localización del robot, realizamos una estimación en línea de los parámetros intrínsecos de la plataforma, lo que mejora significativamente la robustez y precisión de la localización del sistema. Demostramos la efectividad de nuestro enfoque, incluyendo el resultado de los parámetros de la plataforma y la estimación de la pose, así como el método general de T&R, mediante simulación.

I. INTRODUCCIÓN

Los vehículos terrestres no tripulados se utilizan en diversas formas y en numerosos ámbitos de aplicación, incluyendo el aspirado autónomo o el transporte, entre otros. De particular relevancia es su aplicación en depósitos o fábricas, donde el objetivo es lograr el mayor grado de automatización posible. En este contexto, los robots de tracción omnidireccional poseen una ventaja significativa debido a su movimiento 2D sin restricciones. Dada la naturaleza repetitiva de las tareas de los robots y la conveniencia de definir sus trayectorias a través de la demostración, los métodos de navegación *teach-and-repeat* (T&R) tienen una gran relevancia. Estos aspectos sirven de motivación para nuestro trabajo.

Aunque los sensores LiDAR destacan en robustez en comparación con las cámaras, están limitados en ciertas configuraciones geométricas como pasillos largos o entornos grandes con objetos fuera de su alcance, que son comunes en entornos como los mencionados. Por otro lado la precisión de la odometría está limitada por factores como: el deslizamiento de las ruedas, acentuado en robots de tracción omnidireccional debido a la naturaleza de sus ruedas con rodillos. La deformación de las ruedas por desgaste y cambios a la carga de la plataforma. O la variabilidad entre distintos robots parte de una flota.

En este trabajo, formulamos un método novedoso de T&R para robots omnidireccionales que fusiona, de manera fuertemente acoplada las mediciones de LiDAR y odometría en una representación relativa adecuada para el problema planteado. Al fusionar los dos sensores superamos las limitaciones de ambos. También incluimos parámetros específicos de la plataforma en el estado, para auto-calibrar la odometría. En

nuestros experimentos de simulación, demostramos la efectividad de nuestro enfoque en varios escenarios desafiantes.

II. TRABAJO RELACIONADO

La navegación T&R es una técnica clásica en la robótica autónoma y ha sido abordada a lo largo de los años desde varias perspectivas. La mayoría de las investigaciones se han centrado en robots terrestres equipados ya sea con LiDAR [9], [10], [1] o sensores visuales [14], [15], [20], [17]. Por otro lado, los trabajos que abordan otros tipos de locomoción, como submarinos [8] o vehículos aéreos [16], [12], son menos comunes.

En terminos de sensores a bordo del robot, en la literatura se ha explorado una amplia variedad: cámaras monoculares [20], estéreo [6], LiDAR [9], [19], [10], [1], radar [18] y combinaciones multi-modales como LiDAR-radar [1] o estéreo-inercial [7], [12]. Hasta donde sabemos, el enfoque que formulamos en este trabajo utilizando LiDAR 2D y *encoders* en rueda *mecanum* no ha sido abordado en la literatura. En particular, la auto-calibración simultánea de los parámetros de la plataforma, que proponemos en este trabajo, también es novedosa con respecto a trabajos anteriores.

III. MÉTODO

El método de navegación propuesto se basa en la técnica T&R presentada en [11], [12]. En T&R, se demuestra primero una trayectoria objetivo durante la fase de aprendizaje (*teach*) y luego es seguida en forma autónoma por un vehículo durante la fase de repetición (*repeat*). La localización incremental es utilizada durante ambas etapas para construir un mapa compuesto por *keyframes*. Durante la fase de *teach*, se generan *keyframes* los cuales definen la trayectoria objetivo. Durante la fase de *repeat*, el mapa construido en forma incremental se relaciona continuamente contra el mapa de referencia construido durante la fase previa. Así, el robot puede ser controlado para seguir la trayectoria objetivo.

En este trabajo, extendemos la técnica básica T&R para dirigir robots terrestres que cuentan con *encoders* de ruedas y un sensor LiDAR, mientras seguimos la misma formulación relativa del problema donde el mapa está compuesto por transformaciones relativas encadenadas. En este contexto, la

información odométrica que proviene de los *encoders* de las ruedas se incluye en la estimación siguiendo el conocido enfoque de preintegración [5], [4], que se complementa perfectamente con la formulación relativa propuesta. Además, para mejorar la información odométrica, incluimos la estimación en línea del radio de las ruedas. Para tratar los datos del sensor LiDAR, elegimos un enfoque simple y efectivo basado en el algoritmo de registro de escaneo *Iterative Closest Point* (ICP) [3]. Al calcular la transformación relativa a partir de un par de nubes de puntos LiDAR correspondientes a *keyframes* consecutivos, obtenemos una medición de pose relativa. Asimismo mostramos también cómo se puede obtener la incertidumbre correspondiente del algoritmo ICP.

IV. LOCALIZACIÓN INCREMENTAL

El objetivo de la localización incremental es estimar la pose actual del robot en relación con el *keyframe* más reciente del mapa actual, mientras se registran simultáneamente un conjunto de *keyframes* que describen la trayectoria del robot y se calibran los parámetros intrínsecos de la plataforma. Como se mencionó anteriormente, utilizamos un enfoque relativo, lo que significa que el estado a estimar se define como la pose relativa entre el *keyframe* actual y el anterior en la trayectoria:

$$\mathbf{x} = \left\{ \left\{ \mathbf{T}_i^{i+1} \right\}_{i=1 \dots n-1}, \mathbf{r} \right\}, \quad (1)$$

donde

$$\mathbf{T}_i^{i+1} = \begin{bmatrix} \mathbf{R}_i^{i+1} & \mathbf{t}_i^{i+1} \\ \mathbf{0} & 1 \end{bmatrix} \in \text{SE}(3)$$

representa la transformación relativa entre dos *keyframes* consecutivos denotados como i y $i+1$, y $\mathbf{r} \in \mathbb{R}_{>0}^4$ representa el vector de los radios de las ruedas desconocidos, en nuestro caso de un robot de tracción omnidireccional de cuatro ruedas.

A. Iterative Closest Point

Dado un par de nubes de puntos $\mathbf{P}$ y $\mathbf{Q}$ en dos marcos de referencia locales diferentes, el algoritmo ICP (Iterative Closest Point) nos permite encontrar una transformación relativa $\mathbf{L} \in \text{SE}(3)$ que alinea $\mathbf{P}$ con $\mathbf{Q}$ con un error mínimo. En otras palabras, nuestro objetivo es encontrar la transformación $\mathbf{L}$ que satisface la siguiente ecuación:

$$\mathbf{L} = \underset{\mathbf{L}'}{\operatorname{argmin}} C(\mathbf{Z}, \mathbf{L}') \quad (2)$$

donde $\mathbf{Z} = [\mathbf{P} \quad \mathbf{Q}]$ y

$$C(\mathbf{Z}, \mathbf{L}) = \sum_{j=1}^n [(\mathbf{R}\mathbf{p}_j + \mathbf{t} - \mathbf{q}_j) \cdot \mathbf{n}_j]^2 \quad (3)$$

corresponde a la métrica de error *punto-a-plano* definida en [3]. $\mathbf{R}$ y $\mathbf{t}$ se corresponden, respectivamente, con la parte de rotación y translación de $\mathbf{L}$. Las nubes de puntos $\mathbf{P} = \{\mathbf{p}_j\}_{j=1 \dots n}$, $\mathbf{Q} = \{\mathbf{q}_j\}_{j=1 \dots n}$ contienen puntos 3D $\mathbf{p}_j, \mathbf{q}_j \in \mathbb{R}^3$, y $\mathbf{n}_j \in \mathbb{P}^3$ representa la normal de un plano construido a partir de los k vecinos más cercanos de $\mathbf{q}_j$.

B. Residuo de LiDAR

En el contexto de la localización de robots basada en LiDAR, el algoritmo ICP puede utilizarse para determinar el movimiento del robot a partir de un par de escaneos láser adquiridos desde dos posiciones distintas. Para establecer la correspondencia entre los puntos en ambas nubes de puntos, se lleva a cabo un proceso de emparejamiento seleccionando el vecino entre los puntos de ambas nubes (donde se ignoran los puntos que no se emparejan).

Modelamos el residuo del sensor LiDAR de la siguiente manera:

$$\mathbf{r}_{\Delta \mathbf{L}} = \log \left(\left(\Delta \mathbf{L}_i^{i+1} \right)^{-1} \mathbf{T}_i^{i+1} \right)^\vee \quad (4)$$

donde $\Delta \mathbf{L}_i^{i+1}$ se obtiene utilizando el algoritmo ICP (como se describe en la sección anterior) a partir de lecturas de nubes de puntos adquiridas en los *keyframes* i e $i+1$. Como semilla inicial para la minimización de ICP, utilizamos la transformación relativa obtenida de las lecturas odométricas integradas entre i e $i+1$ (ver siguiente sección).

Para modelar la incertidumbre de $\Delta \mathbf{L}$ (donde eliminamos los índices por claridad), adoptamos el enfoque descrito en [2]. Específicamente, tenemos en cuenta la incertidumbre inicial en las lecturas de LiDAR propagando su covarianza a través de una aproximación de primer orden:

$$\Sigma_{\Delta \mathbf{L}} = \mathbf{J}_Z^{\Delta \mathbf{L}} \Sigma_Z \left(\mathbf{J}_Z^{\Delta \mathbf{L}} \right)^\top, \quad (5)$$

donde Σ_Z es la covarianza de las nubes de puntos, que se encuentra usualmente en la hoja de datos del sensor LiDAR.

Si bien $\Delta \mathbf{L}$ no presenta una forma cerrada, ya que se obtiene como resultado de un algoritmo de minimización (2), $\Delta \mathbf{L}$ y $\mathbf{Z}$ están relacionados por una función implícita. Además, sabemos que el gradiente de la función de costo C es cero en $\mathbf{L}$, por lo tanto su gradiente es nulo:

$$\frac{\delta C|_{\mathbf{L}}}{\delta \mathbf{L}} = \mathbf{0}^T \quad (6)$$

Como resultado, como se demuestra en [2], podemos aplicar el teorema de la función implícita y finalmente obtener $\mathbf{J}_Z^{\Delta \mathbf{L}}$:

$$\mathbf{J}_Z^{\Delta \mathbf{L}} = - \left(\frac{\delta^2 C(\mathbf{L}, \mathbf{Z})}{\delta \mathbf{L}^2} \right)^{-1} \frac{\delta^2 C(\mathbf{L}, \mathbf{Z})}{\delta \mathbf{Z} \delta \mathbf{L}} \Big|_{\mathbf{L}=\Delta \mathbf{L}} \quad (7)$$

C. Preintegración de Odometría

Para incluir información odométrica en la estimación, seguimos el enfoque de preintegración propuesto originalmente para mediciones inerciales [5]. Primero, definimos una lectura de los *encoders* $\mathbf{q}$ como:

$$\mathbf{q} = [\varphi_{fr} \quad \varphi_{fl} \quad \varphi_{rl} \quad \varphi_{rr}]^T \quad (8)$$

donde φ_{fr} , φ_{fl} , φ_{rl} y φ_{rr} representan los incrementos rotacionales para las ruedas delanteras-derecha, delanteras-izquierda, traseras-izquierda y traseras-derecha, respectivamente, según lo medido por los *encoders*.

Luego, definimos el modelo cinemático de la plataforma de tracción omnidireccional $\mathbf{K}(\mathbf{c})$ como:

$$\mathbf{K}(\mathbf{c}) = \frac{1}{4} \begin{bmatrix} r_{fr} & r_{fl} & r_{rl} & r_{rr} \\ r_{fr} & -r_{fl} & r_{rl} & -r_{rr} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{r_{fr}}{d} & \frac{-r_{fl}}{d} & \frac{-r_{rl}}{d} & \frac{r_{rr}}{d} \end{bmatrix} \quad (9)$$

con

$$\mathbf{c} = \begin{bmatrix} \mathbf{r} & d \end{bmatrix}^\top$$

$$d = \frac{d_1 + d_2}{2}$$

donde d_1 y d_2 representan la separación horizontal y lateral de las ruedas, respectivamente.

El movimiento incremental del robot, denotado como $\mathbf{O}(\mathbf{q}, \mathbf{c})$, se calcula entonces de la siguiente manera:

$$\mathbf{O}(\mathbf{q}, \mathbf{c}) = \exp(\mathbf{b}(\mathbf{q}, \mathbf{c})^\wedge) \in \text{SE}(3) \quad (10)$$

$$\mathbf{b}(\mathbf{q}, \mathbf{c}) = \mathbf{K}(\mathbf{c})\mathbf{q} \in \mathfrak{se}(3) \quad (11)$$

Para lecturas de *encoder* sucesivas $\mathbf{q}_1, \dots, \mathbf{q}_n$ adquiridas entre los *keyframes* i e $i+1$, podemos entonces construir una medida preintegrada $\Delta \mathbf{T}_i^{i+1}$ como:

$$\Delta \mathbf{T}_i^{i+1} = \mathbf{O}(\mathbf{q}_1, \mathbf{c}) \dots \mathbf{O}(\mathbf{q}_n, \mathbf{c}) \quad (12)$$

$$= \mathbf{O}_1^{n-1} \mathbf{O}(\mathbf{q}_n, \mathbf{c}) \quad (13)$$

Finalmente, el residuo de preintegración de la odometría se puede definir como:

$$\mathbf{r}_{\Delta \mathbf{T}} = \log((\Delta \mathbf{T}_i^{i+1})^{-1} \mathbf{T}_i^{i+1})^\vee \quad (14)$$

D. Estimación del Radio de Rueda

Para realizar la estimación en línea del radios de las ruedas $\mathbf{r}$, podemos adoptar un enfoque similar al descrito en [5] para estimar los términos de sesgo de la IMU. Específicamente, podemos preintegrar las lecturas de odometría utilizando un valor inicial de $\bar{\mathbf{r}}$, e incorporar un término de corrección para tener en cuenta las actualizaciones a $\mathbf{r}$ durante el proceso de estimación. Así, podemos definir:

$$\bar{\mathbf{c}} = \begin{bmatrix} \bar{\mathbf{r}} & d \end{bmatrix}^\top \quad (15)$$

donde d se obtiene antes de la estimación y se mantiene fijo, ya que no se espera que cambie con el tiempo.

De manera similar a [5], y eliminando los índices en $\Delta \mathbf{T}_i^{i+1}$ por claridad, utilizamos la siguiente aproximación para (13):

$$\Delta \mathbf{T}(\mathbf{q}, \mathbf{c}) \simeq \Delta \mathbf{T}(\mathbf{q}, \bar{\mathbf{c}}) \mathbf{J}_c^{\Delta \mathbf{T}}(\mathbf{q}, \mathbf{c} - \bar{\mathbf{c}}) \quad (16)$$

donde $\bar{\mathbf{c}}$ es el valor de $\mathbf{c}$ en el momento en que se inicializa la medición de preintegración y

$$\mathbf{J}_c^{\Delta \mathbf{T}} = \mathbf{J}_{\mathbf{O}_1^{n-1}}^{\Delta \mathbf{T}} \mathbf{J}_c^{\mathbf{O}_1^{n-1}} + \mathbf{J}_c^{\mathbf{O}} \quad (17)$$

donde

$$\mathbf{J}_{\mathbf{O}_1^{n-1}}^{\Delta \mathbf{T}} = \mathbf{J}_{\mathbf{O}_1^{n-1}}^{\oplus} \quad (18)$$

$$\mathbf{J}_c^{\mathbf{O}} = \mathbf{J}_r(\mathbf{b}) \mathbf{J}_c^{\mathbf{b}} \quad (19)$$

$$\mathbf{J}_c^{\mathbf{b}} = \frac{1}{4} \begin{bmatrix} \varphi_{fr} & \varphi_{fl} & \varphi_{rl} & \varphi_{rr} \\ \varphi_{fr} & -\varphi_{fl} & \varphi_{rl} & -\varphi_{rr} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{\varphi_{fr}}{d} & \frac{-\varphi_{fl}}{d} & \frac{-\varphi_{rl}}{d} & \frac{\varphi_{rr}}{d} \end{bmatrix} \quad (20)$$

con $\mathbf{J}_r(\mathbf{b})$ el jacobiano derecho de SE(3) evaluado en $\mathbf{b}$.

E. Covarianza de Odometría

Modelamos el ruido en los *encoders* incrementales como:

$$\mathbf{q} = \tilde{\mathbf{q}} + \epsilon_{\mathbf{q}} \quad (21)$$

donde $\tilde{\mathbf{q}}$ es la medida real del *encoder* y $\epsilon_{\mathbf{q}} \sim \mathcal{N}(\mathbf{0}, \Sigma_{\mathbf{q}})$. Además, dado que asumimos un valor previamente estimado para d , también consideramos su contribución a la covarianza propagada como $d = \tilde{d} + \epsilon_d$, $\epsilon_d \sim \mathcal{N}(0, \sigma_d)$.

Luego, podemos propagar la covarianza de un solo paso de integración (16) como:

$$\Sigma_{\mathbf{O}} = \mathbf{J}_{\mathbf{q}}^{\mathbf{O}} \Sigma_{\mathbf{q}} (\mathbf{J}_{\mathbf{q}}^{\mathbf{O}})^\top + \mathbf{J}_c^{\mathbf{O}} \Sigma_c (\mathbf{J}_c^{\mathbf{O}})^\top \quad (22)$$

donde $\Sigma_c = \mathbf{I}_5 [\mathbf{0} \ \sigma_d]^\top$ y $\mathbf{J}_{\mathbf{q}}^{\mathbf{O}} = \mathbf{J}_{\mathbf{b}}^{\mathbf{O}} \mathbf{J}_{\mathbf{q}}^{\mathbf{b}} = \mathbf{J}_r(\mathbf{b}) \mathbf{K}(\mathbf{c})$. Nótese que las covarianzas del radio de las ruedas no se consideran en este paso ya que estas variables forman parte de la estimación, a diferencia de las separaciones entre las ruedas, que son información previa. Las covarianzas del radio de las ruedas se obtienen como resultado del proceso de estimación.

La covarianza para un solo incremento puede acumularse a la medición completa preintegrada (13) de manera similar a como se hace en [5].

F. Repetición

Para la fase de repetición, adoptamos una metodología similar a la descrita en [12], aunque con adaptaciones para los diferentes sensores y el tipo de plataforma robótica involucrada. Proporcionamos primero un breve resumen del enfoque estándar de localización en el mapa.

La fase de repetición consta de dos procedimientos de localización distintos: 1) el método de localización incremental, que ya ha sido descrito en la sección anterior, y 2) el proceso de localización global sobre el mapa estimado en la fase de aprendizaje. Para realizar la localización global, seleccionamos el *keyframe* K_m del mapa aprendido que está más cercano al *keyframe* más reciente K_l en la fase de repetición, que luego se designa como el *keyframe* de referencia del mapa. Este *keyframe* está sujeto a un cambio continuo a medida que el robot se mueve y se generan nuevos *keyframes* durante el proceso de localización incremental en la fase de repetición.

Ejecutando el algoritmo de ICP entre los escaneos de ambos *keyframe* K_m y K_l , podemos determinar la transformación relativa $\mathbf{T}_m^l$ entre ellos. Posteriormente, para establecer la pose

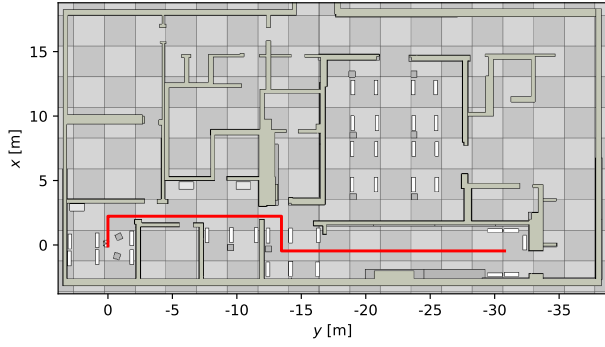


Fig. 1. Vista superior del entorno de simulación y trayectoria de enseñanza.

actual del robot con respecto al mapa anterior, combinamos $\mathbf{T}_m^l$ con la transformación más reciente $\mathbf{T}_l^n$ en la localización incremental en la fase de repetición (última pose en (1)) como:

$$\mathbf{T}_m^n = \mathbf{T}_k^l \mathbf{T}_l^n \quad (23)$$

Esta separación en dos hilos de localización distintos nos permite mantener una pose actualizada $\mathbf{T}_k^n$ en todo momento, incluso durante fallos temporales de la localización global. En tal caso, la pose encontrada previamente $\mathbf{T}_k^*$ se extiende (predice) con transformaciones relativas más nuevas $\mathbf{T}_n^{n+1}$, $\mathbf{T}_n^{n+2}$, etc. Esto añade robustez a todo el sistema T&R e incluso permite desviaciones del mapa aprendido si fueran necesarias (por ejemplo, para evitar un obstáculo no presente previamente).

A partir del conjunto de *keyframes* cercanos al *keyframe* de referencia del mapa, se construye una trayectoria suave que tiene en cuenta las velocidades lineales y rotacionales registradas durante la fase de aprendizaje. Esto permite evaluar la trayectoria en cualquier momento t y obtener un punto de control con el fin de seguir suavemente el camino aprendido (para más detalles ver [12]).

Dado que las contribuciones de este trabajo se centran principalmente en el algoritmo de localización incremental, optamos por una estrategia de control directa basada en un controlador PI de posición/orientación para nuestros experimentos. Dicho controlador opera de manera independiente en cada uno de los tres ejes controlables del robot.

V. RESULTADOS

A. Configuración Experimental

Para evaluar el rendimiento del método T&R propuesto, realizamos una serie de experimentos en el simulador *Coppelia V-Rep*. La plataforma robótica utilizada fue el *KUKA youBot*, que cuenta con un sistema de locomoción omnidireccional compuesto por cuatro ruedas de tipo *mechanum*. Las lecturas de los *encoders* de las ruedas simuladas se obtuvieron convirtiendo la rotación de las ruedas en 2000 *ticks* discretos. Para el LiDAR elegimos el sensor *Hokuyo URG-04LX*, el cual proporciona lecturas a 50Hz hasta una distancia de 4 m sobre un ángulo de 240°. Introducimos ruido en las lecturas del LiDAR perturbando la distancia de cada lectura con ruido

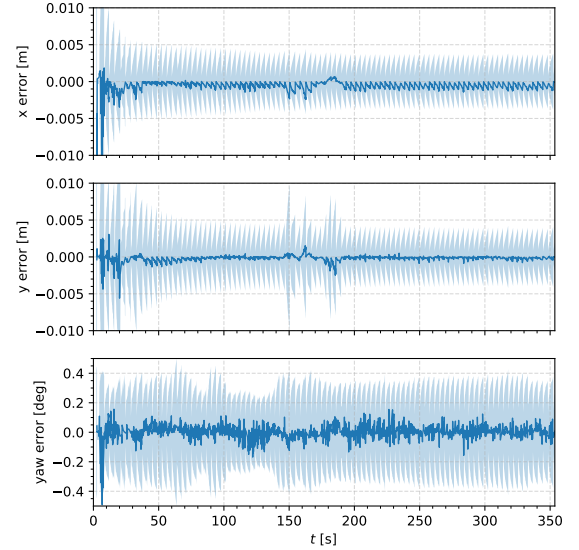


Fig. 2. Errores de pose relativa (en azul oscuro) y incertidumbres estimadas (en azul claro).

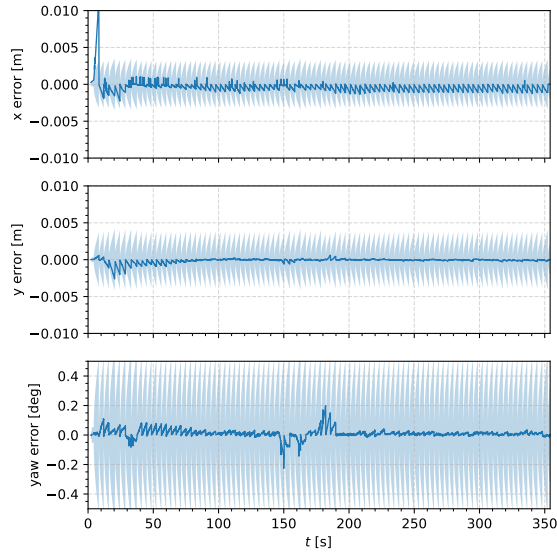
gaussiano, con una incertidumbre del 2 % sobre la distancia, como se propuso en [13].

El entorno simulado corresponde a un modelo de una región del *Centro de Conferencias Marcum* de la Universidad de Miami (ver Figura 1). Los experimentos consisten en una fase de demostración inicial sobre el entorno y posteriores ejecuciones con control a lazo cerrado de la fase de repetición.

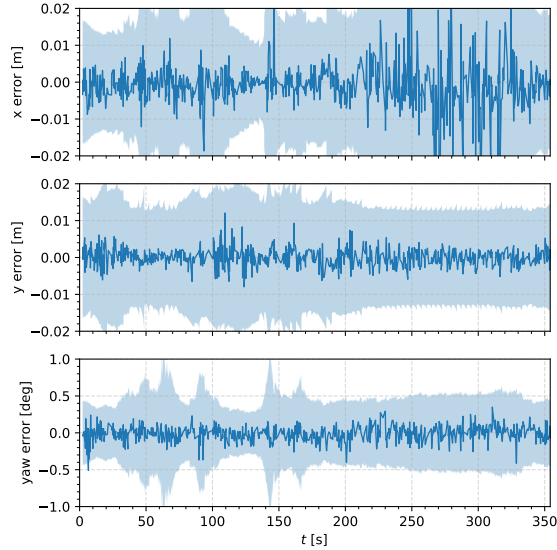
B. Localización Incremental y Calibración en Línea

En esta sección, evaluamos la calidad de la localización incremental mediante la evaluación del error de la última transformación relativa estimada frente al *ground truth*, así como los valores de los radios de las ruedas estimados (comenzando con un error inicial de 2 mm para cada rueda). La Figura 2 muestra el error de pose (en azul oscuro), junto con la covarianza de pose correspondiente (ilustrada con un intervalo de confianza del 99% en azul claro). Además, presentamos la pose relativa $\Delta\mathbf{T}$ y $\mathbf{L}$, junto con su covarianza (fig. 3(a), 3(b)), que proporciona información sobre el rendimiento de cada modalidad de sensado a lo largo del tiempo y su relación con la pose estimada final. Por último, la Figura 4 ilustra el error y las incertidumbres estimadas de los radios de las ruedas durante la calibración en línea.

Los resultados presentados muestran que, en general, los errores de localización son pequeños y consistentes con las incertidumbres estimadas. Al analizar la convergencia de los radios de las ruedas, se observa que el error inicial se reduce rápidamente. Entre los intervalos de tiempo de $t = 255$ s y $t = 307$ s, cuando el robot navega por un pasillo largo, el error en la dirección x así como la incertidumbre de la pose obtenida de ICP aumentan significativamente. Sin embargo, dado que la pose obtenida de la preintegración de odometría no se ve afectada por esta situación, la estimación de pose después de la fusión tiene un error bajo que es consistente con la baja varianza estimada.



(a) Errores de pose relativa (en azul oscuro) e incertidumbres estimadas (en azul claro) para la preintegración de odometría.



(b) Errores de pose relativa (en azul oscuro) y incertidumbres estimadas (en azul claro) para ICP.

Fig. 3. Poses relativas de la preintegración de odometría y ICP.

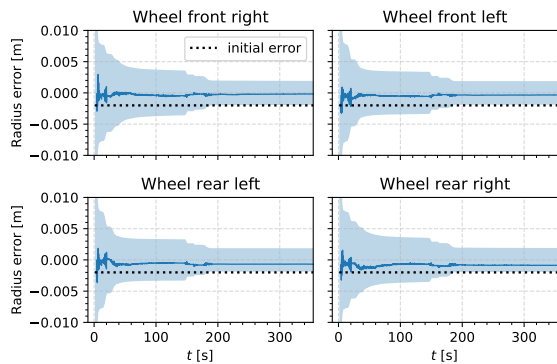


Fig. 4. Resultados de la estimación de los radios de las ruedas (errores en azul oscuro, incertidumbres en azul claro).

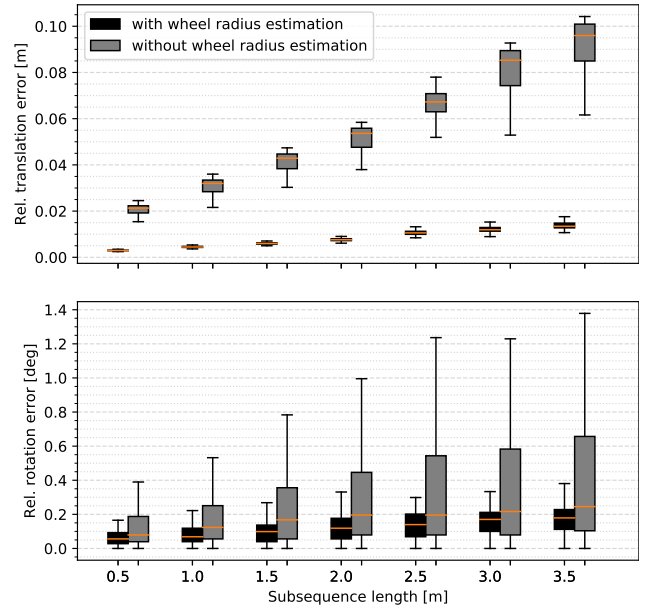


Fig. 5. Diagramas de caja y bigotes que muestran el Error de Pose Relativa (RPE) en subsecuencias de longitud creciente, con y sin estimación de radios de ruedas. Observa la mejora sustancial cuando se estima los radios de ruedas.

Realizamos un análisis adicional del error de localización midiendo el Error de Pose Relativa (RPE), como se define en [12], para subsecuencias de longitudes crecientes (Figura 5), tanto teniendo habilitada y deshabilitada la estimación en línea de los radios de las ruedas. Dado que nuestro enfoque utiliza un método relativo y la integración de la lectura odométrica acumula error entre un par de *keyframes*, este análisis nos permite observar la tasa a la que el error se acumula con respecto a la distancia. De esta manera, podemos hacer un equilibrio entre la precisión de la localización y la precisión del recorrido al decidir a qué distancia colocar los *keyframes*. De la Figura 5 podemos observar el impacto en la estimación de movimiento odométrico que surge de un error relativamente pequeño en los radios de las ruedas, además de cómo mejora el rendimiento general de la localización al re-alizar la estimación en línea de estos parámetros. Esto es cierto incluso para distancias relativamente cortas y particularmente para el componente translacional.

C. Navegación Autónoma durante la fase de Repetición

Para la fase de repetición permitimos que el sistema controle el movimiento del robot para analizar la localización contra el mapa previamente construido durante la fase de enseñanza, como fue descrito en la sección anterior. Para esto presentamos el error entre la pose relativa del robot con respecto al mapa previo (como se explica en la sección IV-F, para más detalles ver [11]) en comparación con la información de *ground truth*, así como su incertidumbre (ver figura 6). También comparamos la trayectoria seguida por el robot en comparación con la de la fase de aprendizaje en la Figura 7.

De manera similar a los resultados obtenidos para la fase de enseñanza, podemos ver que el error en la pose del robot

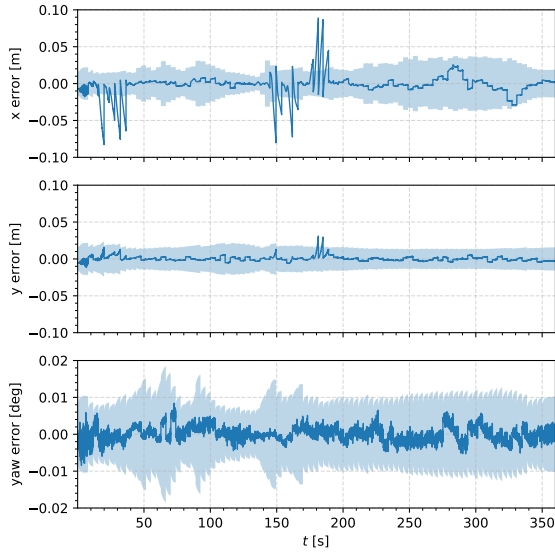


Fig. 6. Error (azul oscuro) y estimación de incertidumbres (azul claro) entre poses relativas en la fase de repetición.

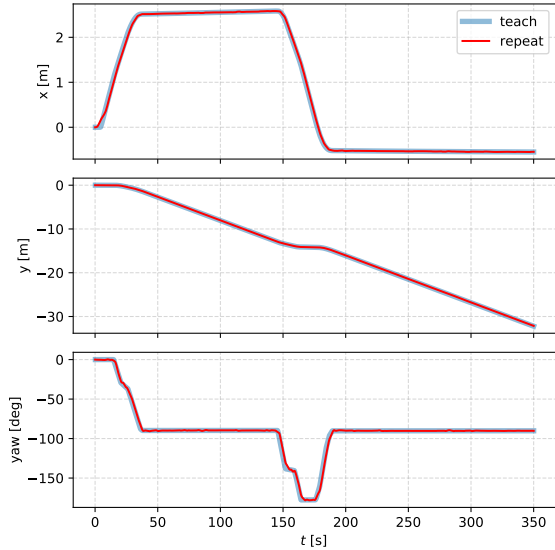


Fig. 7. Trayectorias de enseñanza y repetición. Observar que, utilizando nuestro enfoque, la trayectoria de repetición sigue de cerca la de enseñanza.

con respecto al mapa previo es generalmente pequeño y las incertidumbres estimadas son consistentes con el mismo. Además, podemos ver que el robot sigue con éxito la ruta enseñada previamente.

VI. CONCLUSIONES

En este trabajo presentamos una nueva técnica T&R orientada a plataformas robóticas omnidireccionales, fusionando información de sensores LiDAR y *encoders*, bajo una formulación relativa. Nuestros resultados demuestran que la integración de ambas modalidades de sensado no solo permite una localización robusta en entornos donde los sensores LiDAR a menudo presentan limitaciones, sino que también facilita la estimación en tiempo real de los radios de las ruedas de

la plataforma. Esto, a su vez, mejora significativamente la precisión de la estimación de movimiento basada en el modelo odométrico.

Como parte de nuestro trabajo futuro, planeamos realizar experimentos en un entorno real para validar nuestro enfoque. Además, tenemos como objetivo mejorar nuestro enfoque de calibración en línea para adaptarse a cambios en los radios de las ruedas que puedan surgir debido a variaciones en la carga durante las operaciones de transporte. También pretendemos refinar nuestra estrategia de control del robot para mejorar aún más el rendimiento del método.

AGRADECIMIENTOS

Este trabajo fue financiado por el proyecto PID2021-127685NB-I00 y TED2021-131150B-I00 y el proyecto DGA_FSE-T45_20R del Gobierno de Aragón.

REFERENCES

- [1] Keenan Burnett, Yuchen Wu, David J Yoon, Angela P Schoellig, and Timothy D Barfoot. Are we ready for radar to replace lidar in all-weather mapping and localization? *IEEE Robotics and Automation Letters*, 7(4):10328–10335, 2022.
- [2] Andrea Censi. An accurate closed-form estimate of icp's covariance. In *IEEE International Conference on Robotics and Automation*, page 3167–3172, 2007.
- [3] Yang Chen and Gerard Medioni. Object Modeling by Registration of Multiple Range Images. In *IEEE Conf. on Robotics and Automation*, 1991.
- [4] Jeremie Deray, Joan Solà, and Juan Andrade-Cetto. Joint on-manifold self-calibration of odometry model and sensor extrinsics using pre-integration. In *2019 European Conference on Mobile Robots (ECMR)*, pages 1–6. IEEE, 2019.
- [5] Christian Forster, Luca Carlone, Frank Dellaert, and Davide Scaramuzza. On-manifold preintegration for real-time visual-inertial odometry. *IEEE Transactions on Robotics*, 33(1):1–21, 2017.
- [6] Paul Furgale and Timothy D. Barfoot. Visual teach and repeat for long-range rover autonomy. *Journal of Field Robotics*, (2006):1–27, 2010.
- [7] Fei Gao, Luqi Wang, Boyu Zhou, Xin Zhou, Jie Pan, and Shaojie Shen. Teach-repeat-replan: A complete and robust system for aggressive flight in complex environments. *IEEE Transactions on Robotics*, 36(5):1526–1545, 2020.
- [8] Peter King, Andrew Vardy, and Alexander L Forrest. Teach-and-repeat path following for an autonomous underwater vehicle. *Journal of Field Robotics*, 35(5):748–763, 2018.
- [9] Philipp Krüsi et al. Lighting-invariant Adaptive Route Following Using Iterative Closest Point Matching. *JFR*, 32(4):534–564, 2015.
- [10] Colin McManus, Paul Furgale, Braden Stenning, and Timothy D Barfoot. Lighting-invariant visual teach and repeat using appearance-based lidar. *Journal of Field Robotics*, 30(2):254–287, 2013.
- [11] Matias Nitsche, Facundo Pessag, and Javier Civera. Visual-inertial teach & repeat for aerial robot navigation. In *2019 European Conference on Mobile Robots (ECMR)*, pages 1–6. IEEE, 2019.
- [12] Matias Nitsche, Facundo Pessag, and Javier Civera. Visual-inertial teach and repeat. *Robotics and Autonomous Systems*, 131:103577, 2020.
- [13] Yoichi Okubo, Cang Ye, and Johann Borenstein. Characterization of the Hokuyo URG-04LX laser rangefinder for mobile robot obstacle negotiation. In *SPIE Unmanned Systems Technology XI*, 2009.
- [14] Chris J Ostafew, Angela P Schoellig, and Timothy D Barfoot. Visual teach and repeat, repeat, repeat: Iterative learning control to improve mobile robot path tracking in challenging outdoor environments. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 176–181. IEEE, 2013.
- [15] Michael Paton, Kirk MacTavish, Michael Warren, and Timothy D. Barfoot. Bridging the appearance gap: Multi-experience localization for long-term visual teach and repeat. In *IROS*, 2016.
- [16] Andreas Pfrunder, Angela P Schoellig, and Timothy D Barfoot. A Proof-of-Concept Demonstration of Visual Teach and Repeat on a Quadcopter Using an Altitude Sensor and a Monocular Camera. In *CRV*, 2014.

- [17] Zdeněk Rozsypálek, Tomáš Rouček, Tomáš Vintr, and Tomáš Krajník. Multidimensional particle filter for long-term visual teach and repeat in changing environments. *IEEE Robotics and Automation Letters*, 8(4):1951–1958, 2023.
- [18] Ștefan Săftescu, Matthew Gadd, Daniele De Martini, Dan Barnes, and Paul Newman. Kidnapped radar: Topological radar localisation using rotationally-invariant metric learning. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4358–4364. IEEE, 2020.
- [19] Christoph Sprunk, Gian Diego Tipaldi, Andrea Cherubini, and Wolfram Burgard. Lidar-based teach-and-repeat of mobile robot trajectories. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3144–3149. IEEE, 2013.
- [20] Cheng Zhao, Li Sun, Tomáš Krajník, Tom Duckett, and Zhi Yan. Monocular teach-and-repeat navigation using a deep steering network with scale estimation. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2613–2619. IEEE, 2021.

Base de Datos para Conducción Autónoma. Sensores y Sincronización

Juan Giribet^{1,2}, Ignacio Mas^{1,2}, Agustín Roca¹, Guillermo Marzik^{1,2}, Gabriel Torre¹, Claus Rosito¹ y Gastón Castro^{1,2}

¹Laboratorio de Inteligencia Artificial y Robótica. Universidad de San Andrés y ²CONICET

Email: linar@udesa.edu.ar

Resumen

En este trabajo, presentamos una nueva base de datos diseñada para aplicaciones de conducción autónoma en el Área Metropolitana de Buenos Aires (AMBA). Esta base de datos recopila información de diversas fuentes de sensores, incluyendo cámaras, LiDAR, GPS, IMU y micrófono, ofreciendo una rica diversidad de perspectivas y dimensiones para el análisis. Detallamos el montaje de los sensores y la estructura del conjunto de datos, incluyendo los procesos de calibración y sincronización para garantizar la precisión y la utilidad de la información recogida.

Comparando nuestra base de datos con otras existentes, destacamos las singularidades y las ventajas del enfoque propuesto, que se centra en la variedad y la profundidad de los datos recopilados, especialmente adaptados a las peculiaridades del entorno urbano y vial del AMBA en la Argentina. Presentamos también resultados experimentales que demuestran la aplicabilidad de nuestra base de datos en el desarrollo y la mejora de sistemas de conducción autónoma, proponiendo un marco para futuras investigaciones y aplicaciones en este campo emergente.

Fusión de mapas mediante encuentros frecuentes para la exploración multirobot

Martin Nievas*, Gastón Araguás[†] y Claudio J. Paz[‡]

Centro de Investigación en Informática para la Ingeniería,
Universidad Tecnológica Nacional, Facultad Regional Córdoba

Email: {*,[†]garaguas, [‡]cpaz}@frc.utn.edu.ar

Resumen—Las estrategias de exploración son cruciales para que los equipos de robots puedan explorar y navegar de manera eficiente un ambiente desconocido. En este trabajo se presenta una estrategia de exploración para coordinar la exploración de un equipo de robots con encuentros frecuentes y comunicación limitada. El enfoque propuesto aprovecha los encuentros entre robots para intercambiar informaciones y generar un mapa global a través de la fusión de los mapas parciales de cada robot, mediante un esquema de dos etapas utilizando los algoritmos SVD e ICP. Se presenta un esquema para la detección de fronteras tratando los mapas generados por los robots como imágenes, y guiar la exploración. Finalmente, se presentan resultados tanto de simulación como experimentación en el mundo real.

Keywords—multirobot, simulación, exploración, ROS2

Descomposición Óptima en Subframeworks con Aplicaciones al Control de Redes Multirobot

Juan F. Presenza^{1,3}, Ignacio Mas^{2,3}, Juan I. Giribet^{2,3}, J. Ignacio Alvarez-Hamelin^{1,3}

¹ Facultad de Ingeniería, Universidad de Buenos Aires

² Laboratorio de Inteligencia Artificial y Robótica - Universidad de San Andrés, Buenos Aires, Argentina.

³ CONICET - Argentina

Resumen—Este trabajo presenta una extensión del formalismo que captura la de rigidez de un framework a partir de la subdivisión del mismo en subframeworks. Este abordaje permite modelar una red de robots móviles, y ha demostrado su utilidad en el control de redes multirobot permitiendo el diseño de controladores descentralizados y escalables. Teniendo en cuenta métricas realistas del control multirobot como el tiempo de retardo y la carga de las comunicaciones entre agentes, este trabajo compara cualitativa y cuantitativamente distintos tipos de descomposición. A partir de este análisis, se establecen criterios de optimalidad que permiten seleccionar descomposiciones óptimas. A su vez, se presenta un algoritmo escalable que permite aproximar estas soluciones óptimas. Mediante simulaciones computacionales, se valida el esquema propuesto y se analizan los resultados.

I. INTRODUCCIÓN

En la última década, la coordinación de sistemas de múltiples robots móviles interconectados se ha convertido en un área de intensa investigación, motivada por las ventajas en adaptabilidad, robustez y escalabilidad en una amplia variedad de aplicaciones [1], [2]. La utilización de esquemas de control centralizados, donde un centro de cómputo debe procesar todas las variables del sistema, no resulta apropiada cuando la cantidad de robots es elevada. Esto se debe a principalmente a los altos requerimientos de capacidad computacional del nodo central, los efectos de cuello de botella en los canales de comunicación, y falta de robustez por la existencia de un punto de falla. Todas ellas se traducen en una baja escalabilidad. Por el contrario, en las técnicas descentralizadas, las comunicaciones, el cómputo y la decisión se reparten entre los agentes. Los mismos actúan únicamente en función de la percepción de su entorno a través de sensores, y de las interacciones con otros vehículos a través de comunicaciones punto-a-punto. Se logra así dotar a los esquemas de una mayor escalabilidad, a expensas de una mayor dificultad para que los agentes accedan a la información asociada a la red.

En particular, numerosos desafíos se presentan cuando los agentes únicamente cuentan con sensores que les permiten acceder a información relativa a otros robots, por ejemplo, posición relativa o distancia. En tales casos, la topología de la red subyacente, dada por las interacciones entre los agentes, tiene una gran importancia. Más aún, esta topología suele ser dinámica, es decir, permite la creación y destrucción de enlaces, producto del movimiento de los robots o de cambios en el ambiente circundante. Un caso de particular

importancia en la literatura se presenta cuando los robots cuentan únicamente con mediciones de distancia respecto de sus vecinos [3], [4]. La rigidez de los grafos resulta ser la propiedad necesaria y suficiente para controlar al conjunto de robots a partir de dichas mediciones. Por este motivo, se han realizado importantes esfuerzos para diseñar estrategias de control que permitan mantener o recuperar la rigidez en redes con topología dinámica.

Las estrategias clásicas de control de rigidez pueden ser clasificadas como continuas [5], [6], [7] y combinatorias [8], [9]. En los esquemas continuos, el controlador guía el movimiento de los robots de forma tal de mantener la rigidez del grafo, permitiendo cambios en la topología de la red. Esto se logra a través de la maximización del autovalor de rigidez, un parámetro que sirve como medida del grado de rigidez de una grafo. La implementación descentralizada se basa en la utilización de filtros de consenso que permiten obtener localmente una estimación del autovalor de rigidez y del correspondiente autovector, necesarios para computar la acción de control. El desempeño del control depende en gran medida de la capacidad de estimar rápidamente estos parámetros. Sin embargo, la velocidad de convergencia de los estimadores está relacionada inversamente con el diámetro de la red [10]. Además, cuando el número de conexiones mantenidas por cada robot es limitado, el diámetro crece al menos logarítmicamente con el número de robots, imponiendo un claro límite a la escalabilidad de estos abordajes.

Alternativamente, en los enfoques combinatorios se diseñan reglas locales para la adición y remoción de enlaces entre agentes, las cuales son luego empleadas para dictar los movimientos de los robots. La naturaleza discreta de estos esquemas los dotan de mayor eficiencia, dado que únicamente requieren realizar cálculos e intercambio de información durante las transiciones, es decir, ante la inminente pérdida o creación de un enlace. Estos esquemas poseen mejor escalabilidad dado que las acciones de control se toman con datos locales, sin requerir información global. No obstante, son conservativos en cuanto a la eliminación de enlaces, debido a que conexiones redundantes pueden no ser detectadas de forma local. Al ser enfoques locales, no se puede emplear el autovalor de rigidez ni ninguna medida de rigidez que permite definir un criterio de optimalidad relacionado.

En [11], [12] se buscó superar estas limitaciones a través

de la definición de subframeworks. A partir del análisis de rigidez de los mismos, se puede determinar la rigidez de todo el framework. Esto elimina la necesidad de intercambiar información de extremo a extremo de la red, lo que mejora notablemente la escalabilidad del sistema. De esta manera, se diseñó un controlador descentralizado capaz maximizar los autovalores de rigidez de los subframeworks, y de esta manera garantizando la rigidez de todo el framework. Para el funcionamiento del controlador, la información debe ser propagada en el interior de cada subframework, por lo que el tamaño de los mismos influye significativamente en el desempeño del control. El propósito de este trabajo es analizar este comportamiento y establecer un criterio de optimalidad para la elección de los subframeworks que reflejen objetivos realistas de un sistema de control multirobot.

II. LA RIGIDEZ DE LOS FRAMEWORKS

II-A. Preliminares

Modelamos una red multirobot mediante un grafo $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ donde $\mathcal{V} = \{1, \dots, n\}$ representa el conjunto de agentes y $\mathcal{E} \subseteq \{\{i, j\} \mid i, j \in \mathcal{V}, i \neq j\}$ el conjunto de m enlaces de comunicación punto-a-punto. El framework $\mathcal{F} = (\mathcal{G}, p)$ consiste en una realización de $\mathcal{G}$ dada por la función $p : \mathcal{V} \rightarrow \mathbb{R}^d$ que asigna posiciones, a los nodos, en un espacio d -dimensional. Decimos que un framework es completo si tiene todos los enlaces posibles, es decir, $m = n(n-1)/2$. El número de conexiones de un nodo se representa mediante el grado del vértice $\delta(i)$. La distancia geodésica $d(i, j)$ entre dos nodos i, j es la mínima cantidad de enlaces que deben atravesarse para conectarlos; la excentricidad de un vértice $e(i) = \max_j d(i, j)$, y el diámetro del grafo $\text{diam}(\mathcal{G}) = \max_i e(i)$.

II-B. Rigidez de Frameworks

Intuitivamente, decimos que un framework $(\mathcal{G}, p)$ es rígido cuando, conocido el conjunto de distancias euclídeas asociadas a los enlaces $\{z_{ij} \mid \{i, j\} \in \mathcal{E}\}$, es posible reconstruir el conjunto de posiciones $\{p(i) \mid i \in \mathcal{V}\}$, excepto por rotaciones, reflexiones y traslaciones. A continuación, se presenta la definición formal de este concepto [13].

Dos frameworks $(\mathcal{G}, p)$ y $(\mathcal{G}, q)$ se dicen equivalentes si se cumple

$$\|p(i) - p(j)\| = \|q(i) - q(j)\| \quad \text{para todo } \{i, j\} \in \mathcal{E}; \quad (1)$$

y se dicen congruentes si

$$\|p(i) - p(j)\| = \|q(i) - q(j)\| \quad \text{para todo } i \in \mathcal{V}, j \in \mathcal{V}. \quad (2)$$

El framework $(\mathcal{G}, p)$ es rígido si existe $\varepsilon > 0$ tal que para toda realización q que satisface $\|p(i) - q(i)\| < \varepsilon$ para todo i , se cumple que (1) implica (2). Caso contrario, se dice que $(\mathcal{G}, p)$ es flexible.

Para establecer la rigidez mediante métodos computacionales, primero definimos la matriz de rigidez $R \in \mathbb{R}^{m \times dn}$ de un framework como el jacobiano de la función

$$f(\mathbf{p}) = [\dots \|p(i) - p(j)\| \dots]^T, \quad \mathbf{p} = [p(1)^T \dots p(n)^T]^T$$

donde cada componente del vector $f(\mathbf{p})$ corresponde a un enlace. Si ordenamos los autovalores $\lambda_1, \dots, \lambda_{dn}$ de $R^T R$ de forma creciente, entonces se puede probar (ver [5]) que un framework es rígido si y solo si

$$\lambda_{\frac{d(d+1)}{2}+1}(R^T R) > 0. \quad (3)$$

Dicho valor es denominado autovalor de rigidez del framework, y sirve no solo como condición necesaria y suficiente, sino también como una medida de su grado de rigidez.

III. DESCOMPOSICIÓN EN SUBFRAMEWORKS

En [11], [12] se presentó un esquema de descomposición de frameworks que fue utilizado para el control descentralizado de rigidez. El mismo se define a continuación.

Definición 1 (Descomposición densa). Sea $\mathcal{F}$ un framework y $h : \mathcal{V} \rightarrow \mathbb{Z}_{>0}$ una función que asocia un alcance (medido en distancia geodésica) a cada nodo. Se llama descomposición densa al conjunto $\text{dense}(\mathcal{F}) = \{\mathcal{F}_1, \dots, \mathcal{F}_n\}$ donde $\mathcal{F}_i = (\mathcal{G}_i, p)$ tal que

$$\mathcal{V}_i = \{j \mid d(i, j) \leq h(i)\}, \mathcal{E}_i = \{\{j, k\} \in \mathcal{E} \mid j, k \in \mathcal{V}_i\}. \quad (4)$$

Al robot i se lo llama centro del subframework $\mathcal{F}_i$, y a $h(i)$ se le dice alcance del mismo.

Notar que, para un mismo framework, pueden existir múltiples descomposiciones, correspondientes a las distintas funciones de alcance h . Si $h(i) = 0$ decimos que el subframework $\mathcal{F}_i$ es trivial, y definimos a la descomposición trivial como aquella donde todo subframework es trivial. Análogamente, un subframework es unitario si $h(i) = 1$, y la descomposición unitaria es aquella donde esto ocurre para todo i . Por otro lado, decimos que una descomposición es impropia si existe algún subframework $\mathcal{F}_i$ que cubre todo el framework, es decir, $h(i) = e(i)$.

La rigidez de $\mathcal{F}$ queda determinada por el siguiente teorema, el cual permite expresarla como un conjunto de condiciones locales.

Teorema 1 (Rigidez por Descomposición Densa [11]). Sea $\mathcal{G}$ conexo. Entonces $\mathcal{F}$ es rígido si y sólo si existe una descomposición $\text{dense}(\mathcal{F})$ tal que todos los subframeworks son rígidos.

Notar que este teorema, si bien indica las condiciones, no especifica cómo debe realizarse la descomposición; en otras palabras, resta analizar cómo elegir los alcances $h(1), \dots, h(n)$ de forma apropiada. Como se verá en el apartado III-B, cuando la descomposición es densa, la elección de un vector de alcances óptimo resulta directa. En cambio, cuando la descomposición no es densa, la elección no es tan simple. Este caso es analizado en profundidad en la Sección IV.

III-A. Métricas de la descomposición

Para comparar de forma cuantitativa las distintas descomposiciones, es necesario definir métricas sobre las mismas, que permitan evaluar algún comportamiento deseado.

En primer lugar, para implementar un algoritmo de rigidez de forma descentralizada, los agentes deben intercambiar paquetes de información que pueden recorrer caminos de distinta longitud hasta llegar a su destino. Por lo tanto, es necesario considerar el tiempo transcurrido durante ese recorrido. En los esquemas de control por subframeworks, los nodos de un subframework i se comunican entre sí enrutando mensajes a través del centro, resultando en caminos de longitud menor o igual al doble del alcance $h(i)$. Esto se observa en la Fig. 1a. Por ello, definimos el máximo retardo asociado a una descomposición como:

$$\mathcal{D}(h) = 2 \max_{i \in \mathcal{V}} h(i). \quad (5)$$

Notar que $2 \leq \mathcal{D}(h) \leq 2\text{diam}(\mathcal{G})$, extremos que se alcanzan cuando la descomposición es unitaria, y cuando es impropia, respectivamente. Empleamos $\mathcal{D}_{\text{ref}}(\mathcal{G}) = \text{diam}(\mathcal{G})$ como parámetro de referencia para esta métrica, ya que es el mínimo retardo asociado a los algoritmos de intercambio de información global, como los protocolos de consenso.

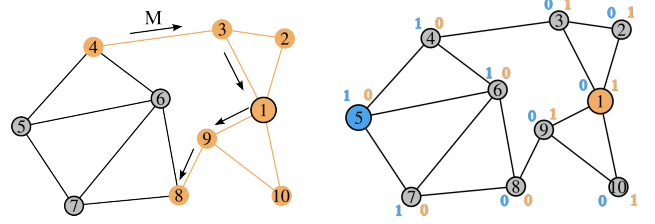
En segundo lugar, el intercambio de información bidireccional entre los nodos de un subframework y su centro, implica la utilización de los canales de comunicación. El esquema de control presentado en [12], requiere que cada robot emita dos tokens: T_A con información de las acciones de control, y T_S con datos del estado del mismo; los cuales son propagados por la red hasta llegar a sus respectivos destinos. El token acción emitido por i debe llegar a todos los nodos del subframework $\mathcal{F}_i$, por lo que es enviado a $h(i)$ hops de distancia; mientras que el de estado es enviado a todo robot cuyo subframework contiene a i , es decir a los nodos $\{j \mid d(i, j) \leq h(j)\}$. Esto implica que T_S debe ser enviado a $s(i)$ hops, donde $s(i) = \max\{h(j) \mid d(i, j) \leq h(j)\}$.

De esta forma, a medida que el alcance de los subframeworks crece, los tokens deben viajar una mayor distancia, lo que produce un aumento de la carga sobre los enlaces. Dada la descomposición dictada por el alcance h , para cuantificar la carga total de comunicación asociada a la propagación de tokens T_A, T_S , proponemos la función

$$\mathcal{L}(h) = \sum_{i \in \mathcal{V}} (N_A(i) + N_S(i)) \delta(i), \quad (6)$$

donde $N_A(i) = |\{j \mid d(i, j) < h(j)\}|$ y $N_S(i) = |\{j \mid d(i, j) < s(j)\}|$ representan la cantidad de tokens de acción y de estado que pasan por el robot i , respectivamente. Esto se observa en Fig. 1b. $\mathcal{L}$ mide la cantidad total de veces que un enlace es empleado para transmitir un token. El valor $\mathcal{L}_{\text{ref}}(\mathcal{G}) = 2m$, donde m es el número total de enlaces, es tomado como referencia, dado que representa la utilización de los canales en una regla de vecinos más cercanos.

En [11] se empleó, de forma exitosa, la función $\mathcal{L}$ como función de costo en un controlador de gradiente descendiente para guiar el movimiento de los robots. De esta forma, se logró inducir, en los robots, trayectorias que tienden a eliminar los enlaces con mayor carga, manteniendo la rigidez del sistema.



(a) Camino de un mensaje en $\mathcal{F}_1$. (b) Carga impuesta por $\mathcal{F}_1$ y $\mathcal{F}_5$.

Figura 1: Métricas para comparar distintas descomposiciones en subframeworks. En (a) se muestra el subframework $\mathcal{F}_1$ con $h(1) = 2$. En (b), se muestran las cargas impuestas por dos tokens: uno de $\mathcal{F}_1$ que debe viajar 2 hops y otro $\mathcal{F}_5$ que debe viajar 1 hop.

III-B. Alcance mínimo de rigidez

Puede observarse que $\mathcal{D}, \mathcal{L}$ son funciones no decrecientes respecto a los valores $h(1), \dots, h(n)$. De esta forma, para minimizar ambas métricas, es apropiado seleccionar los menores alcances posibles. Entonces, si $\mathcal{F}_i[\eta]$ denota el subframework con centro en i y alcance η , definimos el alcance mínimo de rigidez como

$$h_R(i) = \min\{\eta \mid \eta > 0 \text{ y } \mathcal{F}_i[\eta] \text{ es rígido}\}. \quad (7)$$

Luego, h_R es la descomposición que minimiza $\mathcal{D}$ y $\mathcal{L}$ cuando la descomposición es densa, y por lo tanto es la elegida en dichos casos. Es interesante notar que, desde el punto de vista del intercambio de información, $\max_i h_R(i)$ es una medida del grado de descentralización lograda por la descomposición densa. Esto es, si h_R es unitaria, entonces es posible implementar un control de rigidez mediante interacciones entre vecinos únicamente, lo cual representa el mayor grado de descentralización posible. Por otro lado, si h_R es impropia, la descomposición en subframeworks no aporta ventajas respecto de un esquema centralizado.

IV. DESCOMPOSICIÓN DISPERSA

La descomposición dispersa es una modificación del esquema presentado anteriormente, la cual permite extender la familia de descomposiciones válidas. Antes de su presentación formal, el siguiente análisis permitirá comprender la necesidad de su definición.

Dado el grafo rígido de Fig. 2a, la descomposición densa óptima está dada por $h_R(1) = h_R(3) = h_R(4) = 3$ y $h_R(\cdot) = 1$ para el resto. Se puede observar que $\mathcal{F}_1 = \mathcal{F}_3 = \mathcal{F}_4 = \mathcal{F}$. De esta forma, analizar la rigidez de $\mathcal{F}$ mediante la descomposición densa, no presenta ventajas respecto de un enfoque centralizado; pero además se incurre en una gran superposición de subframeworks debido a que, para todo i , $\mathcal{F}_i \subseteq \mathcal{F}_1, \mathcal{F}_3, \mathcal{F}_4$. Esto se evidencia en los altos valores que toman las métricas: $\mathcal{D}(h_R) = 1.5 \cdot \mathcal{D}_{\text{ref}}$ y $\mathcal{L}(h_R) \approx 8.9 \cdot \mathcal{L}_{\text{ref}}(\mathcal{G})$.

Definición 2 (Descomposición dispersa). Sea $\mathcal{F}$ un framework y $h : \mathcal{V} \rightarrow \mathbb{Z}_{\geq 0}$ una función que asocia un alcance (medido en distancia geodésica) a cada nodo. Sea $\mathcal{F}_1, \dots, \mathcal{F}_n$ el conjunto de subframeworks definidos según (4), y $\mathcal{E}_L \subseteq \mathcal{E}$ el conjunto de

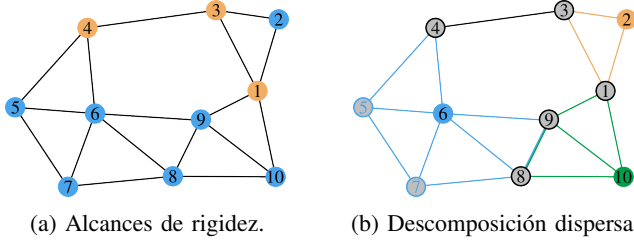


Figura 2: En (a), el color de los nodos según su alcance de rigidez h_R ; en (b) Descomposición dispersa dada por $h(2) = h(6) = h(10) = 1$.

enlaces que no pertenecen a ningún subframework. Entonces, se llama descomposición dispersa al conjunto

$$\text{disp}(\mathcal{F}) = \{\mathcal{F}_i \mid \nexists j : \mathcal{F}_i \subseteq \mathcal{F}_j\} \cup \{\mathcal{L}_\ell \mid \ell \in \mathcal{E}_L\} \quad (8)$$

donde $\mathcal{L}_\ell = (\mathcal{G}_\ell, p)$ y $\mathcal{G}_\ell = (\ell, \{\ell\})$.

Cada descomposición dispersa queda determinada por una función de alcances h , sin embargo, dos funciones distintas podrían inducir la misma descomposición. Como se verá en el resto del trabajo, la descomposición dispersa permite obtener una reducción del número y del tamaño de los subframeworks empleados. De esta forma, se logra reducir el retardo máximo y la carga de comunicaciones, mejorando las características de la descomposición.

Retomando el ejemplo de la Fig. 2a, supongamos que se elige la descomposición dispersa dada por $h(1) = 3$ y $h(\cdot) = 0$ para el resto; es decir $\text{disp}(\mathcal{F}) = \{\mathcal{F}_1\}$. De esta forma se elimina la superposición existente, obteniéndose $\mathcal{D}(h) = 1.5 \cdot \mathcal{D}_{\text{ref}}$ y $\mathcal{L}(h) \approx 4.3 \cdot \mathcal{L}_{\text{ref}}(\mathcal{G})$. No obstante, el retardo máximo permanece constante, como consecuencia de resultar en un esquema centralizado en el nodo $i = 1$, lo cual tampoco es deseable. Además, la carga $\mathcal{L}$ sigue teniendo un valor elevado, producto de que todos los robots (excepto el centro) deben enviar sus tokens de estado a este último, los cuales en algunos casos deben recorrer toda la red. En la Fig. 2 se muestra una descomposición del mismo framework con $h(2) = h(6) = h(10) = 1$, y $h(\cdot) = 0$ para el resto. En este caso, se obtiene $\text{disp}(\mathcal{F}) = \{\mathcal{F}_2, \mathcal{F}_6, \mathcal{F}_{10}, \mathcal{L}_{\{3,4\}}\}$, y las métricas $\mathcal{D}(h) = 0.5 \cdot \mathcal{D}_{\text{ref}}$ y $\mathcal{L}(h) = \mathcal{L}_{\text{ref}}$, es decir, se reducen sustancialmente ambas métricas, incluso alcanzando valores de referencia. Por estas razones, es apropiado analizar exhaustivamente el conjunto de descomposiciones posibles y seleccionar aquellas que minimicen dichas medidas.

IV-A. Rigidez

Para determinar la rigidez de un framework mediante una descomposición dispersa, es necesario reformular el Teorema 1. Para ello definimos la operación K , tal que $K(\mathcal{F})$ es el framework completo inducido por $\mathcal{F}$. Luego, definimos $\mathcal{F}_K$, una versión modificada de $\mathcal{F}$, que surge de reemplazar cada componente $\mathcal{C}$ de $\text{disp}(\mathcal{F})$ por $K(\mathcal{C})$, es decir,

$$\mathcal{F}_K = \bigcup_{\mathcal{C} \in \text{disp}(\mathcal{F})} K(\mathcal{C}). \quad (9)$$

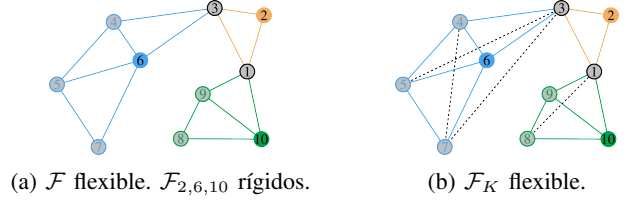


Figura 3: Los subframeworks 2, 6 y 10 son rígidos, pero no están rígidamente vinculados.

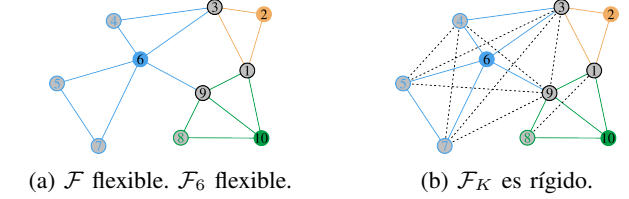


Figura 4: $\mathcal{F}$ es flexible, pero los subframeworks están rígidamente vinculados. La flexibilidad proviene de $\mathcal{F}_6$.

Definición 3 (Vinculación rígida). Sea una descomposición dispersa de $\mathcal{F}$. Decimos que el conjunto $\text{disp}(\mathcal{F})$ está rígidamente vinculado si y sólo si $\mathcal{F}_K$ es rígido.

Esta definición es útil para separar la rigidez por subframeworks en dos condiciones: la primera referida la rigidez de los subframeworks, y la segunda a la forma en la que los mismos están vinculados. Es decir, si un framework es flexible pero todos sus subframeworks rígidos, entonces los mismos no pueden estar rígidamente vinculados. Esto se ilustra en Fig. 3 y Fig. 4. En la primera, se tiene $\mathcal{F}$ flexible (a pesar de que $\mathcal{F}_2, \mathcal{F}_6, \mathcal{F}_{10}$ sean rígidos), y se observa que $\mathcal{F}_K$ también es flexible; por lo que se concluye que los subframeworks no están rígidamente vinculados. En cambio, en la segunda, $\mathcal{F}$ también es flexible, pero $\mathcal{F}_K$ es rígido, es decir, los subframeworks están rígidamente vinculados, por lo que al menos un subframework debe ser flexible.

Teorema 2 (Rigidez por Descomposición Dispersa). $\mathcal{F}$ es rígido si y solo si existe una descomposición $\text{disp}(\mathcal{F})$ tal que son todos sus componentes son rígidos y están rígidamente vinculados.

La prueba de este teorema puede encontrarse en [14, Capítulo 4]. Vale notar que, los componentes dados por los subframeworks triviales y por los enlaces vinculantes, son rígidos por definición. El siguiente lema, que es consecuencia del teorema anterior, es útil para el estudio de la descomposición óptima que presentamos en la siguiente subsección.

Lema 1. Si $\mathcal{F}$ es rígido, entonces para toda descomposición, los componentes asociados están rígidamente vinculados.

IV-B. Descomposición Óptima

Como ya se presentó, es importante abordar el problema de elegir, dado un framework rígido, una descomposición en subframeworks óptima, teniendo en cuenta las métricas de desempeño $\mathcal{D}, \mathcal{L}$. Esta descomposición, para que permita

establecer la rigidez del framework, deberá cumplir las condiciones del Teorema 2. No obstante, gracias al Lema 1, para descomponer un framework rígido no es necesario evaluar la condición de vinculación rígida.

En el caso de descomposición densa, se vio que el alcance mínimo de rigidez (h_R) minimiza ambas funciones. Sin embargo, en el esquema disperso, el conjunto de funciones h posibles es estrictamente mayor al anterior. Surge naturalmente entonces el problema de seleccionar una función de alcance de forma óptima en este nuevo conjunto. En primer lugar, es deseable elegir h tal que el retardo máximo sea menor o igual que el diámetro del grafo, por lo que imponemos la condición $\mathcal{D}(h) \leq \mathcal{D}_{\text{ref}}$. En segundo lugar, queremos minimizar la carga comunicacional, por lo que es propicio emplear $\mathcal{L}$ como un costo a minimizar.

Sin embargo, notamos que la descomposición trivial alcanza el mínimo global de ambas medidas. Esto no es útil desde el punto de vista del control, dado que las condiciones de rigidez por subframeworks del Teorema 2 (que son las que el controlador debe monitorear), equivalen a la rigidez de todo el framework. En otras palabras, los enlaces vinculantes presentan una dificultad para el controlador, ya que no hay subframework que los supervise y decida si se puede perder o no. Debe plantearse otra estrategia para tratar con los mismos; como caso más conservador, mantenerlos en todo momento.

Por este motivo, agregamos a la función de costo un término que incluye a la cantidad de enlaces vinculantes m_ℓ existentes en una descomposición, de forma tal de mantenerla acotada. Luego, definimos el alcance óptimo

$$h^* = \underset{h \in \mathcal{H}}{\text{argmín}} \mathcal{L}(h) + 2m_\ell, \quad (10)$$

donde el espacio de búsqueda es

$$\mathcal{H} = \{h \mid \forall \mathcal{C} \in \text{disp}(\mathcal{F}) : \mathcal{C} \text{ es rígido, y } \mathcal{D}(h) \leq \mathcal{D}_{\text{ref}}\}. \quad (11)$$

La búsqueda exhaustiva de h^* , no resulta un abordaje escalable con el número de robots n . Esto se debe a que debe evaluarse la rigidez de todos sus componentes para cada una de las $(\mathcal{D}_{\text{ref}} + 1)^n$ descomposiciones posibles.

Para solucionar esto, proponemos un algoritmo *greedy* que parte de h_R , y en cada iteración, lleva a cero el alcance del subframework que provoca el mayor decrecimiento de la función de costo. Llamamos $\hat{h}^*$ a la descomposición subóptima obtenida. El procedimiento se muestra en el Algoritmo 1, que aproxima el valor de h^* buscando entre no más $n(n+1)/2$ alcances distintos.

V. RESULTADOS

En Fig. 5 y Fig. 6 se muestran dos ejemplos del alcance de rigidez obtenido y la aplicación del Algoritmo 1 para obtener el alcance subóptimo, y las métricas que lo acompañan. En ambos casos, puede observarse cómo la descomposición dispersa es capaz de capturar la rigidez del framework a partir de un conjunto reducido de componentes, eliminando aquellos de

Algoritmo 1: Búsqueda de descomposición óptima.

Input: $\mathcal{F}$ (framework rígido), h_R (alcance mínimo de rigidez)

Output: $\hat{h}^*$ (alcance subóptimo)

$\hat{h}^* \leftarrow h_R$;

remain $\leftarrow \{1, \dots, n\}$;

terminar $\leftarrow \text{False}$;

mín $\leftarrow \infty$;

while not terminar **do**

collapse $\leftarrow \text{None}$;

for i **in** remain **do**

$g \leftarrow \hat{h}^*$;

$g(i) \leftarrow 0$;

carga $\leftarrow \mathcal{L}(g)$;

if carga $<$ mín **then**

mín $\leftarrow$ carga;

collapse $\leftarrow i$;

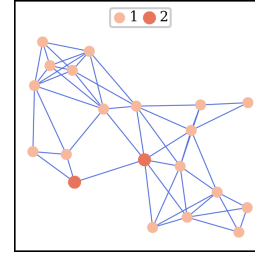
if collapse **is not** None **then**

$\hat{h}^*(\text{collapse}) \leftarrow 0$;

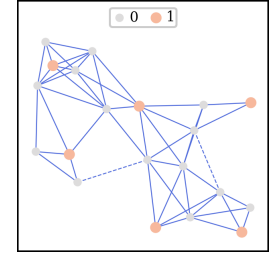
remain $\leftarrow$ remain $- \{i\}$;

else

terminar $\leftarrow \text{True}$



(a) $\mathcal{D} = 4, \mathcal{L} = 730$



(b) $\mathcal{D} = 2, \mathcal{L} = 100, m_\ell = 2$.

Figura 5: (a) descomposición densa vs. (b) dispersa subóptima para un grafo de 20 nodos con $\mathcal{D}_{\text{ref}} = 5$, $\mathcal{L}_{\text{ref}} = 100$. En color y tamaño: los alcances; en línea punteada: los enlaces.

mayor alcance y disminuyendo notablemente la superposición de los mismos. Esto se evidencia en los valores de las métricas que se reducen considerablemente, y manteniendo reducido el número de enlaces vinculantes.

Por otro lado, en Fig. 7 y Fig. 8 se observan los valores típicos de los parámetros de interés para la descomposición densa y dispersa, respectivamente. Los resultados fueron generados para un conjunto de 10 frameworks por cada valor de $3 \leq n \leq 100$. Podemos observar cómo en el caso del retardo máximo, el valor de $\mathcal{D}$ se vuelve prácticamente constante e igual a 2 en el caso disperso, indicando que en promedio, no hay subframeworks de tamaño mayor a 1. En el caso de la carga de comunicaciones, vemos que se logra que se mantenga prácticamente igual al valor de referencia, manteniendo reducido el porcentaje de enlaces vinculantes.

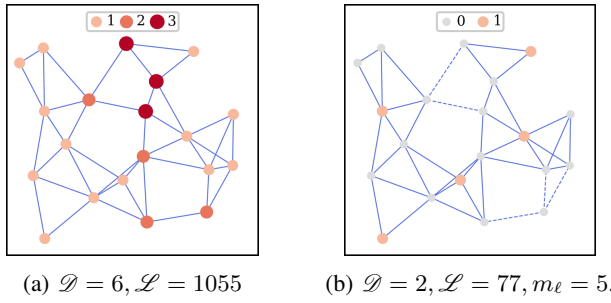


Figura 6: (a) descomposición densa vs. (b) dispersa subóptima para un grafo de 20 nodos con $\mathcal{D}_{\text{ref}} = 5$, $\mathcal{L}_{\text{ref}} = 80$.

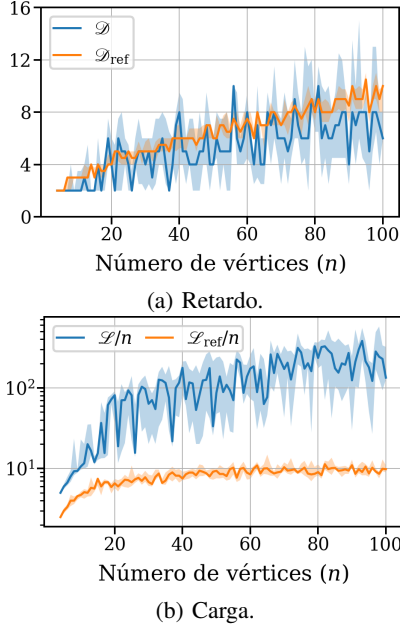


Figura 7: Comparación de las métricas típicas de la descomposición dispersa vs. los valores de referencia.

VI. CONCLUSIÓN

Este trabajo permitió comparar cualitativa y cuantitativamente distintos tipos de descomposición, aportando a una mejor comprensión este problema. A partir del análisis presentado, se establecieron los criterios de optimalidad y los algoritmos que permitieron seleccionar descomposiciones óptimas aproximadas. Mediante simulaciones computacionales, se pudo observar las claras ventajas de la descomposición dispersa respecto de la densa, situando a la primera como la elegida para futuras investigaciones.

REFERENCIAS

- [1] Y. Cao, W. Yu, W. Ren, and G. Chen, "An overview of recent progress in the study of distributed multi-agent coordination," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 1, pp. 427–438, 2013.
- [2] J. K. Verma and V. Ranga, "Multi-robot coordination analysis, taxonomy, challenges and future scope," *Journal of Intelligent & Robotic Systems*, vol. 102, 2021. [Online]. Available: <https://api.semanticscholar.org/CorpusID:233256952>

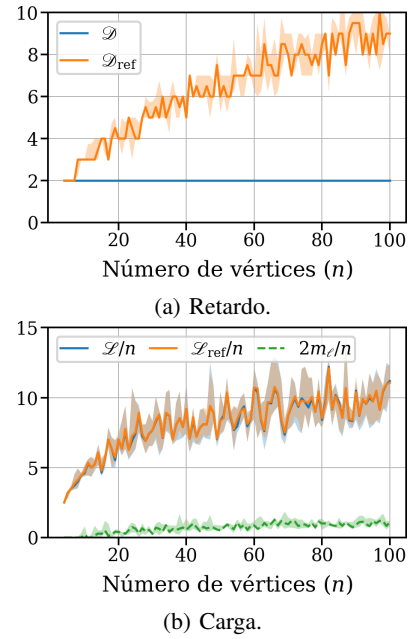


Figura 8: Comparación de las métricas típicas de la descomposición dispersa vs. los valores de referencia. La línea sólida: mediana, área coloreada cuartiles Q_1 (25 %) y Q_3 (75 %).

- [3] J. Aspnes, T. Eren, D. K. Goldenberg, A. S. Morse, W. Whiteley, Y. R. Yang, B. D. O. Anderson, and P. N. Belhumeur, "A theory of network localization," *IEEE Transactions on Mobile Computing*, vol. 5, no. 12, pp. 1663–1678, Dec. 2006.
- [4] B. D. O. Anderson, B. Fidan, C. Yu, and D. Walle, "Uav formation control: Theory and application," in *Recent Advances in Learning and Control*, V. D. Blondel, S. P. Boyd, and H. Kimura, Eds. London: Springer London, 2008, pp. 15–33.
- [5] D. Zelazo, A. Franchi, P. Allgöwer, H. Bühlhoff, and P. Giordano, "Rigidity maintenance control for multi-robot systems," in *2012 Robotics: Science and Systems*, 2012.
- [6] D. Zelazo, A. Franchi, H. H. Bühlhoff, and P. R. Giordano, "Decentralized rigidity maintenance control with range measurements for multi-robot systems," *The International Journal of Robotics Research*, vol. 34, no. 1, pp. 105–128, 2015.
- [7] Z. Sun, C. Yu, and B. D. O. Anderson, "Distributed optimization on proximity network rigidity via robotic movements," in *2015 34th Chinese Control Conference (CCC)*, 2015, pp. 6954–6960.
- [8] R. K. Williams, A. Gasparri, A. Priolo, and G. S. Sukhatme, "Distributed combinatorial rigidity control in multi-agent networks," in *52nd IEEE Conference on Decision and Control*, 2013, pp. 6061–6066.
- [9] A. M. Amani, G. Chen, M. Jalili, X. Yu, and L. Stone, "Distributed rigidity recovery in distance-based formations using configuration lattice," *IEEE Transactions on Control of Network Systems*, vol. 7, no. 3, pp. 1547–1558, 2020.
- [10] J. M. Hendrickx, R. M. Jungers, A. Olshevsky, and G. Vankeerberghen, "Graph diameter, eigenvalues, and minimum-time consensus," *Automatica*, vol. 50, no. 2, pp. 635–640, 2014.
- [11] J. F. Presenza, J. I. Alvarez-Hamelin, I. Mas, and J. I. Giribet, "Subframework-based rigidity control in multirobot networks," in *American Control Conference, Atlanta, Georgia, USA*, 2022.
- [12] J. F. Presenza, J. I. Giribet, J. I. Alvarez-Hamelin, and I. Mas, "Control de rigidez por subframeworks en sistemas multirobot," in *Jornadas Argentinas de Robótica, Bariloche, Argentina*, 2022, 2022.
- [13] L. Asimow and B. Roth, "The rigidity of graphs, ii," *Journal of Mathematical Analysis and Applications*, vol. 68, pp. 171–190, 1979.
- [14] J. F. Presenza, "Redes reconfigurables de vehículos no tripulados," Ph.D. dissertation, Universidad de Buenos Aires, 2023.

Sesión 10: Interacción Humano-Robot

Diseño y desarrollo de un prototipo funcional de exoesqueleto para asistencia en neurorehabilitación de miembro superior

Solana Lopez Agüero

Universidad Nacional de Entre Ríos
Facultad de Ingeniería

Email: solana.lopez@ingenieria.uner.edu.ar

Rosa Weisz

Universidad Nacional de Entre Ríos
Facultad de Ingeniería

Email: rweisz@ingenieria.uner.edu.ar

Leandro Mayrata

Universidad Nacional de Entre Ríos
Facultad de Ingeniería

Email: lmayrata@ingenieria.uner.edu.ar

Abstract—Este estudio presenta el diseño, implementación y evaluación de un prototipo de exoesqueleto de miembro superior de bajo costo destinado a ayudar a personas con discapacidades motoras, especialmente a personas que han sufrido un accidente cerebrovascular, en Argentina. El exoesqueleto utiliza una combinación de un sistema de motor-reductor simple y componentes impresos en 3D para garantizar accesibilidad y replicabilidad. Se implementaron bucles de control para regular los movimientos articulares y garantizar una asistencia segura y efectiva. Las pruebas piloto con voluntarios demostraron el potencial del exoesqueleto para ayudar en el movimiento del miembro superior, aunque con áreas notables de mejora en cuanto a comodidad, ergonomía y adaptabilidad del usuario. Las sugerencias de los voluntarios proporcionan información valiosa para futuras iteraciones.

I. INTRODUCCIÓN

En el contexto mundial de la salud y la rehabilitación, la Organización Mundial de la Salud (OMS) lanzó la iniciativa Rehabilitación 2030 para abordar la significativa necesidad insatisfecha de servicios de rehabilitación en todo el mundo y resaltar la importancia de fortalecer los sistemas de salud para ofrecer estos servicios de manera efectiva [1]. Esta iniciativa representa un enfoque estratégico innovador para la comunidad global de rehabilitación, enfatizando que la rehabilitación debe ser accesible para todas las personas y en todas las etapas de la vida.

El envejecimiento es un fenómeno global que se asocia con un aumento en la incidencia de diversas enfermedades, como los accidentes cerebrovasculares (ACV), enfermedades tumorales y trastornos musculoesqueléticos, entre otros. Estas condiciones pueden resultar en discapacidades en los miembros superiores, afectando la calidad de vida de las personas. La OMS proyecta un aumento significativo en la población mundial de personas mayores de 50 años para 2050.

El incremento en la población de edad avanzada requiere estrategias de rehabilitación más efectivas y accesibles. Se ha demostrado que la neuroplasticidad juega un papel crucial en la recuperación sensoriomotora después de lesiones en la corteza motora del Sistema Nervioso Central (SNC) derivadas de ACV [2]. En este contexto, se busca recuperar la activación fisiológica de las unidades motoras del miembro afectado me-

diante movimientos funcionales y estimulación de receptores específicos.

Una estrategia de rehabilitación que ha demostrado ser efectiva es el uso de exoesqueletos mecánicos vinculados al miembro a rehabilitar [3,4]. Estos dispositivos utilizan actuadores y sensores para generar y registrar movimientos, respectivamente. Estas terapias tienen como objetivo mejorar el rango de movimiento, control del tono muscular, coordinación motora y promover el aprendizaje motor [5,6].

Los exoesqueletos activos proporcionan un contexto estandarizado pero adaptable a cada paciente, permitiendo un alto nivel de repetibilidad y precisión del movimiento. En cuanto a la rehabilitación del miembro superior, se reconoce como un proceso que requiere una integración compleja de las actividades musculares de las partes proximal y distal [7].

Aunque aún se necesita más investigación sobre los mecanismos de neuroplasticidad y los programas de rehabilitación con exoesqueletos, existe consenso en que la participación activa del paciente y la duración de la terapia son factores importantes para obtener resultados positivos [8,9]. Dosificaciones bajas pueden no ser efectivas [10], mientras que dosis más altas han demostrado mejoras clínicamente significativas [11,12].

El creciente interés en estrategias de rehabilitación eficientes y accesibles, junto con la evidencia de los beneficios de los exoesqueletos en la rehabilitación de miembros superiores, justifica el desarrollo y aplicación continua de estos dispositivos en el ámbito clínico.

II. METODOS

La metodología empleada abarcó diversas etapas para el desarrollo y evaluación del exoesqueleto de miembro superior. En primer lugar, se realizó una revisión exhaustiva de las técnicas utilizadas en la rehabilitación del miembro superior en casos clínicos, así como observaciones en el campo para comprender los requerimientos del prototipo. Posteriormente, se procedió con la implementación del prototipo, abordando tanto su diseño mecánico como electrónico, incluyendo los bucles de control esenciales.

El diseño del exoesqueleto se basó en criterios ergonómicos y de movilidad, asegurando su adaptabilidad y facilidad de uso para el paciente. Se tomaron en consideración elementos clave como el motor (servomotor DSS-M15S[13]), la celda de carga de 5 kg y el microcontrolador Arduino Nano, junto con otros componentes necesarios para el funcionamiento del dispositivo. Además, se integraron bucles de control que permitieron una interacción efectiva entre el exoesqueleto y el usuario.

Para evaluar el diseño del exoesqueleto, se llevaron a cabo pruebas de laboratorio exhaustivas, incluyendo pruebas de carga y resistencia, así como mediciones de ángulo para garantizar la precisión y seguridad del dispositivo. Además, se realizaron pruebas piloto con voluntarios sanos para evaluar la eficacia clínica del exoesqueleto, considerando aspectos como la comodidad, facilidad de uso y mejora en la capacidad para realizar tareas específicas.

III. RESULTADOS

La evaluación del exoesqueleto de miembro superior proporcionó una visión detallada de su funcionamiento, rendimiento y seguridad, respaldada por pruebas de laboratorio y piloto.

A. Diseño Final

El diseño mecánico del exoesqueleto desempeña un papel fundamental al asegurar que el dispositivo se sujete al brazo del paciente de manera cómoda y efectiva, transmitiendo las fuerzas de manera armónica y sin causar molestias. Es crucial considerar los requisitos del usuario, como el peso, la comodidad, la estética y el coste, ya que estos factores están estrechamente ligados al éxito de los dispositivos de rehabilitación.

Para abordar estos desafíos, se propone una innovación mediante la aplicación de procesos de fabricación rápidos y adaptables, como la impresión tridimensional (3D). Esta técnica de prototipado es rápida, práctica y rentable, permitiendo la creación de estructuras complejas que serían difíciles o costosas de producir con métodos tradicionales como el mecanizado.

Basándonos en el análisis de pruebas realizadas sobre tres materiales que se presenta en un estudio [13], se concluyó que el PLA demostró una mayor resistencia a la tracción y una dureza razonable en comparación con materiales similares como el PETG y el ABS. Por esta razón, se optó por imprimir el exoesqueleto en PLA.

Un aspecto fundamental en el desarrollo del exoesqueleto fue asegurar su adaptación al brazo del paciente para evitar molestias o lesiones. Para lograr este objetivo, se llevaron a cabo las siguientes etapas:

1. Evaluación de Requisitos del Usuario: Se recopilaron datos detallados sobre las dimensiones y características físicas de los usuarios potenciales, incluyendo la longitud, diámetro y flexibilidad del brazo, así como consideraciones médicas específicas.

2. Modelo EduExo: En las etapas iniciales del proyecto, se utilizó el modelo de código abierto "EduExo" para realizar

pruebas integrales de motores, celdas de carga, bucles de control y la interfaz con el usuario. Estas pruebas fueron fundamentales para el diseño posterior del exoesqueleto.

3. Rediseño y Adaptación: Basándose en los resultados de las pruebas iniciales, se procedió a rediseñar el exoesqueleto para que se ajustara mejor a las especificaciones ergonómicas requeridas. Se realizaron modificaciones en la forma, tamaño y disposición de los componentes para lograr una ergonomía óptima.

El diseño final del exoesqueleto se desarrolló utilizando el programa "Fusión", optimizado para cumplir con los objetivos de comodidad, ajustabilidad y movilidad natural del usuario en el contexto de la neurorehabilitación del miembro superior. Tanto el diseño mecánico como electrónico se centraron en la ergonomía y funcionalidad, asegurando una integración efectiva de los componentes electrónicos con la estructura mecánica. A continuación se presenta una imagen del prototipo final del exoesqueleto, demostrando la exitosa integración de los componentes electrónicos con la estructura mecánica del dispositivo.



Fig. 1. Imagen del prototipo final. Fuente: Fotografía tomada por el autor, 2023.

B. Electronica y Componentes

El diseño e integración del exoesqueleto se fundamentó en una variedad de componentes electrónicos clave. El microcontrolador Arduino Nano actuó como el centro de control del dispositivo, coordinando las operaciones y procesando la información proveniente de los sensores. Por otro lado, el servomotor DSS-M15S, equipado con un potenciómetro integrado, proporcionó el torque necesario para ejecutar los movimientos del exoesqueleto, permitiendo un posicionamiento altamente preciso gracias a su capacidad para medir el ángulo de rotación (posición) del eje del motor. La celda de carga de 5 kg permitió la medición y control precisos de la fuerza aplicada, mientras que se utilizó el amplificador HX711 para mejorar la precisión de las mediciones de carga. Se puede observar a continuación el esquema de los componentes.

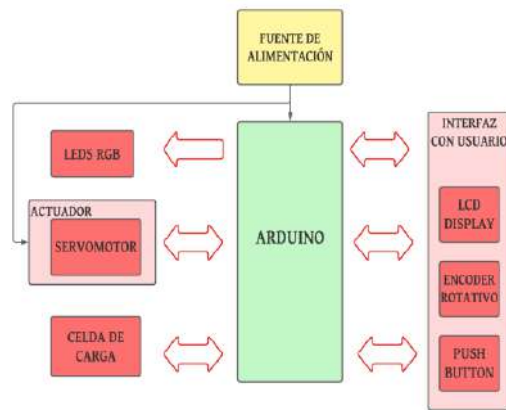


Fig. 2. Gráfico ilustrativo de componentes. Fuente: Fotografía tomada por el autor, 2023.

C. Bucles de Control

Los bucles de control desempeñaron un papel crucial en la operación dinámica del exoesqueleto, permitiendo una interacción adaptativa entre el dispositivo y el usuario. Se implementaron tres tipos de bucles de control para optimizar la terapia a medida que el paciente progresa en su rehabilitación:

- Control de Posición: Este bucle aseguró que el exoesqueleto moviera las articulaciones con precisión a posiciones predefinidas, lo cual es especialmente vital cuando el usuario no puede controlar sus movimientos por sí mismo. Proporcionó una movilidad controlada y segura en ausencia de la habilidad activa del usuario para garantizar una terapia efectiva, particularmente en las etapas iniciales de rehabilitación cuando el paciente puede experimentar espasticidad o necesita reaprender movimientos naturales de las articulaciones. La implementación del bucle se basa en que el servomotor sigue una trayectoria definida y el potenciómetro verifica que no se desvíe de dicha trayectoria.

- Control de Impedancia: En un sistema asistivo, el control de impedancia permitió al exoesqueleto interactuar de manera adaptativa con el usuario, ajustando la resistencia y la respuesta del sistema según las acciones y la intención del usuario. Adaptado a la fuerza y posición del paciente, este control proporcionó asistencia proporcional y natural, mejorando la comodidad y eficacia de la terapia. Este sistema acompaña la mejora del paciente a medida que logra realizar movimientos por voluntad propia. El control de impedancia se basa en la implementación de sensor de fuerza y servomotor; cuando se detecta una fuerza ejercida por el paciente o una intención de movimiento, el motor se mueve proporcionalmente a esa fuerza detectada.

- Control de Fuerza: En un sistema resistivo, el control de fuerza desafió al usuario con una resistencia controlada que debía superarse activamente. Es esencial para proporcionar una carga adicional en los movimientos, beneficiando el fortalecimiento muscular, la rehabilitación y el entrenamiento de fuerza. El paciente trabaja contra esta resistencia para

lograr sus objetivos terapéuticos, típicamente en las etapas más avanzadas de rehabilitación. La selección específica de cada tipo de control (posición, impedancia, fuerza) se alinea con las funciones y objetivos de diferentes sistemas de exoesqueletos, abordando las necesidades del paciente a lo largo de su proceso de recuperación.

D. Pruebas

Se llevaron a cabo pruebas exhaustivas en un entorno de laboratorio controlado para evaluar el funcionamiento y la resistencia del exoesqueleto. Estas pruebas incluyeron:

- Prueba de Carga y Resistencia: Se aplicó una carga gradual al exoesqueleto para evaluar su capacidad para soportar el peso sin deformaciones ni fallos. Se registraron datos sobre el torque generado y la respuesta del dispositivo a diferentes cargas.

- Prueba de Medición de Ángulo: Se utilizó un goniómetro para medir y evaluar con precisión los ángulos de movimiento permitidos por el exoesqueleto. Se compararon los ángulos medidos con los ángulos deseados para determinar la precisión del dispositivo.

- Pruebas de Confort y Eficacia en Voluntarios Sanos: Se realizaron pruebas para evaluar la eficacia y el confort del exoesqueleto. El grupo de voluntarios estaba compuesto por 6 mujeres sanas, con edades entre 20 y 25 años. En esta fase, se llevaron a cabo pruebas piloto para evaluar la seguridad y eficacia clínica del exoesqueleto en términos de adherencia al tratamiento y fatiga percibida por los voluntarios durante su uso. Estas pruebas consistieron en ensayos con un número reducido de voluntarios sanos y estudios de viabilidad iniciales.



Fig. 3. Prueba de Confort y Eficacia en Voluntarios Sanos. Fuente: Fotografía tomada por el autor, 2023.

1) *Resultados Graficos:* Las pruebas proporcionaron datos cuantitativos que se representaron gráficamente para una mejor comprensión. A continuación, se presentan algunas de las gráficas obtenidas durante las pruebas:

Gráfico de relación entre ángulos del servomotor y ángulos medidos con el goniómetro. Fig.4.

Otros gráficos que muestran la respuesta del exoesqueleto a diferentes cargas y condiciones de funcionamiento Fig.5.

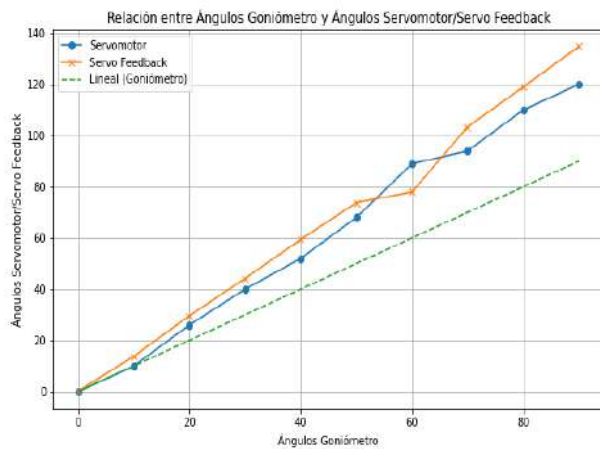


Fig. 4. Gráfica de la relación entre los ángulos medidos del goniómetro, feedback del servomotor y el ángulo real del servomotor.

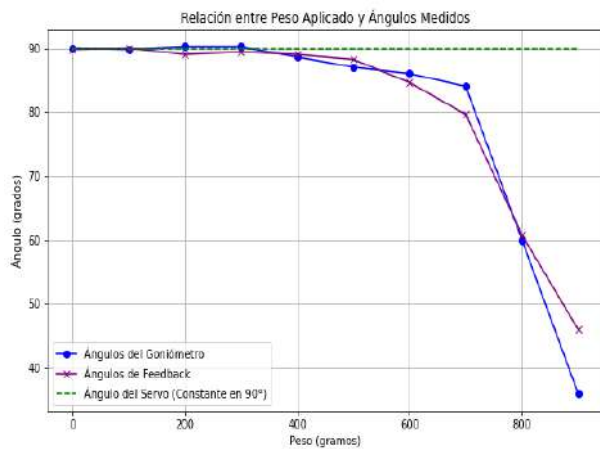


Fig. 5. Gráfico que relaciona el peso aplicado con los ángulos medidos.

Estas gráficas proporcionaron una visualización clara del rendimiento del exoesqueleto en diferentes situaciones, validando su precisión, seguridad y eficacia en la asistencia al movimiento del miembro superior.

Además se puede observar la gráfica obtenida luego de que los usuarios hicieron las pruebas obteniéndose una retroalimentación de confort y funcionalidad del mismo. Fig. 6.

IV. DISCUSION

Este dispositivo fue diseñado con el objetivo principal de ser accesible y fácilmente replicable en el mercado argentino. En esta sección, se analizan las justificaciones detrás de las elecciones de diseño y componentes, así como las lecciones aprendidas durante el proceso de desarrollo.

A. Consideraciones sobre el Diseño:

La elección del motor y la galga se basó en criterios fundamentales, como su bajo costo y disponibilidad en el mercado argentino. Sin embargo, es importante reconocer que estas elecciones podrían limitar el rendimiento del exoesqueleto. Alternativas con motores de mayor torque podrían mejorar la

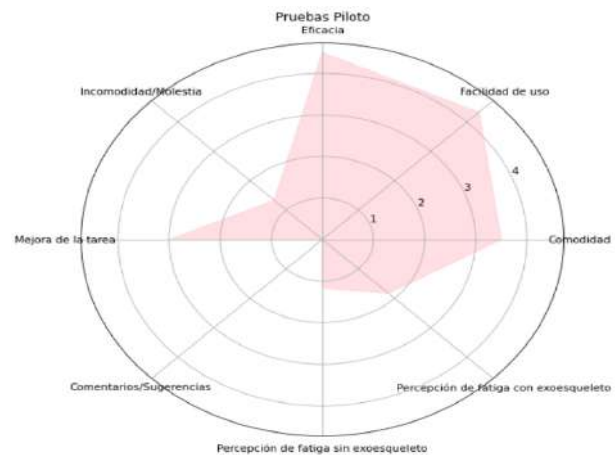


Fig. 6. Gráfica de respuestas obtenidas en un cuestionario a voluntarios sanos sobre confort y funcionalidad.

eficacia del dispositivo, pero implicarían un aumento en los costos de producción. Se seleccionó la impresión 3D como método de fabricación debido a su asequibilidad, capacidad de replicación y su capacidad para reducir significativamente el peso y el tamaño del exoesqueleto. Esto distingue al prototipo de soluciones comerciales, que suelen ser voluminosas, pesadas y costosas.

B. Consideraciones sobre Pruebas:

Consideraciones sobre las Pruebas: Luego de la primera prueba de medición de ángulos, se implementó una corrección mediante regresión lineal en el feedback del servomotor para alinear este ángulo con la posición deseada del servomotor. Se observó que la diferencia absoluta promedio entre el ángulo del servomotor y el ángulo del goniómetro tomado como referencia es aproximadamente 4.72° , representando un 10.49% de esta diferencia respecto al valor medio de los ángulos del goniómetro. Estos resultados indican la discrepancia promedio entre las mediciones realizadas por el servomotor y el goniómetro en los ángulos evaluados.

En la segunda prueba, se observó que a medida que aumenta la carga, las variaciones en los errores entre los ángulos medidos tanto por el servomotor como por el feedback incrementan significativamente. Para cargas más ligeras (por ejemplo, 0 gramos y 100 gramos), la diferencia entre el ángulo medido por el servomotor y el ángulo del goniómetro es mínima, en el rango de $\pm 0.2^\circ$. Sin embargo, a medida que la carga aumenta (especialmente a partir de 500 gramos), esta diferencia se amplía, alcanzando hasta aproximadamente 5.4° en el peor caso (700 gramos).

Además, para el ángulo medido en el goniómetro, las diferencias con respecto a los 90° (posición deseada del servo) varían desde -89.8° hasta -44° a medida que aumenta la carga. Para el ángulo del feedback del servomotor, las diferencias con respecto a los 90° varían desde -90° hasta -54° a medida que aumenta la carga.

En términos porcentuales, las diferencias con respecto a los 90° son negativas y varían desde aproximadamente -0.22%

hasta -100% para el ángulo medido en el goniómetro, y desde aproximadamente -0.22% hasta -60% para el ángulo del feedback del servomotor, a medida que aumenta la carga.

Estos cálculos reflejan cómo las diferencias porcentuales indican la desviación de los ángulos medidos respecto al valor de referencia de 90° bajo diferentes condiciones de carga, lo que muestra la variación relativa en la precisión del sistema comparada con el ángulo deseado.

A pesar de las limitaciones del motor seleccionado, con un torque nominal de $1.57 Nm$ y el torque aproximado del exoesqueleto de $1.4 Nm$, los resultados de las pruebas piloto indican el potencial del dispositivo en la asistencia al movimiento del miembro superior. Sin embargo, también resaltan áreas críticas de mejora, especialmente en términos de comodidad, ergonomía y adaptabilidad a diferentes usuarios.

Las sugerencias de los voluntarios proporcionan una base sólida para futuras iteraciones y mejoras en el diseño del exoesqueleto. Estos resultados respaldan la importancia de la retroalimentación de los usuarios en el desarrollo de tecnologías de asistencia y rehabilitación.

V. CONCLUSION

Se logró satisfactoriamente sus objetivos al diseñar, implementar y evaluar exitosamente el prototipo de exoesqueleto de miembro superior. Se llevó a cabo de manera integral, permitiendo la realización de pruebas con voluntarios y el análisis de los resultados obtenidos. El trabajo se benefició enormemente de las diversas perspectivas y opiniones aportadas por diferentes profesionales, lo que desempeñó un papel vital en todas las etapas del desarrollo del proyecto. El éxito se basó en la colaboración y el esfuerzo conjunto de varios individuos, así como en el compromiso continuo con la mejora y la innovación en el campo de la tecnología de asistencia y rehabilitación. Se reconoce la posibilidad de obtener mejores resultados explorando alternativas de motores con mayor torque y refinando los bucles de control para optimizar el rendimiento del exoesqueleto.

REFERENCES

- [1] Organización Mundial de la Salud, Iniciativa Rehabilitación 2030, techreport, s.f. <https://www.who.int/initiatives/rehabilitation-2030>
- [2] P.-P. Han, Y. Han, X.-Y. Shen, Z.-K. Gao, and X. Bi, Enriched environment-induced neuroplasticity in ischemic stroke and its underlying mechanisms, *Frontiers in Cellular Neuroscience*, vol. 17, jul 2023.
- [3] N. Singh, M. Saini, N. Kumar, M. V. P. Srivastava, and A. Mehndiratta, Evidence of neuroplasticity with robotic hand exoskeleton for post-stroke rehabilitation: a randomized controlled trial, *Journal of NeuroEngineering and Rehabilitation*, vol. 18, no. 1, may 2021.
- [4] R. G. Braun and G. F. Wittenberg, Motor recovery: How rehabilitation techniques and technologies can enhance recovery and neuroplasticity, *Seminars in Neurology*, vol. 41, no. 02, pp. 167–176, mar 2021.
- [5] F. Posteraro, S. Mazzoleni, S. Aliboni, B. Cesqui, A. Battaglia, P. Dario, and S. Micera, Robot-mediated therapy for paretic upper limb of chronic patients following neurological injury, *Journal of Rehabilitation Medicine*, vol. 41, no. 12, pp. 976–980, 2009.
- [6] S. Bhujel and S. Hasan, A comparative study of end-effector and exoskeleton type rehabilitation robots in human upper extremity rehabilitation, *Human-Intelligent Systems Integration*, jun 2023.
- [7] P. Langhorne, J. Bernhardt, and G. Kwakkel, Stroke rehabilitation, *The Lancet*, vol. 377, no. 9778, pp. 1693–1702, may 2011.

- [8] J. van Kordelaar, E. van Wegen, and G. Kwakkel, Impact of time on quality of motor control of the paretic upper limb after stroke, *Archives of Physical Medicine and Rehabilitation*, vol. 95, no. 2, pp. 338–344, feb 2014.
- [9] K. J. Waddell, R. L. Birkenmeier, J. L. Moore, T. G. Hornby, and C. E. Lang, Feasibility of high-repetition, task-specific training for individuals with upper-extremity paresis, *The American Journal of Occupational Therapy*, vol. 68, no. 4, pp. 444–453, jul 2014.
- [10] C. E. Lang, M. J. Strube, M. D. Bland, K. J. Waddell, K. M. Cherry-Allen, R. J. Nudo, A. W. Dromerick, and R. L. Birkenmeier, Dose response of task-specific upper limb training in people at least 6 months poststroke: A phase II, single-blind, randomized, controlled trial, *Annals of Neurology*, vol. 80, no. 3, pp. 342–354, aug 2016.
- [11] J. McCabe, M. Monkiewicz, J. Holcomb, S. Pundik, and J. J. Daly, Comparison of robotics, functional electrical stimulation, and motor learning methods for treatment of persistent upper extremity dysfunction after stroke: A randomized controlled trial, *Archives of Physical Medicine and Rehabilitation*, vol. 96, no. 6, pp. 981–990, jun 2015.
- [12] Y. Iwamoto, R. Tanaka, T. Imura, T. Mitsutake, H. Jung, T. Suzukawa, S. Taki, N. Imada, T. Inagawa, H. Araki, and O. Araki, Does frequent use of an exoskeletal upper limb robot improve motor function in stroke patients?, *Disability and Rehabilitation*, vol. 45, no. 7, pp. 1185–1191, mar 2022.
- [13] Martínez-García, A. B., Valdez-Castillo, M. A., Martínez-García, M. A., & Cárdenas-Campos, A. A. (2021). Mechanical Properties of 3D-Printed Parts Made of Polyethylene Terephthalate Glycol. *Journal of Materials Engineering and Performance*. URL: <https://link.springer.com/article/10.1007/s11665-021-06032-4>.

Control de Admitancia con Modulación de Impedancia para un Bastón Robotizado

Jeysson Tapia Mullo
Instituto de Automática
UNSJ - CONICET
San Juan - Argentina
jtapia@inaut.unsj.edu.ar

Juan Carlos Iturrieta
Instituto de Bioingeniería
UNSJ
San Juan - Argentina
jciturrieta@inbio.unsj.edu.ar

Fernando Tettamanti
Instituto de Bioingeniería
UNSJ
San Juan - Argentina
ftettamanti@inbio.unsj.edu.ar

Alejandro Rodrigo
Instituto de Bioingeniería
UNSJ
San Juan - Argentina
arodrigo@inbio.unsj.edu.ar

Flavio Roberti
Instituto de Automática
UNSJ - CONICET
San Juan - Argentina
froberti@inaut.unsj.edu.ar

Ricardo Carelli
Instituto de Automática
UNSJ - CONICET
San Juan - Argentina
rcarelli@inaut.unsj.edu.ar

Resumen—Actualmente, la robótica se ha integrado en todos los ámbitos de la vida, siendo la salud uno de los principales, brindando soluciones prácticas, novedosas y seguras, que conllevan a que las personas tengan un mejor estilo de vida. En el presente trabajo se presenta el desarrollo un controlador de admitancia con compensación dinámica y se analiza las ventajas de adicionar la modulación de impedancia al controlador. Las estrategias de control son aplicadas en el bastón robotizado desarrollado por el Instituto de Automática, con la finalidad de brindar asistencia a personas con alteraciones en la distribución del peso entre un miembro y el otro. Las respuestas se contrastan mediante un protocolo de pruebas, resaltando el efecto positivo causado por la modulación de impedancia, al obtener respuestas en velocidad tanto lineal y angular más uniformes a lo largo del trayecto. Este trabajo se encuadra en el área de la robótica ya que un bastón automático es un dispositivo robótico que está en interacción con personas mediante contacto.

Index Terms—Robótica de asistencia, bastón robotizado, control de admitancia, modulación de impedancia.

Sesión 11: Robótica Educativa

AelE: una herramienta para la enseñanza de programación basada en Arduino

Gonzalo Pablo Fernández
Departamento de Ciencia y Tecnología
Universidad Nacional de Quilmes
Bernal, Buenos Aires, Argentina
gpfernandez@dc.uba.ar

Christian Cossio-Mercado
Departamento de Computación
Universidad de Buenos Aires
Buenos Aires, Argentina
ccossio@dc.uba.ar

Resumen—La enseñanza de programación aporta al pensamiento abstracto y a la resolución de problemas, entre otros aspectos. Adicionalmente, la utilización de programación de placas *Arduino*, con su diseño de hardware abierto y multiplicidad de recursos disponibles, permite trabajar con programación con salidas físicas, además de facilitar el trabajo con proyectos. Existen muchas herramientas para el trabajo con programación en el contexto escolar, como *Scratch* y *PilasBloques*. En particular, para *Arduino* se suelen utilizar *mBlock* y *Scratch*, además de otras herramientas específicas de kits de robótica *Arduino*-compatibles. Las herramientas disponibles para el trabajo de programación con *Arduino* tienen limitaciones para su uso en la enseñanza, en tanto se requiere programar en texto, con la dificultad que implica para una persona que recién se inicia en la programación. Adicionalmente, las herramientas disponibles por bloques, o bien son específicas de ciertos kits, y no son compatibles con sensores y actuadores genéricos, o bien tienen pocos bloques disponibles, generan código poco legible y no son intuitivas en su uso, además de estar en su mayoría sólo en idioma inglés, entre otras cosas. En este trabajo presentamos la herramienta *Arduino en la Escuela*, que fue diseñada para la enseñanza de programación utilizando placas *Arduino* por medio de un lenguaje por bloques. Se destaca su versatilidad en tanto fue utilizada en contextos variados y con públicos de características muy diferentes, no sólo en edad, sino también en contexto sociocultural y formación previa. Las experiencias con la herramienta fueron mayormente con estudiantes de nivel secundario, pero también se realizaron talleres de formación para docentes e, inclusive, cursos para personas en contexto de encierro. En todos los casos se tuvo una buena aceptación de la herramienta, además de que los participantes pudieron completar adecuadamente las tareas propuestas y trabajar exitosamente con proyectos a elección.

Index Terms—Entorno de programación, Enseñanza de Programación, *Arduino*, Programación por bloques

I. INTRODUCCIÓN

La enseñanza de programación es relevante para la educación obligatoria, lo que se vio reflejado en Argentina en su incorporación en todos los niveles obligatorios en 2018 [1], en tanto aporta al pensamiento abstracto y a la resolución de problemas, entre otros aspectos. Adicionalmente, la utilización de programación de placas *Arduino*, con su diseño de hardware abierto y multiplicidad de recursos disponibles, permite trabajar con programación tangible, además de facilitar el trabajo con proyectos. Existen muchas herramientas que se usan para programación en el contexto escolar, como

Scratch [2] y *PilasBloques* [3]. En particular, para *Arduino* se suelen utilizar herramientas como *mBlock* y el mismo *Scratch*, además de otras herramientas específicas de kits de robótica *Arduino*-compatibles u otras versiones propietarias, como *LEGO Mindstorms*. A nivel nacional, se cuenta con *ArguiBot* [4] que, a pesar de ofrecer varias ventajas, tanto de usabilidad como del punto de vista didáctico, sólo está diseñado para el uso dentro de un curso, y no presenta funcionalidades para uso en forma general. De acuerdo a lo relevado [5], todas las herramientas disponibles para el trabajo de programación con *Arduino* tienen limitaciones para su uso en la enseñanza, en tanto se requiere programar en texto, con la dificultad que implica para una persona que recién se inicia en la programación. Adicionalmente, las herramientas disponibles por bloques, o bien son específicas de ciertos kits, y no son compatibles con sensores y actuadores genéricos, o bien tienen pocos bloques disponibles, generan código poco legible y no son intuitivas en su uso. Además, la mayoría de ellas están sólo en idioma inglés. Así, en este trabajo presentamos la herramienta *Arduino en la Escuela* [6] (*AelE*), que fue diseñada para la enseñanza y el aprendizaje de programación utilizando placas *Arduino* por medio de un lenguaje por bloques.

El desarrollo de *AelE* fue motivado por la necesidad de una herramienta que permita programar placas *Arduino*, pero que a su vez esté especialmente diseñada para el ámbito educativo. Ninguno de los entornos de programación por bloques para *Arduino* cumplía con todos los requerimientos, o eran muy poco robustas en cuanto al código generado a partir de los bloques. Se comenzó haciendo una traducción al español de *Ardublockly* y ampliando su catálogo de bloques para permitir la interacción con más módulos de *Arduino*, aunque, posteriormente, el proyecto tomó entidad propia, diferenciándose lo suficiente como para poder ser una herramienta independiente.

II. CARACTERÍSTICAS DE LA HERRAMIENTA

Se definieron las siguientes dimensiones para describir las principales características de *Arduino en la Escuela*:

II-A. Preparado para la enseñanza y el aprendizaje

En esta sección se describen las características que hacen de *AelE* una herramienta ideal para ser utilizada en el aula

¹*Arduino en la Escuela*- Demo: <http://aele.dc.uba.ar>

ya que facilitan el ingreso al mundo de la programación. Son características que ayudan a generar un ambiente propicio para el aprendizaje asistiendo tanto a estudiantes a ganar autonomía como a docentes para tener más recursos para la enseñanza.

1. **Programación por bloques:** La forma de construir programas es mediante programación por bloques. En contraposición a la construcción de programas mediante la escritura de texto, como en el caso del *IDE de Arduino*, la programación por bloques es mucho más intuitiva para principiantes ya que expone visualmente las reglas de construcción sintáctica, las que suelen dificultar el aprendizaje cuando se empieza en texto.
2. **Bloque principal:** Muchos entornos de programación por bloques generan el código a partir de todos los bloques disponibles en el área de trabajo. Esto puede resultar muy confuso cuando hay una gran cantidad de bloques en diferentes posiciones, ya que no queda claro en qué orden deben ejecutarse o qué es lo que dispara su ejecución. *AeIE* incorpora un bloque que define el ciclo principal de ejecución. Al iniciar el programa, sólo lo que esté contenido dentro de este bloque será ejecutado. Los bloques sueltos se ignoran al no haber una clara definición de cómo deberían ejecutarse.
3. **Mensajes de advertencias y errores descriptivos:** En general, los mensajes de error en los entornos de programación son muy poco descriptivos. Las personas con experiencia se van acostumbrando a entender estos mensajes, pero son muy difíciles de interpretar para quienes recién están aprendiendo a programar. Muchas herramientas orientadas a la educación resuelven este problema poniendo esfuerzo en generar mensajes descriptivos, aunque son pocas las que lo hacen para *Arduino*. En muchos casos, los mensajes de error generados por estas herramientas son los mismos que se muestran en el editor de *Arduino*.
4. **Asistencia en corrección de errores:** Además de contar con mensajes descriptivos, *AeIE* provee algunas sugerencias para solucionar los errores detectados. Así, se busca que el usuario entienda por qué no funciona lo que está intentando hacer y ver cómo podría solucionarlo.
5. **Evita correcciones automáticas:** Muchas herramientas, orientadas tanto a profesionales como a principiantes, hacen uso de correcciones automáticas. Aunque pareciera una característica deseable, atenta contra que los estudiantes aprendan a solucionar los problemas por sí mismos. En la mayoría de las herramientas basadas en *Blockly* no es posible conectar dos bloques si sus tipos no coinciden (e.g., ver qué sucede si se intenta conectar un bloque numérico como condición de un bloque de alternativa). El problema de utilizar este comportamiento es que no permite que los estudiantes comprendan por qué esa composición es inválida. En *AeIE*, en lugar de impedir la conexión, se muestra un mensaje de error explicando claramente por qué no se puede hacer.
6. **Composición de subtareas descendente:** Otra carac-

terística que comparten casi todas las herramientas basadas en *Blockly* es el mecanismo mediante el cual se definen subtareas. Este proceso consiste en crear primero un bloque de definición de subtask, lo cual luego habilita la creación del bloque de invocación de dicha subtask. En el esquema de trabajo descendente (en inglés, *top-down*) se recomienda escribir primero la estrategia de solución del problema principal, invocando a las subtareas correspondientes sin necesidad de haberlas definido. Siguiendo esta lógica, *AeIE* permite crear bloques de invocación a subtareas sin definirlos antes.

7. **Comunicación serial:** En las etapas iniciales de aprendizaje de programación es de suma importancia poder depurar el código. Dado que el código se ejecuta en un hardware distinto al utilizado para escribir el código, sería demasiado complejo implementar un depurador completo. Aún así, es extremadamente útil poder imprimir valores en determinados puntos de la ejecución del programa. Para ello, *AeIE* incluye el manejo de la conexión serial entre la placa y la computadora. Esto no sólo permite depurar el código sino que también es útil para explorar nuevos sensores para los cuales se desconoce el rango de valores que pueden describir.
8. **Ejemplos y Tutoriales:** *AeIE* cuenta con varios ejemplos, básicos e intermedios, que pueden servir como base para implementar otros proyectos o como guía para entender cómo interactuar con un módulo en particular. También cuenta con algunos tutoriales interactivos para empezar a utilizar la herramienta.

II-B. Interfaz de usuario cómoda e intuitiva

Una herramienta usable debe contar con una interfaz de usuario (IU) fácil de navegar e intuitiva. Las IU muy sobrecargadas abruman al usuario con demasiada información, aunque, en el extremo opuesto, una interfaz demasiado básica dificulta encontrar las características requeridas en cada momento. *Arduino en la Escuela* utiliza un enfoque equilibrado, con una interfaz con poco contenido, pero que permite incorporar nuevos elementos/funcionalidades en la medida que sea necesario.

1. **Máximo aprovechamiento del área de trabajo:** La mayor parte de la interfaz está ocupada por el área de trabajo de bloques. En la barra de título hay un botón para abrir el menú lateral en el extremo izquierdo y algunos botones de acceso rápido en el extremo derecho, como el de abrir el menú de configuración o la ayuda. En el centro de la barra de título se encuentra el nombre del proyecto junto a un botón para modificarlo.
2. **Vistas y modo de visualización de vistas:** Además de la vista de bloques hay una segunda vista que muestra el código *Arduino* generado, y se puede alternar entre ambas vistas con el selector de pestañas. También hay un botón que permite ver ambas vistas a la vez, siendo posible redimensionar el ancho de cada una para darle prioridad a una, pero sin dejar de ver la otra.
3. **Monitor serial incorporado:** En la barra inferior se encuentra la ventana del monitor serial que aparece

minimizada por defecto, pero se puede abrir haciendo clic sobre ella. Nuevamente, es posible redimensionar su altura con respecto al resto de la interfaz.

4. **Íconos:** La gran mayoría de los botones incluyen íconos para describir su comportamiento, como los de acceso rápido y los distintos menús, así como las opciones en los menús contextuales. También se encuentran íconos en las categorías del toolbox y en muchos de los bloques.

II-C. Robustez

Una de las características más importantes de *Arduino en la Escuela*, que pocas herramientas comparten, es que el código generado es siempre válido para el *IDE de Arduino*. Así, si *AeIE* no muestra errores en los bloques, entonces el *IDE de Arduino* tampoco generará errores al pasar el código a la placa. Esto es algo que se espera en general de las herramientas de programación por bloques. Sin embargo, es difícil manejar errores de tipos o referencias sin definir en el contexto de los bloques en el que se omiten todas esas declaraciones.

1. **Manejo de tipos:** *Blockly* fue pensado originalmente para lenguajes no tipados o débilmente tipados. Al usarlo para generar el código de un lenguaje tipado surgen dificultades para declarar el tipo de las variables, las funciones y los parámetros. *AeIE* cuenta con un sistema de tipado propio que permite inferir los tipos de todas las variables, funciones y parámetros y generar errores cuando estos tipos no coinciden en distintas apariciones.
2. **Parámetros, variables locales y variables globales:** Una de las principales falencias de los entornos de programación por bloques para *Arduino* es su incapacidad para manejar parámetros, así como para distinguir el variables locales y globales. En *ArduBlockly*, por ejemplo, se optó por anular el uso de parámetros y forzar el alcance global a cualquier variable que se declare. Gracias a su sistema de tipado, *AeIE* puede permitir el uso tanto de parámetros como de variables locales.
3. **Errores y advertencias estáticas:** Como la mayoría de los entornos de desarrollo modernos, *AeIE* analiza el código mientras este está siendo construido y genera notificaciones tanto de errores como de advertencias en el mismo. Estas notificaciones se muestran a través de la característica de *Blockly* que muestra burbujas sobre los bloques. *Blockly* tiende cada vez más a dejar de usar estos mensajes flotantes y restringir las construcciones de código erróneas de otras formas y así también lo están haciendo los entornos basados en *Blockly*. Como se mencionó en la sección II-A, preferimos que se puedan generar construcciones erróneas, pero que se muestre un mensaje que ayude a entender por qué esa construcción es inválida, en lugar de simplemente impedirla. Además, de los errores de tipado ya mencionados, el análisis realizado permite detectar e informar una gran cantidad de errores, como usos inválidos de los pines de la placa o la falta de bloques de configuración para módulos que lo requieren, así como generar advertencias, como elementos que están definidos más de una vez o la

asignación de un valor a un parámetro. Las advertencias pueden ser ignoradas (lo que significa que el código puede generarse aunque tenga advertencias, pero el resultado podría no ser el esperado) mientras que los errores impiden la generación del código *Arduino*.

4. **Errores en tiempo de ejecución:** Además de los errores que se pueden detectar haciendo análisis estático de código, *AeIE* también contempla errores en tiempo de ejecución, como un acceso fuera de rango a una lista o intentar acceder a un archivo de la tarjeta SD que no existe. En el funcionamiento normal de *Arduino*, la mayoría de estos errores detienen la ejecución sin notificar de ninguna forma que ocurrió un error. El código generado por *AeIE* captura estos errores y puede mostrarlos a través del monitor serial.
5. **Control de acceso a placas:** Más allá de los errores en la ejecución del código, el *IDE de Arduino* también genera errores cuando hay problemas de comunicación con la placa. Estos también se evitan con *AeIE*, ya que se asegura que haya una placa conectada y que esta sea aquella para la cual el proyecto está configurado.

II-D. Herramientas de productividad

Aquí se describen características de los entornos de desarrollo para facilitar y acelerar la programación. Tienen el doble objetivo de, por un lado, que sea más fácil aprender a programar con la herramienta y, por el otro, que usuarios expertos puedan crear proyectos más complejos. Algunas de estas funcionalidades están pensadas para quienes ya dominan entornos de desarrollo textuales y encuentran dificultades con la forma de trabajar que proponen los entornos visuales basados en bloques, como la interacción a través del mouse.

1. **Comunicación con el IDE de Arduino incorporada:** La mayoría de los entornos de desarrollo modernos permiten ejecutar y depurar el código realizado desde la misma interfaz del editor. De la misma forma, *AeIE* permite ejecutar el código a la placa, utilizando por detrás el *IDE de Arduino*, sin abandonar la aplicación.
2. **Autocompletado:** Por medio del sistema de tipado mencionado en la sección II-C, *AeIE* provee la funcionalidad de completar automáticamente los bloques que tienen huecos, los cuales se rellenan con bloques por defecto, en función del tipo correspondiente.
3. **Buscador de bloques:** Es muy común que en los entornos de programación por bloques sea difícil encontrar un bloque particular para su uso, sobre todo cuando el entorno incluye una gran variedad de bloques distintos, y los entornos para *Arduino* suelen caracterizarse por ello). Para aliviar este problema, *AeIE* cuenta con un buscador por texto de bloques.
4. **Multiselección de bloques:** Otro inconveniente que suelen tener los entornos de programación por bloques es que sólo permiten seleccionar un bloque a la vez. En cualquier entorno de desarrollo es posible seleccionar varios elementos a la vez y por eso a la interfaz de *Blockly* se le agregó la posibilidad de seleccionar varios

bloques en simultáneo, y poder realizar acciones sobre ellos, como arrastrar y soltar.

5. **Pegar a continuación:** Se busca que *AeIE* se asimile a las herramientas de programación textual al momento de editar código. Así como en texto se cuenta con un cursor, y al realizar alguna operación (e.g., copiar y pegar) el resultado se realiza a continuación de él, en la mayoría de las herramientas basadas en *Blockly* el comportamiento de pegar un bloque crea uno nuevo, suelto en una posición cercana al bloque copiado. En *AeIE* la posición del nuevo bloque depende de lo que esté seleccionado en ese momento. Si hay un bloque seleccionado, primero se intenta conectar el nuevo bloque al bloque seleccionado. Si este tiene huecos y el bloque pegado es una expresión, entonces se usa para rellenar el hueco. Si ambos son comandos, el pegado se conecta como bloque siguiente de la secuencia. Si no es posible conectarlos (e.g., si el seleccionado es una expresión y el pegado un comando) el comportamiento es el de *Blockly*, pero no respecto al bloque copiado sino al seleccionado. Si no hay ningún bloque seleccionado el comportamiento es el mismo, pero respecto al bloque copiado, si es que este sigue existiendo.
6. **Edición rápida en los menús contextuales:** Además de la opción para rellenar huecos automáticamente, se permite la edición rápida desde los menús contextuales de los bloques, siendo una de las más relevantes el convertir a un bloque de comando en un procedimiento y convertir a un bloque de expresión en una función, una variable, un parámetro o un valor constante. También se incluyen opciones adicionales en el menú contextual del área de trabajo, como abrir o cerrar todas las burbujas, los comentarios, los errores o las advertencias.

II-E. Diseño del lenguaje

En esta sección se encuentran las características que hacen que el lenguaje de programación por bloques de *Arduino* en la *Escuela* sea completo y versátil.

1. **Gran variedad de sensores y actuadores:** Encontrar un equilibrio en la cantidad de bloques disponibles es una tarea muy compleja para cualquier entorno de programación por bloques, pero más aún para uno hecho para programar placas *Arduino*, debido a la enorme cantidad de sensores y actuadores disponibles. Muchos entornos optan por los extremos, ya sea incluyendo un conjunto de bloques mínimo, lo que hace que construir programas complejos sea más dificultoso, o incluyen una gran variedad de bloques, lo que hace que sea difícil encontrar un bloque particular cuando se lo necesita. *AeIE* tiene bloques para los sensores y actuadores más comunes, pero asegurándose de no abrumar al usuario con demasiada información a la vez. Además de contar con el buscador de bloques mencionado en la sección **II-D**, los bloques se organizan en categorías y subcategorías que facilitan la navegación.

2. **Gran variedad de elementos de programación:** Al crear una herramienta por bloques se define un nuevo lenguaje, el cual traducirá al lenguaje de base, que generalmente es textual y de más bajo nivel. Muchos entornos utilizan el mismo lenguaje que aquel al que se traducen, pero reemplazando los elementos textuales por bloques. Esto tiene la ventaja de que la conversión de bloques a texto es sencilla y muchas veces hasta puede ser bidireccional, pero, a su vez, tiene la desventaja de que el lenguaje de bloques queda restringido por el lenguaje al que se traduce. En el caso de *Arduino*, al ser un lenguaje de bajo nivel, esto significa un problema para el ámbito educativo. Es por ello que el lenguaje de *AeIE* es de más alto nivel que el de la mayoría de los demás entornos visuales para *Arduino* y por eso permite trabajar con elementos del lenguaje que no están presentes en *Arduino* como listas y eventos.
3. **Variantes y extensiones:** Otra característica que ayuda a evitar el exceso de bloques es que algunos bloques tienen variantes. Es frecuente que los entornos de programación por bloques para *Arduino* cuenten con un bloque que active un zumbador y otro que lo active durante una cantidad de tiempo, mientras que en *AeIE*, el segundo bloque es una variante del primero, por lo que ocupan un único espacio en el menú. Además, el bloque principal permite configurarse para que ejecute una única vez, que es el comportamiento habitual en la mayoría de los demás lenguajes de programación, además de hacerlo indefinidamente, que es el comportamiento por defecto en *Arduino*. Así, se accede a estas variantes desde cada bloque, donde se muestran las configuraciones adicionales que tiene disponibles.
4. **Bloques para describir pines:** Cuando se configura un sensor o un actuador, en *Arduino* se debe especificar en qué pines de la placa está conectado. Esta información puede ser suministrada en un bloque como un argumento o a través de un campo selector. Mientras que la mayoría de los entornos optan por la alternativa del argumento, tienen el problema de que no pueden validar que lo ingresado sea un pin válido, ya que se debe verificar que sea un número entero y que esté en el rango de los pines que la placa posee. Además, esta decisión de diseño permite que el argumento sea una variable o, incluso, el resultado de una función, lo que hace que no se pueda resolver antes de comenzar la ejecución (o, peor aún, que vaya cambiando a lo largo del programa). La alternativa del campo selector es más segura, ya que permite definir qué pines serán utilizados para sensores y actuadores antes de comenzar el programa, pero restringe la libertad del programador impidiendo, por ejemplo, tener definido un pin a través de un nombre y poder cambiarlo en un único lugar para todo el programa. *AeIE* combina varias características para lograr una correcta implementación de la alternativa por argumento, pero sin riesgo de generar programas inválidos: por un lado incorpora bloques específicos para describir a los pines, y usando el sistema

de tipado impide que pasen argumentos que no son pines cuando se espera algo de este tipo, además de que se verifica que los bloques no sean variables ni funciones (además de que las variables no pueden recordar pines ni las funciones pueden devolverlos), y, finalmente, se incorporan bloques de definición de valores constantes para poder asignarle nombre a los pines.

II-F. Personalización

Se espera que *AelE* pueda adaptarse a distintos contextos de uso, por lo que puede personalizarse a través del menú de configuración, más allá de las configuraciones iniciales.

1. **Editor de paletas:** Como ya se mencionó en la sección **II-E**, es difícil diseñar un menú de bloques (paleta) sin llegar a los extremos en los que hay tantos que es difícil encontrar lo buscado o que hay tan pocos que cualquier programa, por más simple que parezca, requiere una gran cantidad de bloques para ser construido. Más allá de que la paleta por defecto está balanceada en ese sentido, también podrían darse situaciones particulares en las que se requiera tener otros sets de bloques. Así, se implementó un editor avanzado de paletas que habilita a que docentes puedan diseñar sus propios set de bloques en función de las necesidades de sus clases particulares. El editor no sólo permite seleccionar qué bloques aparecen en la paleta, sino que también permite agruparlos en categorías personalizables.
2. **Herramientas habilitadas:** Además de personalizar los bloques disponibles en el menú, *AelE* permite controlar qué herramientas del lenguaje se pueden usar. Así, es posible planificar distintas actividades restringiendo el acceso a determinadas herramientas que no se espera que sean utilizadas para la resolución. Adicionalmente, se pueden configurar algunos bloques para que sólo aparezcan en la paleta en ciertas ocasiones (e.g., que los bloques de configuración de módulos sólo aparezcan si en el área de trabajo hay un bloque para tal módulo).
3. **Manejo de errores y advertencias:** Es posible configurar la forma en que *AelE* notifica los errores y las advertencias en el código. Las advertencias pueden ser tratadas como errores (i.e., impiden generar el código), ser tratadas sólo como advertencias (lo cual es comportamiento por defecto) o directamente ser ignoradas. En lo respectivo a los errores, estos no se pueden ignorar, pero se puede decidir si se los muestra apenas son detectados o recién cuando se intenta ejecutar el código. Por defecto, los errores sobre procedimientos o funciones sin definir se muestran al intentar ejecutar el código, mientras que el resto se muestra inmediatamente.

III. RESULTADOS

AelE se pudo implementar en algunos cursos y talleres de programación, con públicos de diferentes edades y características. Se destacó por su versatilidad, ya que fue utilizada en contextos variados y con públicos de características muy diferentes, no sólo en edad, sino también en contexto sociocultural

Cuadro I
PREGUNTAS Y PUNTAJE PROMEDIO DE LAS RESPUESTAS A CADA UNA.

Pregunta	Puntaje
¿Qué tan difícil creías que era programar Arduino ANTES?	3.68
¿Qué tan difícil creés que AHORA?	2.00
¿Cómo se ve la herramienta?	4.74
¿Cuánto cuesta usarla?	2.27
Cuántas cosas se pueden hacer con ella	4.61
¿Te gustaría seguir usándola?	4.70
¿Cómo fue tu experiencia general usando Arduino?	4.41

y formación previa. Aunque la mayoría fueron estudiantes de nivel secundario en contexto extracurricular, también se realizaron talleres de formación docente y cursos de educación en contexto de encierro. En todos los casos la herramienta tuvo buena aceptación de los participantes, quienes pudieron realizar todas las tareas propuestas. En particular se destaca el éxito logrado en el trabajo con proyectos, instancia de evaluación final en la mayoría de los cursos en los que se utilizó la herramienta durante varios encuentros.

En 2019 se realizó un taller sobre robótica y electrónica de cinco encuentros para 60 estudiantes de entre 11 y 17 años. Desde el segundo encuentro se trabajó exclusivamente con *AelE*, y como cierre del taller los participantes tuvieron que presentar un proyecto grupal. Al finalizar el taller 23 participantes completaron una encuesta de satisfacción acerca del taller y de la herramienta usada. Cada pregunta se respondía con un número del 1 al 5. En el cuadro **I** se muestran las preguntas y el puntaje promedio para cada una. Como puede verse, tras finalizar la cursada la percepción de la dificultad de programar con *Arduino* se redujo a casi la mitad.

En 2023 se realizó un taller de programación de seis encuentros para 44 estudiantes de entre 16 y 18 años utilizando únicamente *AelE*, cuya planificación y temario se pueden ver en **II-G**, en el cual los participantes tuvieron que presentar un proyecto final grupal. Luego se realizó una encuesta de satisfacción acerca del taller y de la herramienta usada, que fue respondida por 32 participantes. En el cuadro **II** se muestran las preguntas y el puntaje promedio para cada una (cada pregunta se respondía con un número del 1 al 5).

Durante 2023 se desarrollaron 2 cursos cuatrimestrales de robótica y pensamiento computacional de 2 horas semanales en los penales de Devoto y Ezeiza. Participaron 92 personas el primer cuatrimestre y 47 en el segundo (muchas de las personas que se inscribieron en el segundo habían cursado el primero), y como cierre de cada curso los participantes tuvieron que presentar un proyecto grupal. En ambos cursos hubo una amplia variedad de edades, con una importante proporción de participantes mayores de 60 años, y al nivel educativo alcanzado, ya que se tuvo participantes con educación primaria hasta profesionales universitarios. El primer curso fue aprobado por 50 personas y el segundo por 29. Al finalizar cada curso se realizó una encuesta anónima en la que se pedía calificar la herramienta *AelE*. En el primero, 27 personas la dieron como “Excelente” y 22 como “Buena”. En el segundo, 19 la dieron como “Excelente” y 3 como “Buena”.

Cuadro II
PREGUNTAS Y PUNTAJE PROMEDIO DE LAS RESPUESTAS A CADA UNA.

Pregunta	Puntaje
¿Qué te pareció en general la herramienta AelE?	4.28
Fácil de usar (no presenta complejidades...)	3.59
Intuitiva (es fácil encontrar las cosas que necesito...)	3.69
Entretenida (me divierte hacer actividades y me dan ganas de hacer más actividades similares)	4.13
Potente (me permite realizar las cosas que quiero realizar)	3.69
Educativa (mientras lo uso voy entendiendo las cosas que hago y cuando algo falla me resulta fácil entender por qué)	3.88
Frustrante (me topo con problemas que no puedo resolver y la herramienta no me ayuda a entender cómo solucionarlos)	2.78
Compleja (me cuesta entender qué hace cada cosa y cómo hacer lo que quiero hacer)	2.59
Motivadora (me dan ganas de seguir usándola, de seguir explorando y de seguir pensando qué más podría hacer)	3.78
Visualmente atractiva (la interfaz es agradable a la vista)	3.94
Aburrida (resolver ejercicios es tedioso...)	2.09
En general, ¿cómo fue tu experiencia usando AelE?	4.25

En ningún caso fue evaluada como “Regular” o “Mala”.

La docente a cargo de los cursos también evaluó el uso de *AelE* como herramienta. En el primer curso, introductorio, funcionó muy bien ya que no se realizaron proyectos complejos y sólo se requirieron los bloques más estándar. En el segundo, un poco más avanzado, surgieron algunas dificultades al utilizar bloques o estructuras de código más complejos. Por un lado, dejó de ser fácil e intuitivo ubicar los bloques necesarios en la paleta. Algunos bloques sólo aparecen en determinadas circunstancias (e.g., los bloques de configuración de algunos módulos sólo aparecen cuando hay en el área de trabajo un bloque que utiliza tal módulo) y, si bien se puede configurar la herramienta para cambiar ese comportamiento (como se mencionó en la sección II-F), la docente no tenía conocimiento de ello. Tampoco sabía la docente cómo habilitar las variables globales, que por defecto están deshabilitadas.

Por otro lado, los proyectos más complejos requieren muchos más bloques en el área de trabajo a la vez, lo que lleva a que el código sea mucho menos legible (aun haciendo división en subtarefas) y difícil de navegar, sobre todo en monitores pequeños o de baja resolución. También se notó una gran falta de autonomía en la mayoría de los grupos en el segundo curso con respecto al primero. Adicionalmente, resultó poco intuitivo el hecho de que los bloques de configuración aparezcan “sueltos” en el área de trabajo y no conectados a otro bloque.

IV. DISCUSIÓN

La amplia experiencia con talleres introductorios de programación nos permite afirmar que *AelE* es una excelente herramienta para iniciarse en el mundo de la programación con *Arduino*. Nuestros objetivos iniciales eran que también sirviera para proyectos avanzados y para gente con más experiencia. Sin embargo, no tenemos suficientes experiencias con cursos ni con usuarios avanzados. También es importante mencionar que la mayoría de las experiencias utilizando *AelE* fueron supervisadas por integrantes del equipo de desarrollo, y que, en las pocas instancias en las que la herramienta fue utilizada

sin su supervisión, surgieron complicaciones que los docentes a cargo no supieron cómo resolver.

V. CONCLUSIONES

Arduino en la Escuela fue diseñada y construida para facilitar la enseñanza de la programación de placas *Arduino*, superando las limitaciones de las herramientas existentes, con un enfoque didáctico-pedagógico. Adicionalmente, incluye muchas funcionalidades que no están disponibles en la mayoría de los demás entornos de programación por bloques para *Arduino*. En particular, permite trabajar con una gran variedad de elementos de programación que no son habitualmente soportados, como parámetros, listas y eventos, y cuenta con características que la hacen más cercana a los entornos de desarrollo textuales, como la selección múltiple, pegar a continuación, o el manejo de errores y advertencias. También admite un alto nivel de personalización, que se adecua a las necesidades de cada equipo docente. *AelE* se pudo usar con éxito en distintos contextos, tanto como parte de la educación obligatoria de todos los niveles, en cursos extracurriculares, como parte de formación docente y en otros contextos de enseñanza (e.g., en contextos de encierro).

VI. TRABAJO FUTURO

Se está trabajando en varias mejoras para la herramienta. La principal es la incorporación de una versión textual del lenguaje y la capacidad para hacer la conversión bidireccional entre ambas versiones. Para mejorar la usabilidad en proyectos grandes se está planificando una nueva interfaz que permite manejar varias pestañas a la vez, de forma de poder organizar mejor el código. Además, se planean mejoras de accesibilidad, entre las que están contar con una versión apta para educación inicial o estudiantes con dificultades para la lectoescritura, y se espera desarrollar adaptaciones para su uso por parte de estudiantes con problemas motrices o cognitivos.

REFERENCIAS

- [1] C. F. de Educación, “Núcleos de aprendizajes prioritarios para educación digital, programación y robótica,” *Anexo I de la Resolución CFE Nro. 343/18*, 2018. [Online]. Available: http://www.bnm.me.gov.ar/giga/1/normas/RCFE_343-18.pdf
- [2] M. Resnick, J. H. Maloney, A. Monroy Hernández, N. Rusk, E. Eastmond, K. Brennan, A. Millner, E. Rosenbaum, J. Silver, B. Silverman, and Y. Kafai, “Scratch: Programming for all,” *Communications of the ACM*, vol. 52, no. 11, pp. 60–67, 2009.
- [3] A. Sanzo, F. Schapachnik, P. Factorovich, and F. Sawady O'Connor, “Pilas bloques: A scenario-based children learning platform,” *2017 Twelfth Latin American Conference on Learning Technologies*, pp. 1–6, 2017.
- [4] N. Streri, M. Daniele, and M. Uva, “Arguibot ide: Un entorno de desarrollo basado en la programación en bloque para la enseñanza de la robótica usando la plataforma arduino,” *Jornadas Argentinas de Didáctica de las Ciencias de la Computación 2023*, 2023.
- [5] G. P. Fernández and C. Cossio-Mercado, “Assessment of arduino block-based programming environments for teaching and learning,” *19th WiPSCE Conference on Primary and Secondary Computing Education Research*, 2024. [Online]. Available: https://www.researchgate.net/publication/380568592_Assessment_of_Arduino_block-based_programming_environments_for_teaching_and_learning
- [6] —, “Enseñanza de fundamentos conceptuales de programación usando arduino,” *Jornadas Argentinas de Didáctica de las Ciencias de la Computación 2023*, 2023. [Online]. Available: https://www.researchgate.net/publication/380174232_Enseñanza_de_fundamentos_conceptuales_de_programacion_usando_Arduino

El impacto de la programación interactiva en actividades de robótica educativa

Ricardo Moran

Centro de Altos Estudios en Tecnología Informática
Universidad Abierta Interamericana
Buenos Aires, Argentina
ricardo.moran@uai.edu.ar

Gonzalo Zabala

Centro de Altos Estudios en Tecnología Informática
Universidad Abierta Interamericana
Buenos Aires, Argentina
gonzalo.zabala@uai.edu.ar

Resumen—La programación interactiva es usualmente considerada beneficiosa para el aprendizaje debido a que reduce el costo de la experimentación, facilitando la programación exploratoria. Para medir el impacto de la programación interactiva en robótica educativa, realizamos un estudio piloto con estudiantes de secundaria. Desarrollamos dos configuraciones distintas del mismo entorno de programación: una con todas las funciones del lenguaje habilitadas y otra con las funciones de programación interactiva, monitoreo y depuración desactivadas. Utilizamos estos dos entornos para comparar el rendimiento de un grupo de estudiantes durante un taller introductorio de robótica. Los resultados encontrados son consistentes con algunos estudios similares: la programación interactiva puede ayudar a los estudiantes a progresar más, mejorar su eficiencia, reducir el tiempo requerido para resolver cada tarea y aumentar la frecuencia con la que se prueban los programas en los robots.

Keywords—Programación interactiva, Robótica educativa

I. INTRODUCCIÓN

La programación interactiva es usualmente considerada beneficiosa para el aprendizaje debido a que reduce el costo de la experimentación, facilitando la programación exploratoria [1]. Permite un estilo de programación caracterizado por el ensayo y error, fomentando la prueba de diferentes ideas y acortando el intervalo entre la concepción de una idea y su validación. Asimismo, los beneficios de la programación interactiva en educación radican en hacer visible el comportamiento dinámico de los programas, facilitando así su comprensión e inspección [1]. Aunque la mayoría de los entornos de programación visual (como Scratch, Etoys, Alice y otros) han sido diseñados para ofrecer una experiencia de aprendizaje interactiva, donde el feedback inmediato es una característica fundamental y las modificaciones al programa se evalúan y reflejan automáticamente en la pantalla, un énfasis comparable en la inmediatez y la transparencia no ha sido tan prevalente en los entornos de programación de robótica educativa.

En las últimas décadas, ha habido un aumento constante en el interés y la posterior adopción de entornos de programación visual y kits de robótica específicamente diseñados para principiantes [2]. Y aunque la robótica naturalmente crea un bucle de retroalimentación donde los resultados del programa se traducen en acciones del mundo real por parte del robot, las herramientas para programar estos robots a menudo carecen del mismo nivel de interactividad observado en las contrapartes puramente virtuales. La mayoría de los lenguajes de programación educativos para robótica requieren que los usuarios compilen y envíen sus programas al robot antes de poder ver sus efectos en el mundo real (por ejemplo, Arduino, LEGO Mindstorms, MakeCode, entre otros) [3][4]. En otros casos, el feedback que los estudiantes

pueden obtener está limitado por la falta de herramientas de monitoreo que muestren el estado interno del robot, obligando a los principiantes a "imaginar" lo que está sucediendo con sus robots. Esto puede ser especialmente difícil cuando el robot no hace lo que el usuario espera. Finalmente, ciertos entornos proporcionan un bucle de retroalimentación rápido y una experiencia interactiva, pero esto a menudo se hace a expensas de la autonomía del robot [5][6][7].

Para abordar este problema, hemos desarrollado Physical Bits [8], un entorno de programación basado en web para robótica educativa que admite programación interactiva y autonomía mediante un lenguaje de programación híbrido de bloques y texto. Con el objetivo de medir el impacto de la programación interactiva en robótica educativa, realizamos un estudio piloto con estudiantes de secundaria utilizando este entorno de programación. Al diseñar nuestro propio entorno, pudimos ajustar fácilmente las características del lenguaje, como ocultar herramientas de monitoreo o desactivar la programación interactiva. Aprovechando esta adaptabilidad, desarrollamos dos configuraciones distintas: una con todas las funciones del lenguaje habilitadas y otra con las funciones de programación interactiva, monitoreo y depuración desactivadas. Luego dividimos a los estudiantes en dos grupos (cada uno utilizando una configuración diferente) y comparamos su rendimiento durante un taller introductorio sobre robótica. Este artículo analiza los resultados de este estudio.

II. TRABAJO RELACIONADO

Varios trabajos previamente han analizado la importancia de la programación interactiva y del feedback continuo, con resultados diversos.

Un estudio que compara la efectividad de desarrolladores de JavaScript trabajando con un entorno interactivo con aquellos que trabajan sin él encontró que aquellos participantes que utilizaron programación interactiva detectaron y corrigieron errores más rápido, realizaron modificaciones al código uniformemente distribuidas en el tiempo y verificaron el código con mayor frecuencia sin una disminución en el tiempo de finalización de la tarea [9].

Otro estudio intentó verificar la efectividad de proporcionar feedback en tiempo de ejecución automáticamente después de cada modificación del programa [10]. Los resultados sugieren que presentar feedback al usuario (ya sea automática o manualmente) ayuda a desarrollar programas con menos errores sintácticos y semánticos. Sin embargo, la visualización automática del feedback no muestra beneficios significativos en comparación con el feedback manual, siempre y cuando éste se presente sin demora.

Un estudio independiente intentó evaluar el impacto del feedback continuo en la tarea de depuración. Sus hallazgos indican que si bien el feedback visual continuo no muestra una mejora significativa en la depuración en general, sí demuestra un efecto positivo significativo en circunstancias específicas[11].

Estudios adicionales han destacado la importancia de mantener un bucle de retroalimentación corto durante el aprendizaje de programación, postulando que ver inmediatamente los efectos de los cambios en el programa anima a los usuarios a experimentar con el código [12][13].

Finalmente, se llevó a cabo un estudio piloto con el objetivo de comprender el impacto de la programación interactiva en las interacciones con dispositivos de computación física, con niños de entre 11 y 15 años [14]. En este estudio utilizaron dos entornos de programación comparables: uno compatible con la programación interactiva y otro no. Los hallazgos indican que la interactividad realmente influye en las interacciones de los niños tanto en el juego libre como en las actividades estructuradas. Sin embargo, los autores señalan una limitación en su estudio: el uso de entornos de programación distintos, lo que podría haber impactado en los resultados, especialmente en términos de las tasas de éxito dentro de cada grupo.

Hasta la fecha, no ha habido ningún estudio que evalúe la influencia de la programación interactiva en el rendimiento de un grupo de estudiantes utilizando el mismo entorno de programación para robótica educativa.

III. METODOLOGÍA

Se decidió realizar un taller extracurricular para estudiantes de secundaria. En el mismo participaron 19 alumnos de entre 17 y 18 años, todos provenientes del mismo curso. Los alumnos expresaron haber tenido poca o nula experiencia en programación.

El taller consistió de 5 encuentros de 4 horas cada uno, y se realizó un encuentro por semana. Durante la primera sesión, se presentó la actividad y se llevó a cabo un pretest para estimar el conocimiento previo de cada estudiante. El pretest consistió principalmente en ejercicios simples de lógica y programación. Los resultados del pretest se utilizaron para dividir a los estudiantes en 2 grupos, procurando mantener puntajes similares en ambos grupos: el grupo experimental (E), que recibió la versión completa del entorno; y el grupo de control (C), que recibió el entorno configurado con algunas características deshabilitadas (la interactividad, las herramientas de monitoreo, y las herramientas de depuración). Aunque el género no fue un factor a considerar en la separación de los estudiantes, en el grupo experimental la proporción fue equitativa, con una distribución de 5 varones y 5 mujeres, mientras que en el grupo de control participaron 6 varones y 3 mujeres.

Para simplificar el problema y facilitar que se concentren sólo en la programación se dio a cada alumno una placa de pruebas que incluía ya resuelta toda la electrónica necesaria para resolver los ejercicios. La placa estaba compuesta por un Arduino Nano, 3 LEDs, 2 botones, 1 potenciómetro, 1 LDR, y pines para conectar servomotores y motores de corriente continua. El haber resuelto la electrónica previamente permitió una mejor utilización del tiempo de actividad, lo que llevó a la definición de una serie de 70 ejercicios. Estos ejercicios fueron diseñados con la intención de suavizar la

curva de dificultad y minimizar el impacto de la frustración en los alumnos.

El diseño inicial de la actividad contemplaba comenzar el taller explicando los conceptos elementales de programación y robótica usando solo la placa de pruebas conectada por USB a la computadora, y luego, ya en los últimos encuentros, anexar una estructura de ruedas que permitiera convertir a la placa de pruebas en un robót móvil. Sin embargo, la limitación de tiempo y el progreso de los estudiantes en la resolución de los ejercicios no permitió alcanzar el objetivo.

Todas las sesiones comenzaron con una breve presentación por parte del instructor explicando los temas a abordar en los ejercicios del día. Después de la presentación, cada estudiante se sentó en su computadora y comenzó a trabajar en los ejercicios de forma individual. Para facilitar la tarea del instructor, se desarrolló una aplicación que presenta los ejercicios y permite a los estudiantes "enviar" sus soluciones [15]. Una vez completada la actividad, la tarea del instructor consistió en recopilar los archivos generados por la aplicación. Se grabaron videos de ejemplo para explicar las consignas, asegurando que ningún estudiante tuviera que depender únicamente de su propia interpretación; en su lugar, tenían un ejemplo funcional para ver y comparar con su implementación. La aplicación, además, permitió garantizar una resolución estrictamente secuencial de los ejercicios, y se encargaba de registrar la hora en que se enviaba la solución de cada ejercicio, facilitando la tarea posterior de segmentar los eventos correspondientes a cada ejercicio. Sin embargo, la verificación de soluciones no pudo automatizarse con la aplicación, por lo que cada estudiante tuvo que esperar la verificación del instructor antes de pasar al siguiente ejercicio.

IV. PREGUNTAS DE INVESTIGACIÓN

El objetivo de esta experiencia es medir el impacto de la interactividad en la resolución de ejercicios de programación y robótica. Para ello, se buscan responder las siguientes preguntas de investigación:

- RQ1: ¿Cómo impacta la interactividad en el progreso de los alumnos?
- RQ2: ¿Cómo impacta la interactividad en el tiempo de resolución de cada ejercicio?
- RQ3: ¿Cómo impacta la interactividad en la forma de trabajar de los estudiantes?
- RQ4: ¿Cómo impacta la interactividad en la duración del bucle de retroalimentación?

V. EJERCICIOS

Se planificaron 70 ejercicios a completarse durante el transcurso de 4 sesiones. Si bien esta es una lista de ejercicios ambiciosa para un período de tiempo tan corto, experiencias previas con estudiantes sugieren que una curva de dificultad gradual sin saltos abruptos en complejidad es preferible. El escaso tiempo disponible para la actividad implicó una limitación al diseño del taller, imposibilitando trabajar en problemas más libres o proyectos de mayor duración. Como se puede observar en la tabla 1, los conceptos a enseñar durante el taller incluyeron temas relacionados con la robótica (entradas y salidas, motores, sensores, etc.) y la programación (bucles, condicionales, variables, etc.).

Se decidió organizar estos conceptos en una jerarquía de "subconjuntos de lenguaje" donde cada subconjunto es

completo e incluye los subconjuntos anteriores [16]. La lista completa de ejercicios puede verse online en el siguiente link: <https://richom.github.io/pbits-exercises/>

TABLE I. CONCEPTOS A EXPLICAR Y RANGO DE EJERCICIOS

Conceptos	Ejercicios
Salidas digitales Delays con valores constantes Repetición	1 a 10
Entradas analógicas Multitasking Delays con valores variables	11 a 20
Entradas digitales Condicionales Operaciones lógicas y comparación	21 a 40
Manejo de botones Generación de audio Operaciones aritméticas Uso de variables	41 a 60
Servomotores Motores de corriente continua	61 a 70

VI. ENTORNO DE PROGRAMACIÓN

El entorno de programación que se utilizó en el taller, denominado Physical Bits, es un desarrollo propio que incluye una serie de características que lo distinguen de otras herramientas similares.

En primer lugar, el entorno soporta un modelo de programación híbrido basado en editores de bloques y de código que se pueden utilizar simultáneamente. Este modelo está diseñado especialmente para facilitar la transición gradual de los alumnos a lenguajes de programación más sofisticados.

En segundo lugar, el entorno usa un lenguaje de programación simplificado diseñado específicamente para robótica educativa y basado en la sintaxis de C.

En tercer lugar, el entorno soporta programación interactiva sin sacrificar la autonomía del robot. Es decir que los cambios en el programa tienen un efecto visible inmediato en el comportamiento del robot y la ejecución de los programas se realiza siempre en el robot sin requerir una conexión constante con la computadora. Para ello, se desarrolló un firmware que se instala en el robot y que implementa una máquina virtual encargada de la ejecución de los programas del usuario. La comunicación entre el robot y el entorno de programación sólo es necesaria para la programación, y puede realizarse por USB o por Bluetooth.

Finalmente, el entorno incluye además herramientas de monitoreo y depuración, algunas de las cuales son prácticamente inexistentes en otras herramientas similares.

VII. LECCIONES APRENDIDAS

Haber preparado la electrónica con anticipación permitió establecer un número sustancial de ejercicios, aproximadamente 20 por día. Al final, aunque los estudiantes no pudieron completar toda la actividad, el alto número de ejercicios ayudó a suavizar la curva de dificultad, introduciendo conceptos con un ejercicio simple y luego complicándolo gradualmente con problemas más sofisticados,

combinándolo con otros conceptos y requiriendo una lógica más compleja. Sin embargo, el inconveniente de este enfoque es que algunos estudiantes se quejaron de la monotonía de los ejercicios, y el entusiasmo que caracteriza a los talleres de robótica no fue tan evidente al finalizar el taller.

A pesar de diseñar los ejercicios para tener un aumento gradual en la dificultad, ninguno de los estudiantes logró completar los ejercicios más complejos, que involucraban motores y una programación más sofisticada. Quizás hubo demasiados ejercicios para tan poco tiempo, y podría haber sido preferible tener menos ejercicios pero más desafiantes.

Por otro lado, la automatización de las entregas utilizando una aplicación resultó ser un éxito. Aunque llevó tiempo desarrollarla, posteriormente simplificó el trabajo en las clases, permitiendo que el instructor se enfocara sólo en responder preguntas y verificar la corrección de las soluciones. Desafortunadamente, el proceso de verificación no pudo automatizarse, por lo que dependía del instructor informar a los estudiantes si podían proceder al siguiente ejercicio o si había un error en el ejercicio actual que necesitaba corrección. Esto normalmente solo tomaba unos segundos, por lo que la intervención tuvo poco impacto en el tiempo de trabajo de los estudiantes

Al discutir diferentes formas de programar Arduino, algunos estudiantes comentaron que los lenguajes basados en texto parecían más “serios/profesionales”, mientras que los lenguajes basados en bloques eran más “de juguete” (esta afirmación es consistente con los hallazgos en [17]). En respuesta a este comentario, se les mostró a los estudiantes un el editor de código del entorno de programación. Un estudiante se quejó de que el idioma estaba en inglés, lo cual es algo que a menudo se pasa por alto pero que representa un desafío adicional en el aprendizaje. Esta anécdota sirve para resaltar una ventaja de los lenguajes basados en bloques, ya que son fáciles de traducir a otros idiomas.

Finalmente, cuando se trató de ejercicios involucrando condicionales, el depurador de Physical Bits resultó ser muy útil para explicar la diferencia entre condicionales anidados y secuenciales. Ejecutar el programa paso a paso utilizando el depurador hizo que fuera muy fácil visualizar el flujo de ejecución. Esto solo fue posible en el grupo experimental, ya que el grupo de control tenía esta herramienta desactivada. Esta ventaja puede haber contribuido a la diferencia en el progreso de los estudiantes, como se verá a continuación.

VIII. RESULTADOS

Después de completar la actividad, se recopilaron los archivos de solución generados por los estudiantes y los archivos de log generados por la aplicación. A partir del procesamiento de estos archivos, se obtuvieron los siguientes conjuntos de datos:

- estudiantes: Información relacionada con cada uno de los estudiantes (19 registros)
- ejercicios: Información sobre la lista de ejercicios (70 registros)
- soluciones: Información sobre las soluciones a los ejercicios (491 registros)
- eventos: Información sobre los eventos registrados por la aplicación (332851 registros)

Dado el número limitado de participantes en el estudio y la duración de la actividad, los datos recopilados no son suficientes para obtener resultados estadísticamente significativos. Sin embargo, existen tendencias interesantes que vale la pena explorar en un estudio más grande.

RQ1 - Impacto en el progreso de los estudiantes

Para calcular el progreso de cada estudiante, se consideró el porcentaje de ejercicios resueltos sobre el total (70 ejercicios). Tanto para el grupo de control como para el grupo experimental, se observa un progreso similar, siendo el promedio más alto para el grupo experimental. Es notable también que el estudiante más avanzado en el grupo experimental logró resolver el 77.14% de los ejercicios, mientras que el estudiante más avanzado en el grupo de control solo alcanzó el 44.28%.

Dado que los ejercicios están ordenados siguiendo una curva de dificultad ascendente tiene sentido analizar el progreso de los alumnos encuentro por encuentro. De esta forma, se puede observar en la figura 1 que la mayor diferencia se presenta en el último encuentro, donde el grupo experimental tuvo en promedio un progreso 4 veces mayor que el grupo de control.

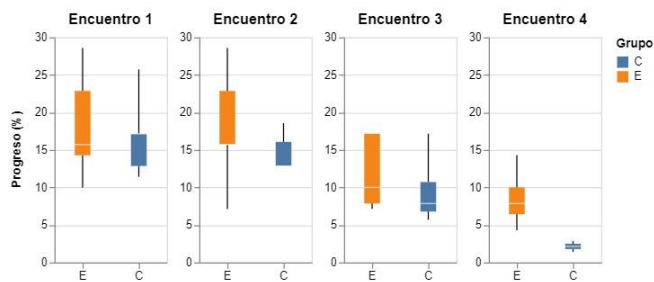


Fig. 1. Progreso de los estudiantes en cada encuentro

RQ2 - Impacto en el tiempo de resolución de cada ejercicio

Como puede observarse en la figura 2, no encontramos una diferencia significativa en el tiempo de resolución para los ejercicios iniciales, pero a medida que aumenta la dificultad, comienzan a surgir diferencias entre los grupos.

Los datos indican que, para los ejercicios más simples, no hay diferencias notables en el tiempo de resolución. Sin embargo, a medida que aumenta la complejidad de los

ejercicios, también se observa que el grupo de control tarda más en resolverlos, lo que sugiere que los beneficios de la programación interactiva son más pronunciados a medida que aumenta la complejidad del problema a resolver.

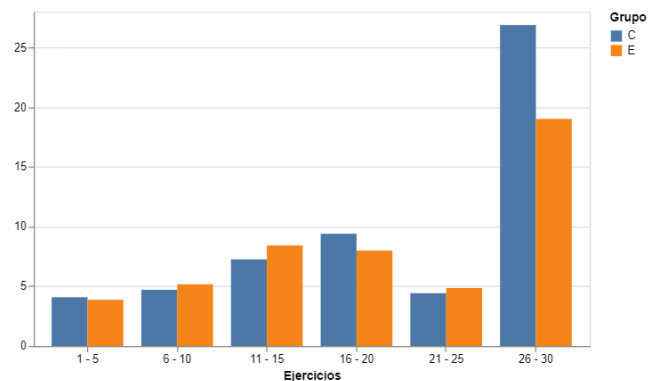


Fig. 2. Tiempo de resolución por ejercicio (mediana en minutos)

RQ3 - Impacto en la forma de trabajar de los estudiantes

Para analizar el trabajo de los estudiantes se utilizarán principalmente 2 métricas: el ritmo de trabajo y la eficiencia.

Physical Bits compila automáticamente el programa ante cada cambio, por lo que el tiempo entre compilaciones es una estimación razonable del ritmo de trabajo de cada estudiante. Como se observa en la figura 3, los estudiantes de ambos grupos trabajaron a un ritmo aproximadamente similar.

Por otro lado, la estimación de la eficiencia de cada estudiante se calculó contando cuántos ejercicios pudieron resolver por hora. En este caso, si observamos el resumen de todas las sesiones, vemos que, en promedio, la eficiencia fue muy similar entre ambos grupos, resultando en un rápido descenso a medida que avanzaban las sesiones. Coherente con los gráficos anteriores, esto tiene sentido considerando la creciente dificultad de los ejercicios.

Sin embargo, al analizar cada encuentro por separado, se puede observar que aunque el grupo de control muestra una eficiencia más alta en el primer encuentro, en las sesiones posteriores, su eficiencia promedio siempre está ligeramente por debajo de la del grupo experimental. La brecha aumenta, especialmente en el último encuentro, donde la eficiencia del grupo experimental triplicó la del grupo de control (figura 4).

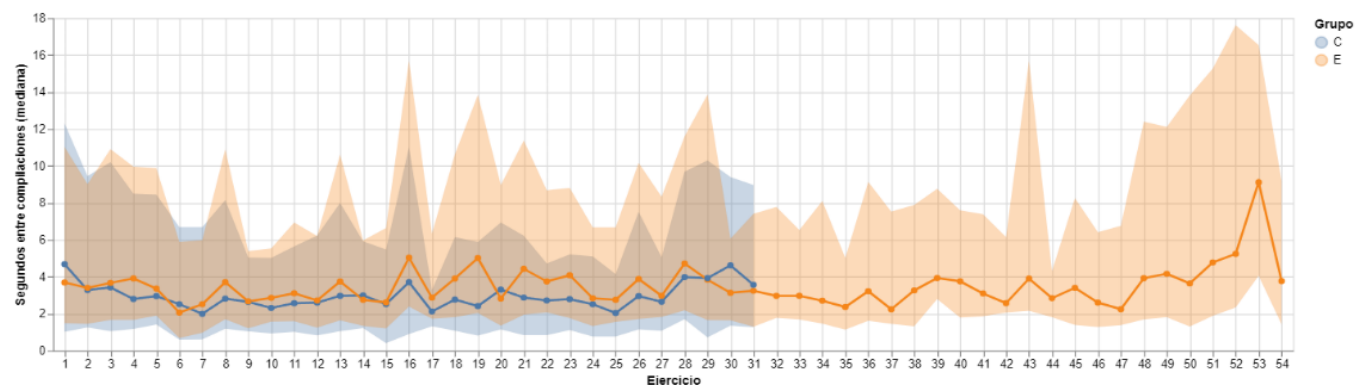


Fig. 3. Tiempo entre compilaciones por ejercicio (mediana en segundos)

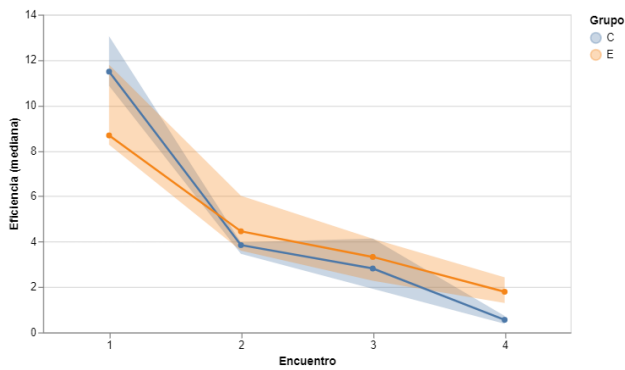


Fig. 4. Eficiencia de los estudiantes en cada encuentro

RQ4 - Impacto en el bucle de retroalimentación

La forma más sencilla de medir la duración del bucle de retroalimentación es considerar el tiempo desde que cada estudiante comienza a trabajar en un ejercicio hasta que puede ver el programa ejecutándose en el robot. En este sentido, se puede observar que el grupo experimental generalmente tardó mucho menos tiempo en ejecutar el programa por primera vez. Los datos indican que el tiempo hasta tener un programa en funcionamiento en el robot es, en promedio, un orden de magnitud menor para el grupo experimental que para el grupo de control (figura 5).

Otra forma de analizar la duración del bucle de retroalimentación es considerar las ejecuciones por minuto.

En este caso, la diferencia entre el grupo experimental y el grupo de control también es evidente (figura 6).

Finalmente, también puede calcularse el tiempo entre ejecuciones. Al igual que en los gráficos anteriores, el grupo experimental exhibe ejecuciones mucho más frecuentes que el grupo de control (figura 7).

Estos datos sostienen la importancia de la programación interactiva en la resolución de problemas: un entorno interactivo implica menos obstáculos para probar programas, lo que a su vez fomenta ejecuciones más frecuentes y rápidas. En general, esto redundará en un mayor progreso en la resolución de ejercicios.

IX. CONCLUSIONES

Este estudio piloto arroja luz sobre el impacto positivo de la programación interactiva en el rendimiento de los estudiantes en un taller de robótica. Los hallazgos enfatizan las ventajas de facilitar un feedback inmediato a las acciones de los estudiantes, lo que resulta en mejores oportunidades para la experimentación y un progreso más rápido en la resolución de problemas. Aunque nuestro estudio se centró en resultados de rendimiento en lugar de aprendizaje, sienta las bases para investigaciones futuras.

Es crucial reconocer, sin embargo, las limitaciones del estudio y la necesidad de una investigación más integral con un mayor número de participantes para establecer evidencia definitiva sobre las ventajas de la programación interactiva para actividades de robótica educativa.

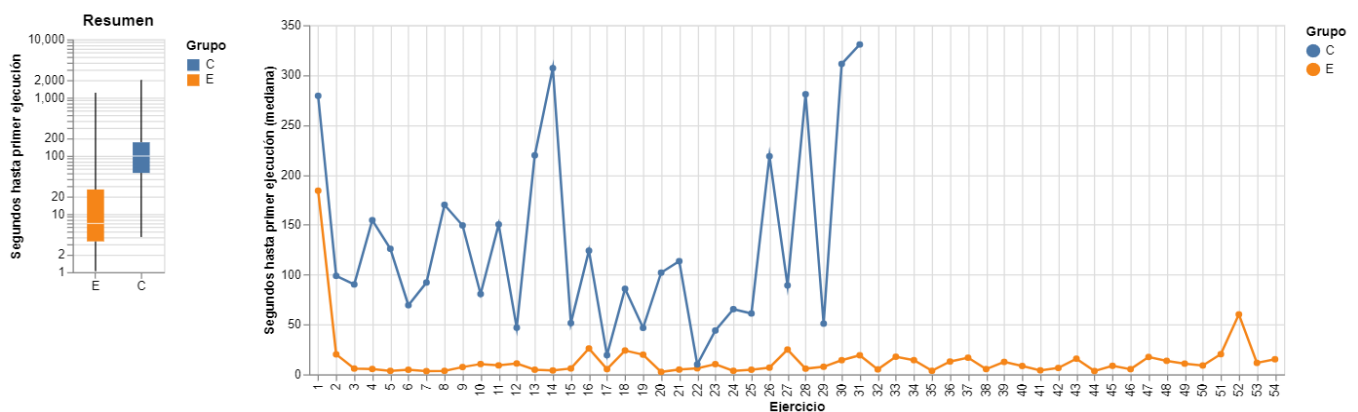


Fig. 5. Segundos hasta primer ejecución (izquierda: resumen de todos los encuentros; derecha: mediana por ejercicio)

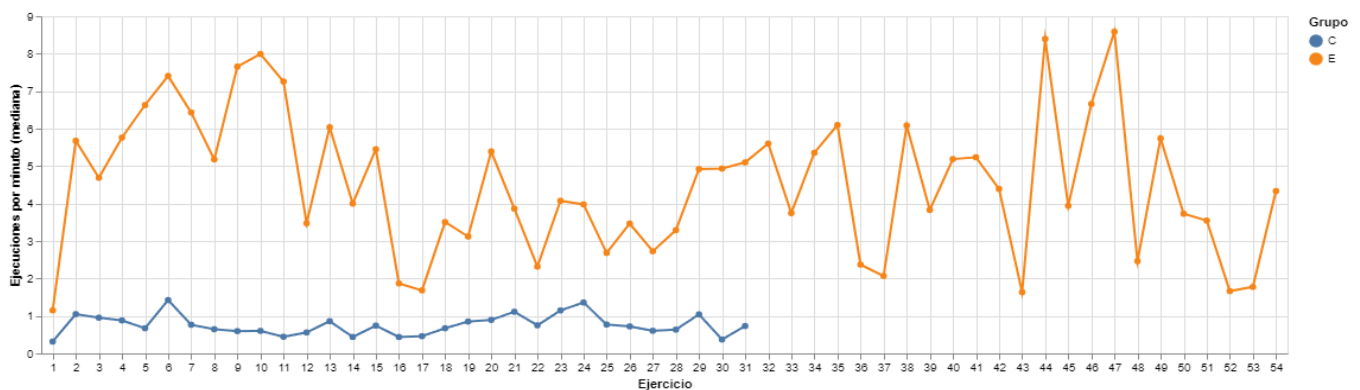


Fig. 6. Ejecuciones por minuto para cada ejercicio (mediana)

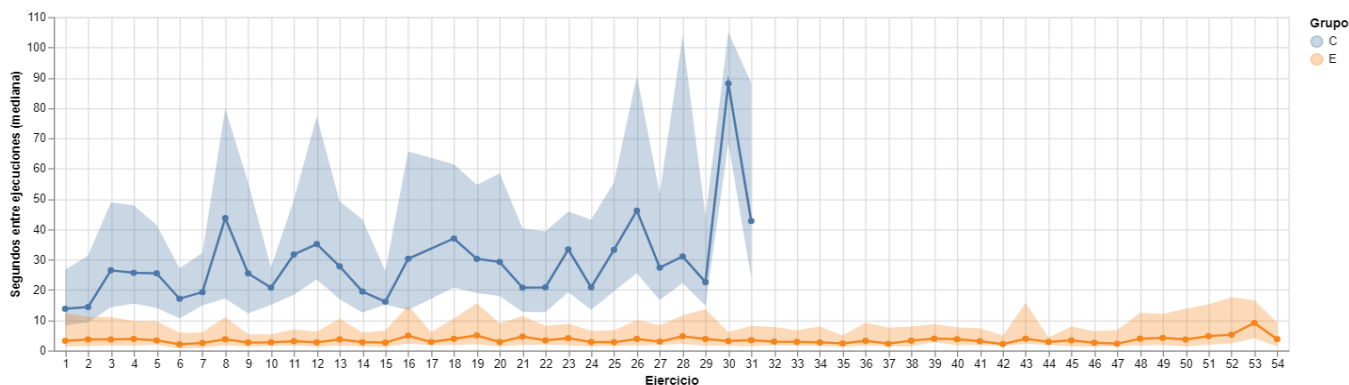


Fig. 7. Tiempo entre ejecuciones del programa para cada ejercicio (mediana en segundos)

REFERENCIAS

- [1] P. Rein, S. Ramson, J. Lincke, R. Hirschfeld, and T. Pape, "Exploratory and Live, Programming and Coding: A Literature Study Comparing Perspectives on Liveness," *Programming*, vol. 3, no. 1, p. 1, Jul. 2018, doi: 10.22152/programming-journal.org/2019/3/1.
- [2] S. Anwar, N. A. Bascou, M. Menekse, and A. Kardgar, "A Systematic Review of Studies on Educational Robotics," *Journal of Pre-College Engineering Education Research (J-PEER)*, vol. 9, no. 2, Jul. 2019, doi: 10.7771/2157-9288.1223.
- [3] J. Devine, J. Finney, P. de Halleux, M. Moskal, T. Ball, and S. Hodges, "MakeCode and CODAL: Intuitive and efficient embedded systems programming for education," *Journal of Systems Architecture*, vol. 98, pp. 468–483, Sep. 2019, doi: 10.1016/j.sysarc.2019.05.005.
- [4] "Electronics | Free Full-Text | HYDRA: Introducing a Low-Cost Framework for STEM Education Using Open Tools." Accessed: Jan. 30, 2024. [Online]. Available: <https://www.mdpi.com/2079-9292/10/24/3056>
- [5] P. Molins-Ruano, C. Gonzalez-Sacristan, and C. Garcia-Saura, "Phogo: A low cost, free and 'maker' revisit to Logo," *Computers in Human Behavior*, vol. 80, pp. 428–440, Mar. 2018, doi: 10.1016/j.chb.2017.09.029.
- [6] A. Pina and I. Ciriza, "Primary Level Young Makers Programming & Making Electronics with Snap4Arduino," in *Educational Robotics in the Makers Era*, D. Alimisis, M. Moro, and E. Menegatti, Eds., in *Advances in Intelligent Systems and Computing*. Cham: Springer International Publishing, 2017, pp. 20–33. doi: 10.1007/978-3-319-55553-9_2.
- [7] P. Plaza et al., "Scratch as Driver to Foster Interests for STEM and Educational Robotics," *IEEE Revista Iberoamericana de Tecnologías del Aprendizaje*, vol. 14, no. 4, pp. 117–126, Nov. 2019, doi: 10.1109/RITA.2019.2950130.
- [8] R. Moran, M. Teragni, and G. Zabala, "Physical Bits: A Live Programming Environment for Educational Robotics," in *Robotics in Education*, W. Lepuschitz, M. Merdan, G. Koppensteiner, R. Balogh, and D. Obdržálek, Eds., in *Advances in Intelligent Systems and Computing*. Cham: Springer International Publishing, 2021, pp. 291–303. doi: 10.1007/978-3-030-67411-3_26.
- [9] J.-P. Kramer, J. Kurz, T. Karrer, and J. Borchers, "How live coding affects developers' coding behavior," in *2014 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, Jul. 2014, pp. 5–8. doi: 10.1109/VLHCC.2014.6883013.
- [10] C. D. Hundhausen and J. L. Brown, "An experimental study of the impact of visual semantic feedback on novice programming," *Journal of Visual Languages & Computing*, vol. 18, no. 6, pp. 537–559, Dec. 2007, doi: 10.1016/j.jvlc.2006.09.001.
- [11] E. M. Wilcox, J. W. Atwood, M. M. Burnett, J. J. Cadiz, and C. R. Cook, "Does continuous visual feedback aid debugging in direct-manipulation programming systems?," in *Proceedings of the ACM SIGCHI Conference on Human factors in computing systems*, in *CHI '97*. New York, NY, USA: Association for Computing Machinery, Mar. 1997, pp. 258–265. doi: 10.1145/258549.258721.
- [12] M. Lodi, D. Malchiodi, M. Monga, A. Morpurgo, and B. Spieler, "Constructionist Attempts at Supporting the Learning of Computer Programming: A Survey," *Olympiads in Informatics*, vol. 13, pp. 99–121, Jul. 2019, doi: 10.15388/oi.2019.07.
- [13] D. Weintrop and U. Wilensky, "To block or not to block, that is the question: students' perceptions of blocks-based programming," in *Proceedings of the 14th International Conference on Interaction Design and Children*, Boston Massachusetts: ACM, Jun. 2015, pp. 199–208. doi: 10.1145/2771839.2771860.
- [14] L. Cabrera, J. H. Maloney, and D. Weintrop, "Programs in the Palm of your Hand: How Live Programming Shapes Children's Interactions with Physical Computing Devices," in *Proceedings of the 18th ACM International Conference on Interaction Design and Children*, in *IDC '19*. New York, NY, USA: Association for Computing Machinery, Jun. 2019, pp. 227–236. doi: 10.1145/3311927.3323138.
- [15] R. Moran, "RichoM/pbits-exercises." Jan. 05, 2023. Accessed: Apr. 04, 2024. [Online]. Available: <https://github.com/RichoM/pbits-exercises>
- [16] P. Brusilovsky, "Teaching Programming to Novices: A Review of Approaches and Tools," 1994. Accessed: Dec. 16, 2022. [Online]. Available: <https://www.semanticscholar.org/paper/Teaching-Programming-to-Novices%3A-A-Review-of-and-Brusilovsky/bf44088aaa3b47fe2081a9641fe34a8d715dcfac>
- [17] J. Rodríguez, G. Parra, D. Dolz, and R. Ramírez, "Transición desde programación basada en bloques a basada en texto: una revisión del campo," 2019. Accessed: Jul. 22, 2022. [Online]. Available: <http://sedici.unlp.edu.ar/handle/10915/91318>

Plantas experimentales para diseño de controladores SISO con fines educativos

Claudio Pose^{1,2} y Ezequiel Pecker-Marcosig^{1,3}

¹Laboratorio de Automática y Robótica, Facultad de Ingeniería - Universidad de Buenos Aires

²CONICET - Laboratorio de Inteligencia Artificial y Robótica - Universidad de San Andrés

³CONICET - Laboratorio de Simulación de Eventos Discretos - Universidad de Buenos Aires

Resumen—Este artículo presenta un conjunto de plantas experimentales construidas con el objetivo de funcionar como plataformas para el diseño e implementación de controladores SISO, con una dificultad adecuada para carreras de grado de ingeniería. Las mismas fueron concebidas para la realización de prácticas de laboratorio dentro del área de control automático. Se presentarán un conjunto de plantas construidas con ese fin, y un ejemplo de aplicación para el control de un péndulo simple y un péndulo rotativo.

Index Terms—Control SISO, Planta experimental, Educación, *Learning-by-doing*.

Line Follower Robot_with PID control

1st Bruno Chiabrando
Ingeniería Electrónica UTN-
FRSFCO
Universidad Tecnológica Nacional
Facultad Regional San Francisco
San Francisco, Córdoba Argentina
bchiabrando@facultad.sanfrancisco.utn.edu.ar

2nd Angel Aguilar
Ingeniería Electrónica UTN-
FRSFCO
Universidad Tecnológica Nacional
Facultad Regional San Francisco
San Francisco, Córdoba Argentina
aaguilar@facultad.sanfrancisco.utn.edu.ar

3rd Rodolfo Podadera
Ingeniería Electrónica UTN-
FRSFCO
Universidad Tecnológica Nacional
Facultad Regional San Francisco
San Francisco, Córdoba Argentina
rpodadera@facultad.sanfrancisco.utn.edu.ar

Abstract—Line Follower Robot is controlled by a proportional-integral-derivative system known as PID is developed, the parameters of this system can be configured at any time through a simple user interface (UI), also the error obtained by the TCRT5000 sensor array is plotted in real time. The ESP32 microcontroller with FreeRTOS is used to be able to perform the multiple tasks simultaneously, it controls sensors, actuators and the communication with the web server through HTTP Request/Response. The final device has a correct operation and a high fidelity when it comes to stabilizing in the changes of the circuit to follow, in this way it meets the main objectives, reaching greater efficiency than conventional speed robots.

Keywords—PID, Robot, Follower, FreeRTOS, HTTP

I. INTRODUCCIÓN

El presente trabajo busca realizar un robot que sea capaz de recorrer cualquier circuito que posea una línea central continua de manera autónoma. El dispositivo cuenta con un microcontrolador, encargado de controlar a través de un sistema proporcional – derivativo – integral (PID) la velocidad de los motores lo cual nos permite alcanzar una mayor eficiencia y velocidad a la hora de seguir la línea. Los parámetros de dicho controlador PID pueden ser monitoreados y modificados desde una interfaz de usuario (UI) la cual se puede acceder de manera local a través de un servidor web creado por el microcontrolador ESP32. Para alcanzar una mayor eficiencia tanto en el funcionamiento como en la comunicación con el usuario se hace uso de FreeRTOS que nos permite ejecutar las distintas tareas necesarias en simultaneo, optimizando los recursos brindados por el microcontrolador seleccionado.

II. OBJETIVOS

- ✓ Elaborar un control PID para el manejo de velocidades de los motores que permita el seguimiento de una línea.
- ✓ Programar la comunicación entre cualquier dispositivo con conexión a la red local y el microcontrolador ESP32 para la calibración inalámbrica del controlador del robot seguidor de línea.
- ✓ Alcanzar luego de pruebas y re calibraciones una mayor eficiencia que robots velocistas convencionales.

III. METODOLOGÍA DE DESARROLLO

A. Robots seguidores de línea

Son capaces de seguir un camino trazado por una línea central continua. La cual es de un color que contrasta con el del piso,

por lo general la línea es blanca y el resto del suelo es negro. Una de las virtudes de esta configuración es que se puede realizar de múltiples maneras y pueden ser tan complejos como la persona que lo realice quiera. Año tras año las tecnologías utilizadas para desarrollarlos evolucionan y uno de los motivos principales son las distintas competencias que se realizan. El robot desarrollado cumple con las reglas explícitas por la Liga Nacional de Robótica de Argentina por lo que se encuentra apto para competir en las próximas ediciones.

B. Chasis

Se fabricó de manera aditiva (impresora 3D) respetando las dimensiones máximas de los robots velocistas que deben ser de 20 cm de largo x 14 cm de ancho, con un límite en altura de 10cm. En la figura 1 se puede apreciar un render 3D del diseño.

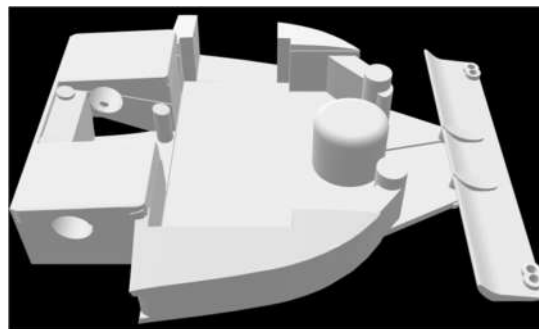


Figura 1. Render de chasis 3D.

C. Actuadores

Los mismos son los elementos principales del robot velocista, y de ellos depende el desempeño que se pueda alcanzar, aquí se utiliza un moto reductor pololu HP, 6V de corriente continua. El término reducción refiere a cuantas vueltas dará el motor para que los engranajes produzcan una vuelta a la salida del eje, de esta manera se incrementará el torque a costas de una reducción de velocidad.



Figura 2. Moto reductor.

D. Sensores

Para la detección de la línea y ubicación del robot se utilizaron ocho sensores ópticos infrarrojos TCRT5000 dispuestos en un circuito impreso y fueron trabajados de manera digital conectando cada uno de los sensores a una entrada digital y se los alimentó con 3.3 V.

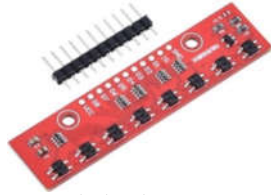


Figura 3. Arreglo de ocho sensores TCRT5000.

E. Circuito impreso de potencia

Para el control de los actuadores se diseñó un circuito encargado de recibir una señal de control PWM y alimentar a los motores en mayor o menor medida.

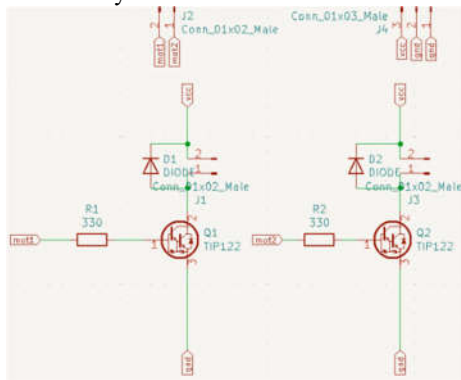


Figura 4. Esquemático para controlar actuadores.

El TIP122 está formado por un par de transistores NPN en conexión Darlington, su funcionamiento es igual al de un NPN. Excitando la base del mismo a través de una señal PWM podemos establecer un flujo de corriente entre emisor y colector y así alimentar a los motores.

En esta placa podemos encontrar el conexionado de los dos motores como la alimentación y la señal de PWM proveniente de la placa de control.

F. Microcontrolador

Como se dijo anteriormente para el control de este dispositivo se utiliza el microcontrolador de Espresif ESP32 en formato dev module. El mismo integra Wifi, Bluetooth, los GPIOs necesarios y también brinda la posibilidad de uso de sistema operativo FreeRTOS, que nos permite administrar un sistema multitareas robusto para obtener aprovechar de la manera más óptima el algoritmo.



Figura 5. Microcontrolador.

G. Circuito impreso de control

Para controlar el robot se diseñó una segunda placa en donde va ubicado el módulo ESP32 junto con sus periféricos, los

distintos pines para conexionado de sensores, motores, y alimentación de la placa de potencia.

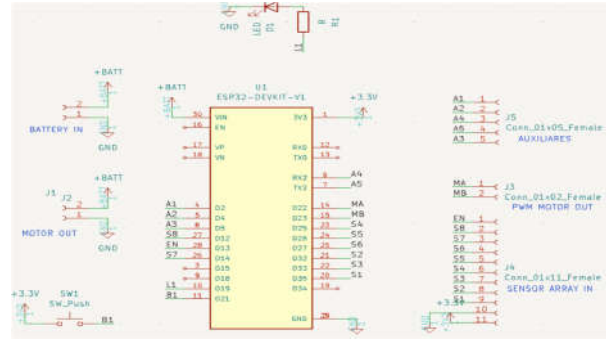


Figura 6. Esquemático Circuito de Potencia.

IV. CONTROLADOR PID

El control PID (control proporcional, integral y derivativo) es un control por realimentación que calcula la desviación o error entre un valor medido y el valor que se quiere obtener (set point), para aplicar una acción que corrija el proceso. En el caso del robot seguidor de línea, el controlador PID (rutina basada matemáticamente), procesa los datos del sensor y los utiliza para controlar el ángulo de giro a través de la velocidad enviada a cada motor, para de esta forma mantenerlo en curso.

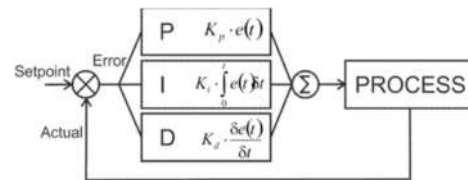


Figura 7. Esquema controlador PID (Aprendiendo facilelectrónica-2014).

Para el análisis de nuestro robot seguidor de línea se toma de base la estructura básica de un sistema de control de lazo cerrado. Lo cual en una representación en bloques queda como se observa.



Figura 8. Diagrama de Sistema de control a lazo cerrado.

Cada una de las etapas presentes se modela de manera virtual a través de MATLAB para poder realizar una simulación del robot en su totalidad.

A. Modelado de cinemática directa del robot

Comenzamos con el modelado del movimiento del robot, para esto se asumieron una serie de consideraciones para simplificar el análisis: el desplazamiento es causado únicamente por la rotación de las ruedas y la superficie en donde se desplaza es totalmente lisa. En este caso se toman en cuenta condiciones ideales para simplificar el modelo al considerar las pequeñas dimensiones del móvil. En una configuración diferencial se consideran tres grados de libertad: las distancias x e y de posición y el ángulo ϕ de orientación. Para poder lograr un movimiento controlado, se deben tener dominio sobre "Vr" la velocidad de la rueda derecha y "Vl" la velocidad de la rueda izquierda. El objetivo de modelar el robot es buscar una relación directa de cómo afectan las entradas "Vr y Vi", a los estados del sistema "x,

y, φ ". Podemos observar los distintos grados de libertad, y se resalta su dependencia de la velocidad de cada motor.

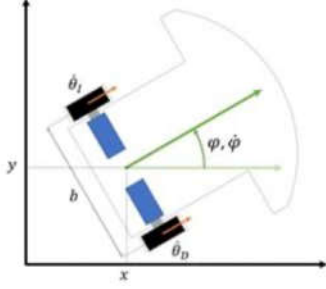


Figura 9. Movimiento de robot tipo tracción diferencial.

Una vez definidas las entradas y salidas de nuestro sistema, notamos que nuestro robot puede trasladarse con una velocidad lineal " v " y rotar con una velocidad angular " ω ".

$$v = r (Vr + Vl) / 2 \quad (1)$$

$$\omega = r (Vr - Vl) / b \quad (2)$$

Siendo r = Radio de la rueda. b = distancia entre las ruedas Vr = Velocidad motor derecha. Vl = Velocidad motor izquierda. Teniendo las velocidades definidas planteamos las distintas ecuaciones que describen el movimiento del robot.

$$x = v * \cos \varphi \quad (3)$$

$$y = v * \sin \varphi \quad (4)$$

$$\varphi = \omega \quad (5)$$

Trabajando algebraicamente con las ecuaciones alcanzadas llegamos a las ecuaciones 6, 7 y 8 que determinan el movimiento a partir de la tracción diferencial.

$$x = [(Vr + Vl) r \cos \varphi] / 2 \quad (6)$$

$$y = [(Vr + Vl) r \sin \varphi] / 2 \quad (7)$$

$$\varphi = [(Vr - Vl) r / b] \quad (8)$$

Al integrar las ecuaciones (6), (7) y (8) se consigue la posición y orientación en un sistema de referencia absoluto. Utilizando programación a bloques en Simulink MATLAB modelamos la planta y nos queda como se observa en la figura 10.

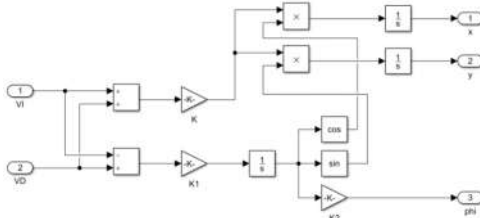


Figura 10. Subsistema, cinemática de la planta.

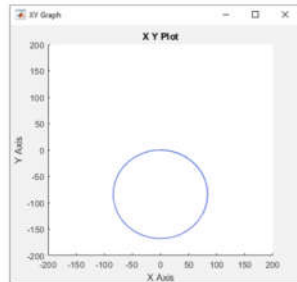


Figura 11. Desplazamiento en el plano.

Una de las primeras pruebas realizadas fue ingresar dos velocidades diferentes y graficar su posición en x e y a través

de un gráfico XY y obtuvimos que el robot describe un movimiento circular tal como se observa en la figura 11 de esta manera verificamos que el modelado descrito responde correctamente con lo esperado.

B. Modelado de motores de corriente continua

El movimiento del robot, se realiza con los motoredutores de 6V de corriente continua, cuyas características extraídas de hoja de datos se pueden apreciar en tabla 1:

Parámetro	Símbolo	Valor
Relación	η	30:1
Voltaje nominal	$V_{nominal}$	6V
RPM sin carga	Ω_{free}	104.72 rad/s
Corriente máxima	i_{max}	1.6 A
Corriente nominal	i_{free}	100 mA
Torque máximo	T_{max}	0.0557 Nm

Tabla 1. Tabla de valores extraídas de datacheet motor DC

Para obtener el modelo se analiza la velocidad angular de cada rueda a partir del circuito equivalente para un motor de corriente directa, tal como se muestra en la figura 12.

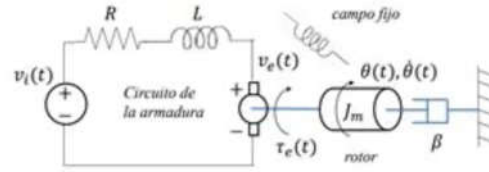


Figura 12. Circuito electromagnético de motor DC

Debido a que los parámetros proporcionados por el fabricante pertenecen a la salida del tren de engranes, se reducirá el estudio al considerar solamente un momento de inercia. A continuación, se obtienen las ecuaciones diferenciales de la parte eléctrica por la ley de voltajes de Kirchoff y de la parte mecánica por la ley de Newton.

$$Ri(t) + L di(t)/dt + v_e(t) = v_i(t) \quad (9)$$

Las ecuaciones del subsistema magnético son:

$$T(t) = P(t) / \omega_m(t) = k_t I(t) \quad (10)$$

$$e(t) = k_e \omega_m(t) = k_e d\theta(t)/dt \quad (11)$$

Reemplazando (11) aplicando Laplace a la ecuación (9), obtenemos la ecuación para la parte eléctrica.

$$R I(s) + L s I(s) + k_e s \theta(s) = V_i(s) \quad (12)$$

Por otro lado, para la parte mecánica del motor aplicamos las leyes de Newton para movimientos mecánicos rotacionales y obtenemos la ecuación 13.

$$T(t) = J_m d^2\theta(t)/dt^2 + b d\theta(t)/dt \quad (13)$$

$$\mathcal{L}[T(t)] = T(s) = J_m s^2 \theta(s) + b s \theta(s) \quad (14)$$

Reemplazando 10 en 14 se obtiene:

$$k I(s) = J_m s^2 \theta(s) + b s \theta(s) \therefore k I(s) / (J_m s + b) = \theta(s) \quad (15)$$

Reemplazando 15 en 12 llegamos a:

$$R I(s) + L s I(s) + k_e [k I(s) / (J_m s + b)] = V_i(s) \quad (16)$$

Finalmente, para obtener la relación buscada entre la salida del motor y la entrada de voltaje tal como se observa en la ecuación 17.

$$\begin{aligned} \Theta(s)/V_i(s) &= \\ k_t / [(R + Ls)(J_m s + b) + k_e k_t] &= \\ k_t / [J_m L s^2 + (J_m R + b L)s + R b + k_e k_t] \end{aligned} \quad (17)$$

Las características eléctricas de Resistencia e Inductancia de armadura se obtienen a través de la medición del dispositivo físico, mientras que el voltaje de entrada será el valor nominal que especifica el fabricante. $R = 31,3 \, \Omega$ $L = 5,1 \, \text{mH}$ $v_i(t) = 6V$. Los valores de las constantes eléctricas y magnéticas los calculamos con los datos presentados en la tabla 1 correspondientes a los actuadores utilizados.

$$k_e = v_{\text{nominal}} / \Theta_{\text{free}} = 0.0573 \, \text{v} / (\text{rad/seg}) \quad (18)$$

$$k_t = T_{\text{max}} / I_{\text{max}} = 0.034 \, \text{N m / A} \quad (19)$$

Los parámetros mecánicos de momento de inercia y la constante de fricción al ser más difíciles de cuantificar, se hicieron de uso de unos valores pequeños obtenidos por un motor similar al utilizado para este trabajo, los valores son los siguientes:

$$J_m = 1 \times 10^{-5} \, \text{kgm}^2 \quad \beta = 1 \times 10^{-7} \, \text{Nm/(rad/s)}$$

Una vez obtenidos los modelados de estos bloques que son de vital importancia para nuestro sistema, unimos cada uno de ellos y finalmente obtenemos un diagrama de nuestra función de transferencia final. Como vemos en la figura 13. En nuestro diagrama se presenta un voltaje inicial base con el cual ambos motores se ven afectado en un inicio, nuestra retroalimentación va a afectar negativamente al voltaje de entrada del motor izquierdo y positivamente al voltaje de entrada del motor derecho y de esa manera logrará el giro deseado.

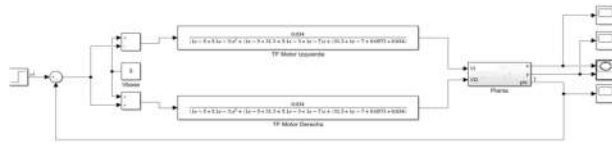


Figura 13. Simulación del sistema en Simulink

Al iniciar la simulación con un escalón unitario en su entrada obtenemos la respuesta que se puede observar en la figura 14, en la misma se presenta una oscilación elevada previa a su estabilización, esta es la que se corregirá cuando se aplique el controlador PID.

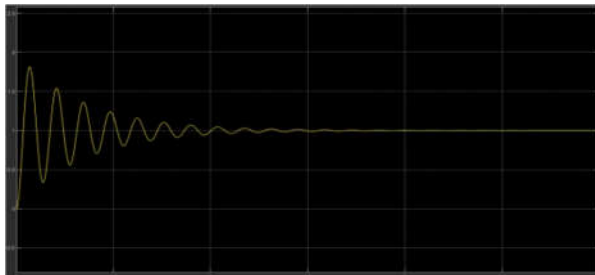


Figura 14. Respuesta al escalón de lazo cerrado (Simulink)

Para obtener una única función de transferencia de nuestro sistema se pueden aplicar técnicas de simplificación o algún software que facilite cálculos, dicha ecuación ya resuelta es la 20.

$$\frac{1.27e07s^2 + 7.814e10s + 4.872e11}{s^5 + 1.227e04s^4 + 3.774e07s^3 + 4.696e08s^2 + 1.464e9s} \quad (20)$$

Como vemos en la figura 15 la respuesta al escalón calculada con Matlab es la misma a la obtenida anteriormente, por lo que podemos verificar que la función de transferencia obtenida es verdaderamente la que describe a nuestro sistema. Además, se observa la respuesta en lazo abierto en donde se ve como no converge en un punto.

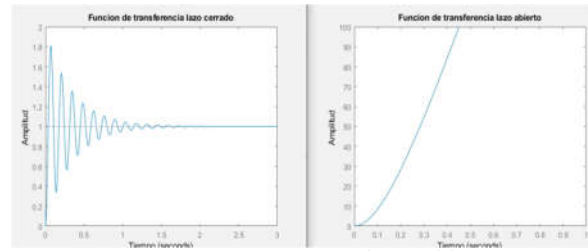


Figura 15. Respuesta a escalón (Matlab)

Una vez obtenida esta respuesta hacemos uso de nuestro controlador PID para mejorar la estabilización de nuestro sistema, a través de Matlab llegamos a la respuesta que se observa en la figura 16, esta respuesta fue obtenida sin hacer uso del término integrador del controlador ya que no existe error estacionario y con las constantes $K_p = 0.01$ y $K_d = 0.001$.

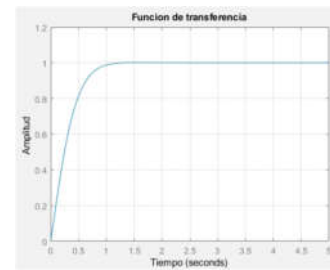


Figura 16. Respuesta con controlador PID (Matlab)

Junto con la respuesta obtenida se calculó el lugar geométrico de las raíces además del mapa de polos y ceros, estos gráficos se pueden observar en la figura 17, la cual nos va a ser útil para realizar algunos análisis.

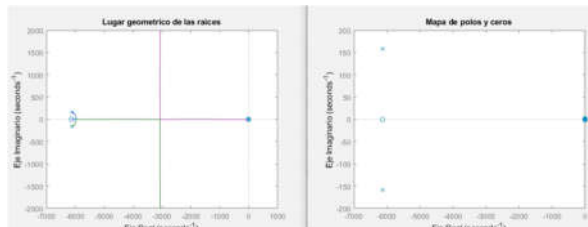


Figura 17. Gráficos de polos y ceros (Matlab)

En estos gráficos podemos observar tanto la posición de los polos y las raíces de nuestro sistema como los posibles valores que pueden tomar al modificar alguno de los parámetros. Al ser una función de transferencia de grado 5 el análisis puede llegar a ser más complejo de lo habitual, pero podemos notar que nuestro sistema no puede caer en la inestabilidad ya que los posibles valores que pueden tomar son negativos. Las respuestas que se pueden alcanzar son “Sub-amortiguado”, “críticamente amortiguado” o “sobre amortiguado”. Si analizamos, además, el origen vemos que existe la presencia de un polo en él, esto es así ya que en la función de transferencia (ecuación 20) no tenemos un término independiente en el denominador. En términos matemáticos esto significa que la función de transferencia ya posee un término integrador $1/s$, lo que ocasiona que no sea necesario

utilizarlo en el controlador PID. Por otro lado, también se obtuvo el diagrama de bode correspondiente a nuestro Sistema como se aprecia en la figura 18, con el efecto de sus polos y ceros y donde se puede analizar la frecuencia de cambio de set point a la que se ve afectada la fase y la amplitud.

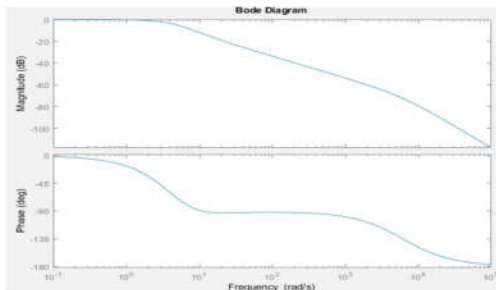


Figura 18. Diagrama de bode del sistema (Matlab)

Tal como se realizaron los cálculos y simulaciones para un tiempo continuo se hicieron también, para un tiempo discreto obteniendo respuestas similares (figura 19).

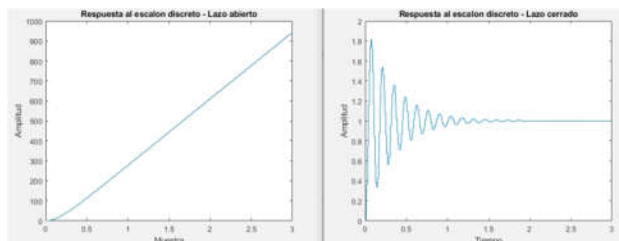


Figura 19. Respuesta al escalon discreto (Matlab)

De la misma manera que en la simulación continua, en la discreta le aplicamos un controlador PID lo calibramos y obtenemos una salida como la que se muestra en la figura 20. Las constantes de calibración alcanzadas son $K_p=130$, $K_d=1$ y nuevamente no se hizo uso del término integrador.

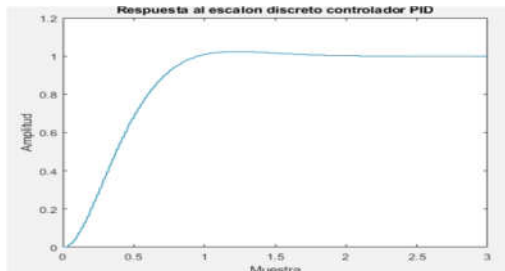


Figura 20. Respuesta discreta con controlador PID

Con todos los valores calculados, podemos finalizar la construcción de la simulación en Simulink, uniendo los bloques modelados junto al controlador PID como indica la figura 21. Se puede observar el ángulo de giro y el desplazamiento en el plano de nuestro robot simulado a través de XY ploter.

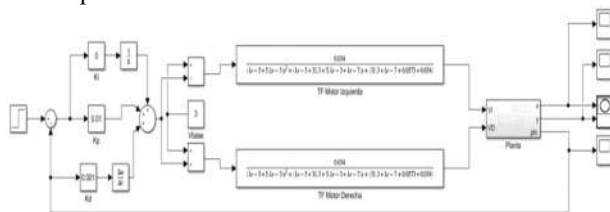


Figura 21. Line follower Robot modelado

V. FIRMWARE

El código se hizo de manera modular, separando las funcionalidades en librerías para un mejor manejo y simpleza a la hora de realizar modificaciones.

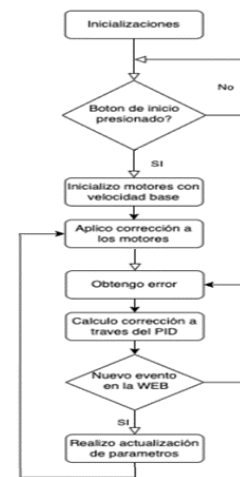


Figura 22. Diagrama de flujo del código

En la figura 22 podemos observar de una manera resumida las distintas tareas que tiene que ejecutar el micro. Para dar inicio al programa, por reglamentación de la Liga Nacional de Robótica, se debe presionar un pulsador y al liberarlo recién pueden inicializarse los motores. Para realizar este proceso se utilizó una máquina de estado encargada de eliminar los posibles rebotes al pulsar y de inicializar los motores en el momento que se libera el botón. Para el caso de la lectura de los sensores se hizo una librería utilizando programación orientada a objetos para facilitar la utilización de las distintas funciones a lo largo del código.

Figura 23. Interfaz de usuario

La interfaz de usuario mostrada en la figura 23 se realizó con HTML, estilizándolo con CSS, dándole funcionalidad con JS. Estos tres archivos se encuentran guardados en la memoria flash de la ESP32 a través de SPIFFS que es un sistema de archivos ligero creado para microcontroladores.

En la misma interfaz se graficará el error sentido por el robot en tiempo real, para así poder ver cómo responde la corrección de nuestro controlador PID. Una vez calculado el ajuste haremos uso de nuestra librería para el manejo de los motores, la misma le brinda a cada uno de ellos la misma velocidad de partida a través de dos canales de PWM con una frecuencia de 5Khz y 10 bit de resolución dando la posibilidad de un PWM de 0 a 1023.

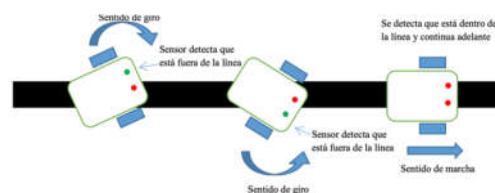


Figura 24. Esquema de detección de línea – Extraída de Martínez Ortiz, David (2015-2016)

El ajuste calculado por el PID se aplicará de manera positiva en el canal PWM correspondiente al motor de la derecha y negativamente al motor de la izquierda, debido a la manera en que planteamos el error.

En la figura 24, se puede observar cómo reacciona el robot al detectar que está fuera de línea (su set point), a través de la corrección calculada por el PID genera una diferencia de velocidad entre los dos motores logrando así un giro en un sentido determinado.

El algoritmo para realizar dicha corrección se ejecuta de manera continua, utilizando el tiempo transcurrido para calcular tanto el término derivativo como el integral, y el proporcional es obtenido con el error sensado. Para hacer uso de este código se necesita trabajar de manera asincrónica haciendo uso de distintos núcleos o de tareas en simultáneo. Para el control general de las distintas etapas del robot se programó una máquina de estados encargada de monitorear y comandar al robot, esta es una de las tareas principales y corre en el núcleo 1 del microcontrolador.

```
int PIDUpdate(int error){
    currentTime = millis();
    elapsedTime = (double)(currentTime - previousTime);
    cumError += error * elapsedTime; // La integral es el error acumulativo en el tiempo
    rateError = (error - lastError)/elapsedTime; // La derivada es la tasa de cambio del error

    int out = Kp*error + Ki*cumError + Kd*rateError;

    lastError = error;
    previousTime = currentTime;

    return out;
}
```

Código 1. Algoritmo de controlador PID

En simultáneo una tarea se ejecuta en el núcleo 2, para que permita obtener el error a través de los sensores, el cual será enviado al controlador PID para que realice los cálculos correspondientes y se obtenga la corrección de velocidad que será aplicada a los motores para alcanzar nuevamente el set point.

El prototipo final se puede ver en la figura 25. En el chasis se encuentran montados todos los periféricos que se nombraron anteriormente, los motores, el array de sensores, los circuitos impresos y las baterías interconectadas.

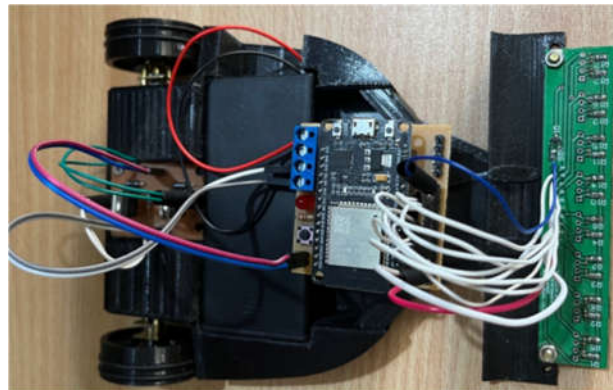


Figura 25. Prototipo Final

VI. RESULTADOS

En el primer encendido del prototipo se debió obtener a prueba y error la velocidad de partida necesaria para que el robot se traslade sobre una línea recta y que dicho valor no sea demasiado alto como para que salga de pista. Una vez configurado el valor de velocidad de partida se calibraron los parámetros del controlador PID, primeramente, la constante del término proporcional Kp, la misma presentó un correcto funcionamiento al valor de 50, en el caso de Ki al no existir

error estacionario se dejó en 0, y Kd dio buenos resultados en un valor de 4. Esto se pudo calibrar de una manera sencilla desde la interfaz de usuario.

VII. CONCLUSIONES

La interfaz de usuario permite una configuración inalámbrica y facilita la rapidez y facilidad de la calibración.

La posibilidad de monitorear gráficamente el error actual en todo momento brinda facilidad para deducciones y nuevas calibraciones. Tanto el algoritmo del controlador PID como el de FreeRTOS permite una rápida acción de los motoredutores, ya que con cada lectura de los sensores en pista puede predecir los errores y usar los anteriores para mejorar su funcionamiento y, por tanto, la autonomía del movimiento de cada motor sea independiente según lo requiera.

Gracias al análisis de los polos y ceros se detectó que el termino integrador ya estaba en la función de transferencia que describía al sistema, ya que no existía término independiente en el denominador lo que generaba un polo en el origen. Esto describe un integrador mecánico, que en nuestro sistema se presenta ya que al realizar una corrección tras un error sensado no vuelve al origen si no que se mantiene en esa corrección. Al usar nuestro prototipo por un tiempo prolongado notamos que la respuesta de nuestro sistema varía, exigiendo una nueva calibración para volver a su comportamiento ideal. Esto es así ya que el sistema con el cual estamos trabajando no es invariante en el tiempo, luego de un determinado periodo su fuente de alimentación varía, ocasionado por la descarga de las baterías lo que lleva a una des calibración del sistema.

VIII. REFERENCIAS

- [1] Parikh, P., Shah, H., & Sheth, S. (2014, June). A Mechatronics design of a line tracker robot using Ziegler Nichols control technique for P, PI and PID controllers. In International Mechanical Engineering Congress (IMEC-2014) (pp. 13-15).
- [2] Martínez Ortiz, David (2015-2016). Robótica para seguimiento de líneas (robotics for line tracking). Escola Técnica Superior d'Enginyeria de Telecomunicació de Barcelona (pp. 1-106).
- [3] López Alarcón, Luis David (2021). Implementación de un algoritmo de control para un robot line follower usando red neuronal. Universidad Politécnica Salesiana Sede Guayaquil. (pp. 1-107).
- [4] L. E. Solaque Guzmán, M. A. Molina Villa y E. L. Rodríguez Vásquez, «Seguimiento de Trayectorias con un Robot Móvil de Configuración Diferencial,» Ing. USBMed, vol. 5, n° 1, 2014.
- [5] J. I. Collazo Cuevas, E. Gorrostieta Hurtado, J. C. Pedraza Ortega, U. G. Villaseñor Carrillo, R. A. Romero Torres y M. A. González Aguirre, «Modelación de un Robot Móvil de Dos Ruedas con Tracción Diferencial,» 2009.
- [6] Aucatoma Tiban Jairo Patricio (Febrero 2022). Diseño E Implementación De Un Robot Velocista Controlado Mediante Wifi Utilizando Una Tarjeta De Desarrollo Esp32- Wroom. Instituto Superior Tecnológico Vida Nueva (pp. 27-40).
- [7] Aprendiendofacilelectronica. (2014). «Robot Velocista de competencia Algoritmo PID,» Diciembre 2014. [En línea]. Available: http://aprendiendofacilelectronica.blogspot.com/2014/12/robot-velocista-decompetencia_4.htm