Multimedia Tools and Applications
https://doi.org/10.1007/s11042-025-20748-8

G-ARM: An open-source and low-cost robotic arm integrated with ROS2 for educational purposes

Julio Vega[1] · **Vidal Pérez**[1]

Received: 23 October 2024 / Revised: 28 January 2025 / Accepted: 28 February 2025

Abstract
The high cost of industrial robots limits their accessibility in academic settings. This research addresses this by developing a low-cost, 3D-printable robotic arm for educational use, designed using the open-source tool FreeCAD and affordable hardware components. The robot is integrated with ROS 2 Humble and MoveIt 2, enabling motion planning and control, and includes a simulator for virtual testing and prototyping. The robot was evaluated over eight months in the Industrial Robotics and Software Architectures for Robots courses at Rey Juan Carlos University. It demonstrated durability, ease of use, and practical feasibility, making it a suitable platform for training robotics professionals. This work highlights the potential of affordable robotics to enhance education and provides a scalable, replicable solution for academic institutions.

Keywords ROS · Low-cost · MoveIt · Robotic arm · Educational robotics · 3D printing · STEM learning · Autonomous systems · Open-source robotics · Teaching tools

1 Introduction

Currently, robotics is present in numerous sectors, encompassing a wide variety of applications. This versatility has led to the emergence of various groups and categories that allow us to classify different types of robots. A clear example of this is industrial robotics, which revolutionizes manufacturing by integrating advanced robotic systems into production processes. These highly specialized machines are designed to perform tasks with precision, speed, and consistency, ultimately enhancing efficiency and productivity in various industries. From automotive assembly lines to electronics manufacturing, industrial robots handle a wide range of applications, including welding, painting, material handling, and assembly. Their flexibility and programmability allow for seamless adaptation to evolving production demands, driving innovation and streamlining operations.

Vidal Pérez contributed equally to this work.

✉ Julio Vega
julio.vega@urjc.es

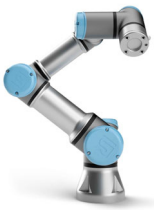
Vidal Pérez
v.perez@urjc.es; v.perezb.2019@alumnos.urjc.es

[1] Universidad Rey Juan Carlos, Madrid, Spain

Published online: 26 March 2025

(a) Turtlebot 2

(b) Dobot Magician



(c) UR3e

Fig. 1 Robots used in universities

With cutting-edge technologies such as artificial intelligence and machine learning, industrial robotics continues to push the boundaries of what's possible in modern manufacturing, promising a future where automation plays an increasingly vital role in shaping the industrial landscape. However, as the demand for skills in robotics and automation increases, the need to train more people in this field becomes evident, with educational robotics being a fundamental tool for this purpose.

For this reason, the field of educational robotics has gained great relevance in recent years, due to the increasing need to educate the new generations in technological skills [16, 17]. In secondary schools, robotics has become an effective pedagogical tool for developing skills and knowledge in areas such as programming, mathematics, electronics, and problem-solving. This is known as STEM (Science, Technology, Engineering, and Mathematics). Thanks to this training, students learn to design, build, and program simple robots to perform specific tasks, which helps them understand science and technology concepts in a more practical and interactive way [2] [1] [2] [3].

At the university level, robotics has become an essential discipline for educating future engineers [4, 12]. Students learn to design and build more advanced robots, and they reinforce their programming skills and control of complex systems. In these institutions, various types of robots are used, such as industrial arms and mobile robotic platforms as shown in Fig. 1. Since these are systems used in the professional world, students can learn with robots similar to those they will use in the future.

However, an inherent problem with this training is that, although some units of these professional robots are usually available in classrooms, their use is often limited by their high economic cost, and therefore, their availability is restricted. This deviates from the ideal scenario where each student could have their own unit to practice freely and comfortably. As a result, there is a growing need for affordable robots in education [3, 15].

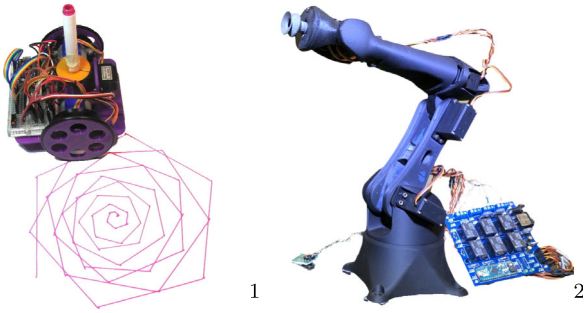
In the context of educational robotics, several key criteria must be addressed to ensure these robots effectively support teaching and learning. These criteria include:

1. Cost-effectiveness: Low-cost robots are essential for widespread adoption in classrooms, as expensive industrial robots can limit accessibility. Affordable robotic solutions allow institutions to equip more students with practical learning tools, democratizing access to advanced robotics education [16].
2. Accessibility and Ease of Use: The robot must be easy for students to operate, with a simple interface and minimal setup requirements. This ensures that students can focus on

[1] https://www.turtlebot.com/turtlebot2/

[2] https://www.robotlab.com/store/dobot-robotic-arm

[3] https://www.universal-robots.com/es/productos/robot-ur3/



(a) OSTR (b) Diy-robotics 6-DOF

Fig. 2 Low-cost robots

the educational aspects of the robot, such as programming and problem-solving, rather than on the complexities of the system [17].

3. Educational Value: Educational robots must help students develop critical STEM skills, including problem-solving, programming, and systems integration. The robot should be capable of supporting a range of learning activities that align with these objectives, such as designing, building, and programming robots to perform specific tasks [2, 5].
4. Robustness and Durability: Given the high frequency of use in educational settings, the robot must be durable and capable of withstanding the wear and tear of regular handling by students. This is critical to ensuring continuous learning without the interruptions caused by hardware failures [11].
5. Adaptability and Integration with Educational Software: To maximize educational outcomes, the robot should integrate easily with widely used platforms such as ROS2 (Robotic Operating System) and MoveIt 2, which provide robust tools for motion planning, simulation, and programming. This compatibility allows students to work on real-world problems, ensuring that the learning process is closely aligned with industry standards [7].
6. Scalability: The robot should be scalable to accommodate a range of skill levels. From basic programming to advanced control systems, students should be able to progressively use the same robotic platform as their knowledge and skills grow, ensuring continuity throughout their educational journey [8].
7. Diversity of Applications: The robot must be versatile enough to support a variety of applications, from automation and control theory to artificial intelligence. This adaptability ensures that the robot can be used across multiple disciplines, giving students the flexibility to explore different areas of robotics [9].

The present work focuses on the development of a small-sized and low-cost industrial robotic arm, designed with the purpose of facilitating its use and control in university teaching, specifically in the subjects of the Degree in Software Robotics Engineering at the Rey Juan Carlos University. This robot has been completely 3D printed and is integrated with tools widely used in the field of robotics, addressing all the key criteria outlined above[4] [5] (Fig. 2).

[4] https://www.instructables.com/Low-Cost-Arduino-Compatible-Drawing-Robot/

[5] https://diy-robotics.com/educative-robot-cell-free-package/?utm_source=Instructables.com&utm_medium=referrer&utm_campaign=Educative%20Cell

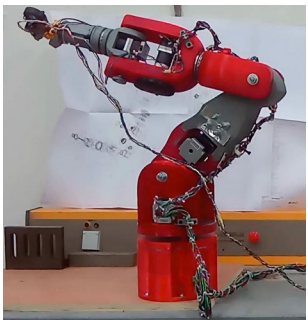


Fig. 3 Robot HydraX

2 Related work

In this section, some of the most relevant works, that explore industrial robots for educational purposes, are described. Various robotic arm solutions are analyzed based on key criteria such as cost, ease of assembly, integration with educational software, and performance. A comparative synthesis is provided to identify existing gaps that justify the development of a new robotic arm solution.

Several studies have investigated the feasibility of 3D-printed robotic arms for educational environments [6, 14, 18]. For instance, Krimpenis et al. [10] introduced HydraX (Fig. 3), a 6-DOF robotic arm designed for machining applications. With a reach of nearly one meter and a weight of approximately 13 kg, it offers a robust platform for precision tasks. However, its relatively high cost (over *1000* €), lack of integration with the Robot Operating System (ROS), and time-consuming assembly process (200 hours) present challenges for educational adoption (Table 1).

Adediran et al. [1] proposed UIArm (Fig. 4), a smaller, 4-DOF robotic arm aimed at sorting plastic bottles. The arm features an open-source design and is primarily 3D printed, which simplifies production and reduces costs. Despite its accessibility, the robot's limited workspace and potential accuracy issues due to 3D-printed gears restrict its applicability for more complex educational tasks (Table 2).

Another notable contribution is the Niryo One robotic arm (Fig. 5), which was developed through a crowdfunding campaign on Kickstarter and integrates ROS 1 and MoveIt [13]. This 6-DOF arm is designed for educational purposes and supports automation through various programming interfaces. However, its relatively high cost (approximately *1000* €) and reliance on proprietary software such as SolidWorks limit its widespread adoption, especially in cost-sensitive educational institutions (Table 3).

MeArm (Fig. 6) is another open-source robotic arm widely used in educational settings due to its simplicity and affordability. Designed for beginners, it features an easy assembly

Table 1 HydraX Robot Strengths and Weaknesses

Strengths	Weaknesses
Repetitiveness of $\pm$ 0.04 mm	Lack of ROS integration
Large workspace	Material cost exceeding 1000 €
More than 3 DOF, various orientations	Requires 200 hours to print and assemble
Max load of 12 kg, practical at 3 kg	Proprietary SolidWorks files

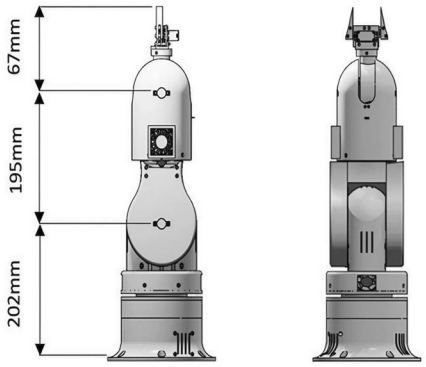


Fig. 4 Robot UIArm

process and a lightweight structure. However, its reliance on small servos and the absence of bearings reduce precision and durability, making it less suitable for advanced robotics courses (Table 4).

A recent development in the field is *LeRobot*, an affordable educational robotic arm costing only $110. This robot offers a complete AI-based control toolchain and is designed to integrate seamlessly with machine learning courses. Despite its cost-effectiveness and AI capabilities, its limited degrees of freedom and hardware constraints may restrict its use in more complex robotic applications (Table 5).

Table 6 presents a comparative analysis of these robotic arms based on key criteria relevant to educational applications.

The review highlights several limitations across the existing robotic solutions, including high costs, limited degrees of freedom, lack of ROS2 integration, and low precision. These limitations underscore the need for a new cost-effective robotic arm that is easy to assemble, fully compatible with ROS2, and offers improved precision and robustness. The proposed robotic arm in this work aims to address these challenges by providing an affordable, 3D-printed solution tailored for university courses in industrial robotics.

3 Software and hardware tools

This section provides a concise overview of the software and hardware tools used to meet the research objectives, emphasizing the key choices made based on the project criteria.

3.1 Software tools

The software tools selected for this project were chosen for their ease of use, seamless integration, and flexibility in control. Well-established, widely supported, and flexible, these

Table 2 UIArm Robot Strengths and Weaknesses

Strengths	Weaknesses
Open-source design	Limited workspace
Uses small motors with integrated reducers	Accuracy issues with 3D-printed gears
Fewer parts reduce assembly time	Lack of ROS integration

Fig. 5 Robot Niryo One

Table 3 Niryo One Robot Strengths and Weaknesses

Strengths	Weaknesses
Open-source with full documentation	Higher cost for 3D-printed version (1000 €)
Integration with ROS1 and MoveIt	Uses Raspberry Pi, increasing costs
6 DOF with repeatability of 0.5 mm	Not officially adapted for ROS2
Can adapt to various tools	Designed using proprietary SolidWorks

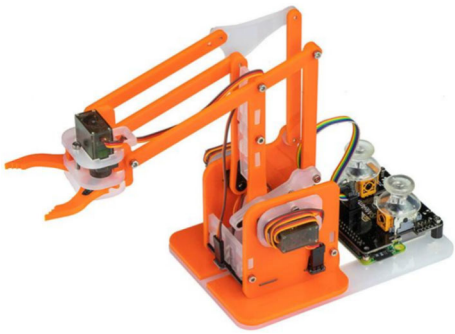


Fig. 6 MeArm open-source robotic arm

Table 4 MeArm Robot Strengths and Weaknesses

Strengths	Weaknesses
Highly accessible	Limited force due to small servos
Easy assembly with many guides	Lack of bearings reduces lifespan
Low center of mass for rapid movements	Vibrations under load affect performance
Lightweight design	Limited precision

Table 5 LeRobot Strengths and Weaknesses

Strengths	Weaknesses
Affordable at $110	Limited degrees of freedom
AI-based control toolchain	Limited hardware capabilities compared to more expensive models
Easy to build and use	May not be suitable for industrial applications
Seamless integration with AI	Simple design may limit complex applications

Table 6 Summary of Robotic Arms in Educational Settings

Robot	Cost	DOF	Integration with ROS	Educational Value
HydraX	1000+	6	No	High
UIArm	Low	4	No	Moderate
Niryo One	€1000	6	Yes (ROS1)	High
MeArm	Very Low	3	No	Moderate
LeRobot	$110	4	Yes (AI-based control)	High

tools are ideal for educational environments, offering scalability and cost-effectiveness while meeting the project's specific needs.

- Robot Operating System (ROS 2): ROS 2 was chosen for its modularity, robust communication capabilities, and decentralization, which was a significant improvement over the original ROS. The adoption of ROS 2 enhances system reliability and allows for the integration of various robotic components in a seamless manner. ROS 2's node-based architecture is ideal for managing multiple tasks concurrently, which is crucial for educational environments where students must engage with complex systems.
- MoveIt!: MoveIt! was incorporated to enable motion planning and manipulation tasks, which are essential for educational applications in industrial robotics. Its integration with ROS 2 allows for simplified programming of advanced tasks like grasping and object manipulation. The flexibility of MoveIt! makes it suitable for various robots and tasks, aligning with the project's goal of providing a general-purpose robotic arm for educational use.
- FreeCAD: FreeCAD was selected for designing the robotic components due to its parametric modeling approach, which offers flexibility in adjusting designs and creating parts that are easy to modify. FreeCAD's open-source nature and wide community support also made it a practical choice for creating customizable parts, which is essential in an educational and research context where modifications and iterations are common.
- G-code and Grbl: The project leverages G-code programming for controlling the robotic arm's movements. Grbl firmware was adapted to control the motors, with particular adjustments made to accommodate the rotary joints of the robotic arm. This choice aligns with the need for cost-effective, open-source solutions, enabling easy control of the robotic arm using standard CNC software.

3.2 Hardware tools

The hardware choices for the project were made with a focus on affordability, accessibility, and the ability to meet performance requirements. Components were selected for their cost-effectiveness, ease of use, and reliability, ensuring the robotic arm can be built and operated efficiently in an educational setting without the need for extensive technical expertise.

- MKS DLC32 motherboard: This open-source motherboard was chosen for its compatibility with Grbl, ability to control three motors, low cost, and ease of integration, making it ideal for educational robotics. Compared to the Arduino CNC Shield V3, the MKS DLC32 offers superior features, including an advanced ESP32 microcontroller for faster processing and a high-power MOSFET for efficient 24V output control via PWM. Its design reduces electrical assembly errors and integrates components effectively on a PCB

base plate, ensuring robust performance and cost-effectiveness with only a minimal price difference.

- Nema 17 stepper motors: These motors were chosen for the robotic arm due to their reliability, availability, and sufficient torque of 0.65 Nm, providing the precision and control needed for educational tasks. Unlike conventional brushed motors, which lack holding torque when stationary, stepper motors maintain significant torque and enable accurate angular position tracking through discrete steps, even without encoders. Though relatively heavy, their cost-effectiveness makes them ideal for this application. A mathematical analysis, illustrated in Fig. 7 and detailed in (1), determined the minimum torque required for the arm's most demanding scenario, with the arm fully extended.

$$\begin{aligned} T1 &= (m1 * (L1/2) + m2 * (L1 + L2/2) + payload * (L1 + L2)) * g \\ T2 &= (m2 * (L2/2) + payload * (L1 + L2)) * g \end{aligned} \tag{1}$$

Equaiont 1 Calculation of the necessary torque for the most demanding joint

The robot's links, each 17 cm long and weighing 200 g, require minimum torques of 1.66 Nm and 0.92 Nm for the first and second joints, respectively. Nema 17 stepper motors, chosen for their affordability and availability, are used in a 60 mm length to provide 0.6 Nm of torque each. While a single motor cannot meet the torque requirements alone, a gearbox amplifying the torque by a factor of five ensures these motors can deliver the necessary performance, balancing cost-effectiveness with functionality.

- Gearbox: To achieve precise and reliable force transmission while reducing speed, the prototype uses toothed belts instead of gears due to their superior precision and constant tension, avoiding the play that gears can introduce. The GT2 belt type is chosen for its affordability and suitability, with a 6 mm width meeting the application's needs, and lengths detailed later. Cost-effective and customizable GT2 pulleys include a mix of commercial and 3D-printed designs. A metal pulley with 20 teeth and a 3D-printed one with 100 teeth provide a 1:5 reduction for two degrees of freedom, while a 120-teeth pulley achieves a 1:6 reduction for the third, ensuring precise, efficient movement.
- TMC2209 stepper drivers: These drivers were selected for their advanced features, including low noise operation, high current handling, and compatibility with Nema 17 motors, ensuring smooth and precise motor control essential for educational robotics. While the maker community offers various interchangeable stepper motor drivers for low-current applications (Table 7), the TMC2209 was chosen over alternatives like the DRV8825 for its superior technology. It provides quieter operation, improved signal accuracy, reduced step loss, and greater energy efficiency, aided by a larger heat sink that minimizes heat production, making it an optimal choice for the project.
- Power supply: A 24-volt, 5-amp (120W) power supply was selected to ensure stable and reliable power delivery to the robot's components, accommodating the estimated total power consumption of approximately 35W. This supply exceeds the minimum require-

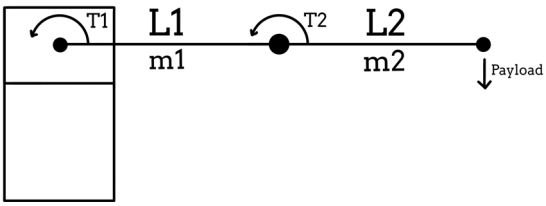


Fig. 7 Dynamics of the fully extended manipulator

Table 7 Comparison of specifications for existing drivers

Model	Supply Voltage	Current Max	Microstepping	Noise Level	Price
A4988	8-35V	2A	Up to 1/16	Very noisy	1€
DRV8825	8.2-45V	2.5A	Up to 1/32	Acceptable noise	2€
TMC2225	4.7-36V	2A	Up to 1/32	Low noise	2.8€
TMC2209	5.5-28V	2.5A	Up to 1/256	Completely silent	3.4€

ment, offering flexibility and efficiency, as higher voltages reduce current and manage voltage drops better. While any 12-24V supply capable of delivering over 40W could suffice, the chosen 24V power supply ensures robust performance, meeting the 21W required by the motors, 4W for the microcontroller and drivers, and 3W for the electromagnet, with ample capacity for operational stability.

- Other components: The project incorporates flanged bearings to enhance joint efficiency and ensure smooth rotation. These bearings, commonly used in 3D printers, are cost-effective and designed with a flange that secures integration into 3D-printed parts, preventing dislodgement when properly fixed. Microswitches are employed for end-of-travel detection and position establishment due to their affordability and sensitivity. Additionally, an electromagnet (model D20H15) is used during testing to enable the robot to manipulate metallic objects. With a diameter of 20 mm, height of 15 mm, and a lifting capacity of up to 3 kg, the electromagnet exceeds the robot's expected payload requirements, making it a practical choice for the application.

3.3 System hardware and software relationships

To better understand the interaction between the hardware and software components in the proposed robotic system, the structure outlined below describes how key software elements, such as ROS 2, MoveIt, and G-code, work in tandem with the hardware components, including motors, controllers, and power supplies.

Hardware Layer

- Motors (Nema 17): These are the actuators that move the robot's joints, powered by the stepper drivers.
- Stepper Drivers (TMC2209): These translate the commands sent from the control system to the motors, managing the motor's operation.
- Microcontroller (MKS DLC32): This board acts as the intermediary between the hardware and the software, receiving commands from the software and controlling the motors and other hardware components.
- Power Supply: A 24V power supply, which powers the entire system, ensuring that the motors and other components receive the required energy.

Software Layer

- ROS 2: The central framework that ties all the software components together. ROS 2 facilitates communication between software modules and the hardware.
- MoveIt!: Integrated with ROS 2, MoveIt! handles the motion planning and execution for the robotic arm, allowing for precise control over the robot's movements.

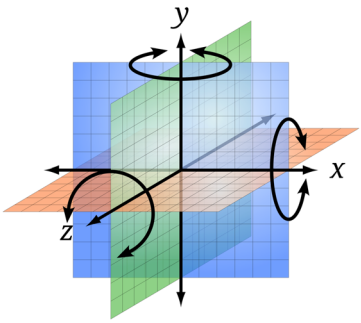


Fig. 8 Three-dimensional space

- G-code: The motion control commands, typically generated by MoveIt! or other planning tools, are interpreted by the firmware on the microcontroller to drive the hardware.
- Custom Python Scripts: Python is used for scripting, orchestrating the interactions between ROS 2, MoveIt!, and the hardware, ensuring smooth control flow.

4 G-Arm design

In this section, a comprehensive account of the development process is provided, detailing each stage from the initial conceptualization of the idea to the final realization of a fully functional robotic arm.

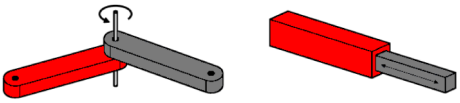
4.1 Manipulator geometry

Initially, it is essential to determine the number of degrees of freedom (DOF) required for the manipulator. In the context of three-dimensional space, as illustrated in Fig. 8, the robot requires 6 degrees of freedom: 3 for positioning (`X`, `Y`, `Z`) and 3 for orientation (`RX`, `RY`, `RZ`). Consequently, a minimum of 3 degrees of freedom is necessary to enable the robot's end effector to be positioned at any arbitrary point within the three-dimensional space.

In light of these considerations, a six-degree-of-freedom design was chosen instead of a more complex configuration to adhere to the constraints on cost and complexity as outlined in requirements 1 and 6. Subsequently, the selection of joint types became critical. In robotics, while multiple joint types are available, the two most commonly utilized are:

- Revolute Joints (Fig. 9a): Facilitate rotational movement around a fixed axis.
- Prismatic Joints (Fig. 9b): Enable linear sliding movement along a specific axis.

Revolute joints were implemented for their simplicity and reduced component count, aligning with the goals of cost-effectiveness and ease of assembly.



(a) Revolute Joint (b) Prismatic Joint

Fig. 9 Commonly used joint types in robotics

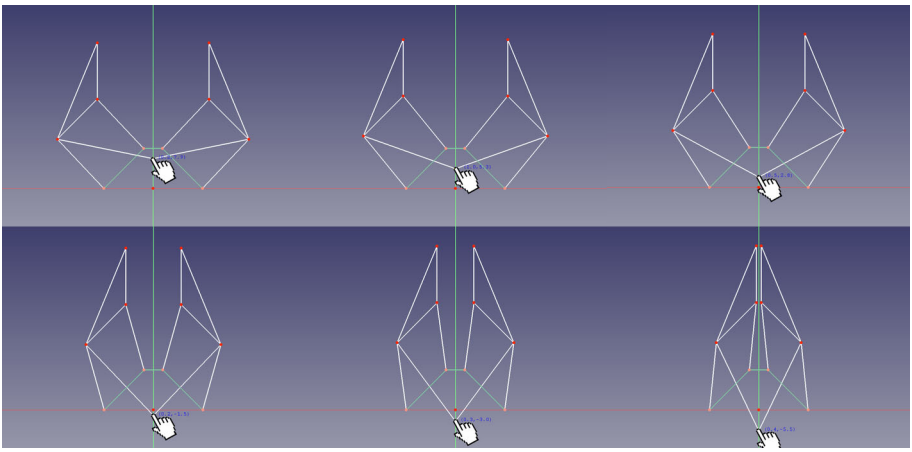


Fig. 10 Parallel gripper with 1 degree of freedom

In designing the manipulator's geometry, it is essential to consider the constraints imposed by having only three degrees of freedom. While this configuration allows for reaching any point within a three-dimensional space, it does not provide control over the orientation of the end effector at that point. To address this limitation, existing robotic solutions with three degrees of freedom were investigated, focusing on their approaches to handling orientation control. Two notable solutions emerged: SCARA robots and parallel robots based on parallelogram structures. Both types share the feature of maintaining the end effector parallel to the ground, effectively fixing two of the three possible orientations at any given position.

To achieve control over the final orientation, specifically the rotation around the Z-axis, a fourth degree of freedom is typically required. Although the design does not include this additional degree of freedom, tools such as electromagnets or suction cups will be used to facilitate simple object manipulation tasks. The approach involves implementing a parallelogram-based robot that operates on the same principle as those used in palletizing applications, where objects are placed on a pallet in a systematic and efficient manner.

4.2 Wireframe model of the manipulator

The wireframe model serves as a method for analyzing the movement of mechanical systems composed of axes and links. This model simplifies the visual representation of the system by emphasizing the spatial relationships between different components using lines and symbolic connections. An example of this approach is illustrated in Fig. 10.

For the development of the wireframe model used in this project, the MeArm robot model was adopted, as illustrated in Fig. 6. Specifically, the wireframe model designed for the Mechatronics[6] course at Rey Juan Carlos University was used.

The robot is characterized by several parameters: `L1`, `L2`, `P1`, `P2`, `P3`, and `A`, as shown in Fig. 11. These parameters define the arm's behavior and must be optimized through experimentation. The following procedure was employed to determine their values:

1. Initially, the lengths of the primary links `L1` and `L2` were set. For a desktop-sized robot, 17 centimeters per link was selected as a starting value.

[6] https://github.com/myTeachingURJC/Mecatronica/wiki/S3:-Estructuras-mec%C3%A1nicas-(II)

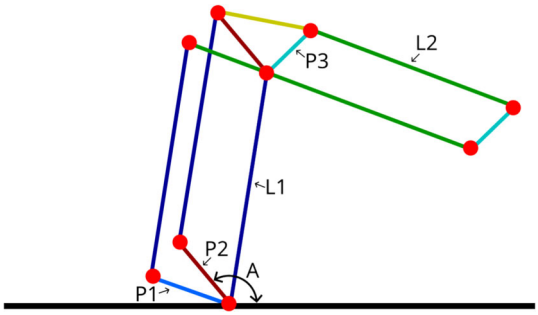


Fig. 11 Fundamental principle of the MeArm arm

2. The next step involved selecting `P1`, `P2`, and `P3`, which represent the shorter sides of the parallelograms. These lengths were adjusted to maximize their size without causing interference, ensuring that the parts do not collide during operation.
3. Finally, the angle `A` of the parallelogram, which keeps the end effector parallel to the ground, was set. An initial angle of 120°was chosen, as this value is critical for the arm's range of motion and accessibility near its base.

Figure 12 illustrates the final wireframe model of the manipulator, named G-Arm[7], with the parameter values provided in Table 8.

4.3 Sketches

Before advancing to the Computer-Aided Design (CAD) phase, a miniature prototype was developed as a proof of concept. This process began with creating a detailed sketch based on the wireframe model described in the previous section. The sketch was then digitized and 3D printed. The evolution of this process is illustrated in Fig. 13.

The miniature prototype, which was entirely 3D printed, successfully demonstrated both functionality and robustness, thus validating the initial concept. This prototype facilitated a better understanding of the spatial arrangement of each component, which proved to be highly beneficial for the CAD design process detailed in Section 4.4.

4.4 CAD design

The following describes the design considerations for the 3D-printed mechanical components, with a focus on the robotic arm project. The design process utilized both *Fusion 360* and *FreeCAD*, with the latter ensuring the project is open-source and parameterized, facilitating community access and modification.

Main base The main base component (Fig. 14 left) serves as the foundation for the robot, anchoring it to the ground and housing the base plate, drivers, and electronics, excluding the power supply which can be powered from multiple sources. Designed for easy 3D printing, it features two circular pieces with wide spacers for added robustness. The upper circular piece includes holes and hexagonal recesses for M3 nuts to secure a 120-tooth pulley and a

[7] Named after the programming language used for communication

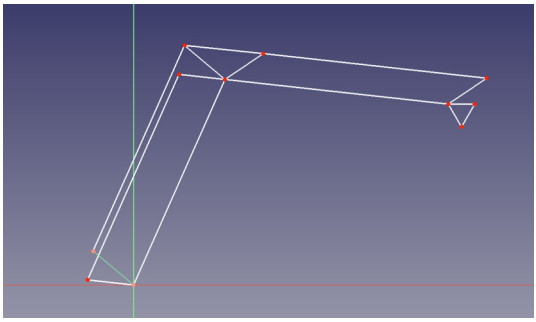


Fig. 12 Wireframe model of G-Arm

40mm recess for mounting a 4010 fan. The spacers are perforated cylinders for M5 screws, while the lower circular piece has mounting holes for the base plate and additional holes for securing the robot to the ground.

Motor base This assembly (Fig. 14 right) which houses the three motors and rotates on the main base, consists of two side pieces, a bottom piece, and a reinforcing spacer. It is designed for horizontal 3D printing to enhance resistance and accuracy. The assembly is secured with threaded rods and nuts and features a belt tensioning system with adjustable motor positioning. It also includes holes for limit switches and slots for organized cable management.

Parallelograms The wireframe model components, designed in 3D, transfer movement from the motor base to the robot's end, featuring slots for cable guidance and an end stop that activates at the travel's end, along with an adjustable sliding block for precision. Each joint is stabilized with press-fitted flange bearings. For the GT2 timing pulleys, a parametric tool was used to generate the DXF outline, which was imported and extruded in FreeCAD, with hexagonal holes and cutouts added for M3 nuts to secure the pulleys. Additionally, the robot's name is engraved on a visible link for personalization (Fig. 15 left).

Terminal element This component, illustrated in Fig. 15 right, is situated at the end of the robot and serves as the connection point to the tool. Its design features a 45-degree groove with a through-hole for a screw, ensuring a solid and definitive attachment. The angled walls of the groove center the tool, preventing rotation and play, thus providing a robust and precise connection.

Electromagnet Tool To equip the robot with functionality, a tool incorporating an electromagnet has been developed, as shown in Fig. 16. This tool is designed to fit into the robot's

Table 8 Parameters of the G-Arm Wireframe Model

Parameters	Values
L1	170mm
L2	170mm
P1	35mm
P2	35mm
P3	25mm
A	135°

(a) Final Sketch (b) Design in FreeCAD (c) Post-printing (d) Prototype Size

Fig. 13 Evolution of the proof of concept

end using the previously described groove shape. Additionally, the corners have been rounded to ensure a better visual fit with the robot's end. The assembly utilizes the electromagnet's threaded hole for secure attachment.

4.5 Printing and assembly

The following outlines a step-by-step process for replicating the G-Arm project, focusing on the 3D printing process. It includes detailed assembly instructions and specifications to ensure the accurate replication and proper functionality of the robotic arm.

The G-Arm was printed using an Ender-3 Pro 3D printer and 1kg of Red PLA filament. The 3D printed parts were designed with a layer height of 0.12mm and three wall lines for durability and precision, with varying fill densities ranging from 15% to 100% depending on the structural requirements. Key components include the lower base (15% fill), base spacer (100% fill), upper base (15% fill), 120T pulley (20% fill), motor base (15% fill), and various other parts such as limit adjusters and bearing spacers. The robot's name is also engraved on a visible link for personalization.

The project also requires the following screws: M3 screws in 12mm (8), 16mm (11), 20mm (10), and 25mm (13) lengths; M4 screws in 30mm (3); and M5 screws in 25mm (1), 30mm (3), 40mm (1), 50mm (5), 55mm (1), and 60mm (2) lengths. The necessary nuts include 6 M3 standard nuts, 29 M3 locknuts, 8 M4 locknuts, 5 M5 standard nuts, and 10 M5 locknuts. For washers, 29 M3 washers and 16 M5 washers are required. Additionally, threaded rods of 1 meter each are needed for M3, M4, and M5 sizes for assembly purposes.

Table 9 specifies the components that need to be purchased. The total estimated cost for the required components is €171, comprising €156.3 for the parts and approximately €15 for additional hardware. The printing process for all parts is expected to take around 60 hours, while assembly can be completed in about 2 hours, depending on the user's expertise. For

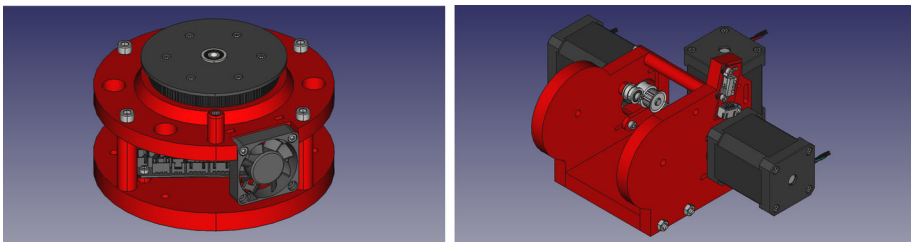


Fig. 14 Main base and motor base

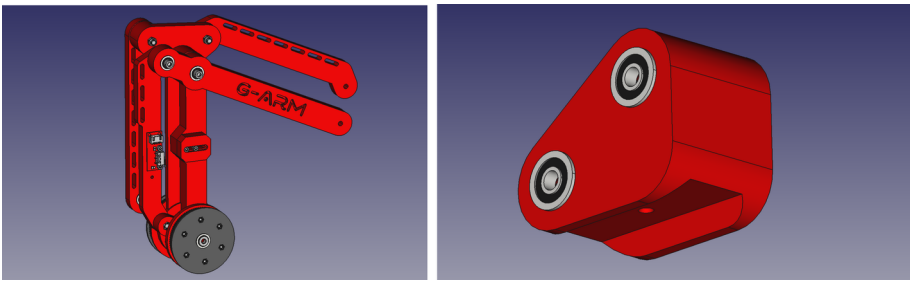


Fig. 15 Parallelogram assembly and terminal element

detailed assembly instructions, including the position of each part and the necessary screws, it is recommended to consult the complete assembly guide[8].

The assembly of the electronics is straightforward. Each connector on the printed circuit board is clearly labeled for ease of assembly. The primary consideration is the current adjustment of the three TMC2209 drivers, which involves using a small potentiometer and a multimeter to measure the values. Figure 17 illustrates the completed robot, which matches the CAD design in both appearance and functionality. The cable management was efficiently handled due to the inclusion of designated slots in the design.

5 Software for G-Arm

This section explores the software development for the G-Arm robot, focusing on the integration of Grbl firmware and the ROS 2 framework. It also discusses the challenges and solutions encountered during the development process.

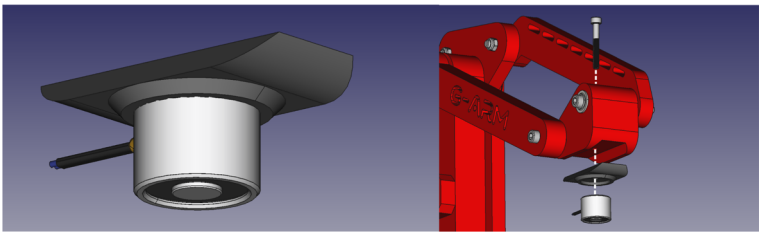
5.1 Grbl for actuator control

The G-Arm's actuator control is based on Grbl firmware, which is widely known for its reliability in CNC machines and stepper motor control. Although initially designed for CNC applications, Grbl was adapted to meet the specific needs of the G-Arm, including its requirement for more than three degrees of freedom. Key considerations included configuring Grbl to handle rotational movements and achieving the necessary precision for robotic tasks.

Challenges and Limitations of Grbl Adapting Grbl, which is optimized for linear movement along fixed axes, to the G-Arm's multi-DOF design required significant adjustments. Parameters such as steps per unit and maximum speed had to be calibrated for accurate motion control. A custom interface was also developed to enable communication between the G-Arm and Grbl, as there was no existing simulator available for pre-assembly testing. This meant that adjustments had to be tested in real-time, complicating debugging and performance prediction.

Communication with Grbl Grbl supports multiple communication interfaces, including USB (via serial UART) and Wi-Fi (via Telnet). The USB interface was chosen for its faster data transmission, facilitating responsive control. A Python class was developed to simplify communication between the software and Grbl, abstracting technical details and offering a

[8] https://github.com/RoboticsURJC/tfg-vperez/blob/280861172bce3b1c0cfbb155a434364ea68eeb30/src/design/FreeCad/%230_ASSEMBLY.FCStd



(a) Part (b) Assembly

Fig. 16 Electromagnet Tool

streamlined API for actuator control and position retrieval. Although the full code is available in the project repository[9], this class was pivotal for the G-Arm's software integration.

Configuring Grbl for Robotics Adapting Grbl for the G-Arm required fine-tuning several configuration parameters. For instance, parameters like steps per unit and maximum speed were adjusted to accommodate the robotic arm's specific motion requirements. Some important settings included:

- **$1**: Set to 255 ms to keep stepper motors energized and prevent mechanical collapse.
- **$100**, **$101**, and **$102**: Defined steps per unit for the X, Y, and Z axes, customized for the arm's rotational joints.
- **$110**, **$111**, and **$112**: Set maximum speeds based on experimental safety and performance tests.
- **$120**, **$121**, and **$122**: Defined acceleration limits to ensure safe movement.

These settings were crucial for enabling precise motion control, and further details can be found in the project repository[10].

5.2 Integration with ROS 2

Integrating the G-Arm with the ROS 2 ecosystem enables enhanced motion planning, control, and simulation capabilities. This section describes the integration process, focusing on the creation of the robot's description, its simulation within ROS, and its connection with the MoveIt 2 framework.

Robot description In ROS 2, the G-Arm's design was represented using the URDF (Unified Robot Description Format) and Xacro files, which describe the robot's geometry, joints, and properties. These files were generated from CAD models exported in the Collada (.dae) format, ensuring accurate representation of the robot's structure. Additionally, simplified collision meshes were created for effective interaction with the simulation environment. The robot description was visualized and interacted with using RViz, which allows for dynamic motion planning and control within both simulated and real environments.

Integration with MoveIt 2 MoveIt 2, a powerful motion planning framework for ROS 2, was used to handle the G-Arm's motion planning and trajectory execution. The integration

[9] https://github.com/RoboticsURJC/tfg-vperez/blob/96fc1e44bef6a31c272fb0673b5a33a7571c5ee7/src/software/g_arm/g_arm/g_arm_lib/grblAPI.py

[10] https://github.com/RoboticsURJC/tfg-vperez/blob/280861172bce3b1c0cfbb155a434364ea68eeb30/src/software/grblTests/grblConfig.txt

Table 9 Required hardware components

Component	Model	Quantity	Total Price
Nema 17 Motor	17HS24-2104S	3	56€
Controller	TMC2209	3	10€
Mainboard	MKS DLC32	1	16€
Endstop	MakerBot (red)	3	5€
Power Supply	24V 5A (optional)	1	15€
Bearing	F695-2RS Fushi	22	15€
Bearing	F623RS Fushi	6	5.5€
GT2 Pulley	Belt:6mm ID:5mm	3	1.5€
GT2 Belt	Belt:6mm Length:252mm	2	3.5€
GT2 Belt	Belt:6mm Length:280mm	1	1.8€
Fan	24V 4010	1	2€
Electromagnet	D20H15mm 3KG 24V	1	3€
Plastic for Printing	PLA/PETG 1Kg	1	22€

process involved generating a MoveIt configuration package based on the robot description. Using the MoveIt setup assistant, various robot parameters such as self-collisions, planning groups, and end-effectors were defined. This allowed the system to plan and execute complex motions based on the robot's kinematic model.

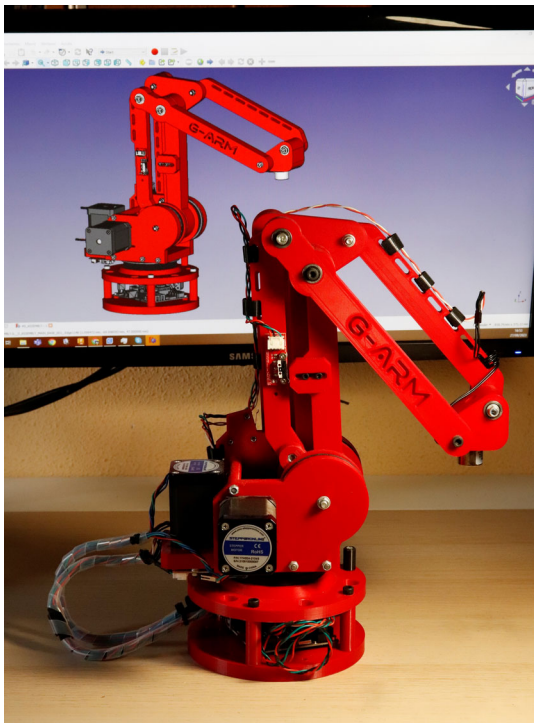


Fig. 17 Actual robot assembly

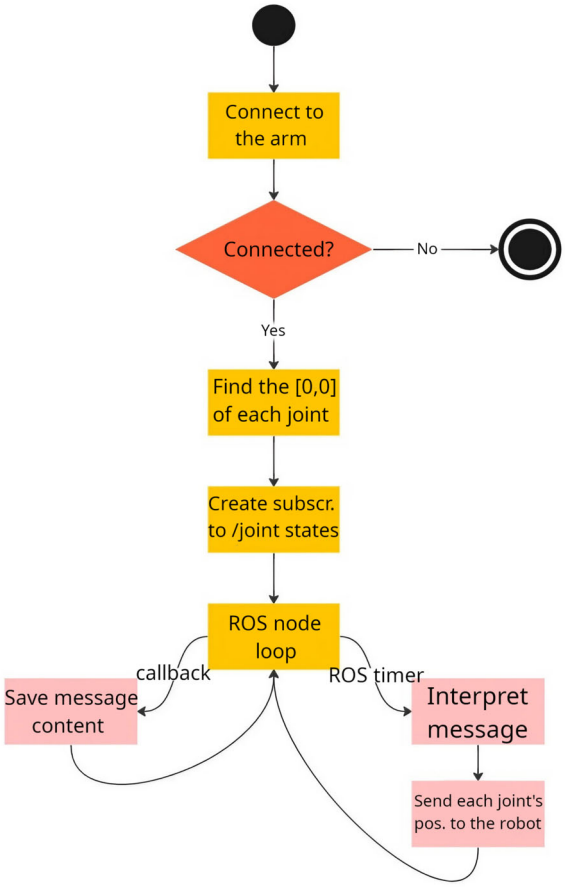


Fig. 18 Activity diagram of the driver

Once the MoveIt configuration was set up, the G-Arm could perform tasks such as automatic trajectory generation and collision avoidance. The integration also enables real-time control and simulation, which are essential for both development and practical deployment.

Driver for ROS 2 The final integration step involved creating a ROS 2 driver node that could execute motion commands on the G-Arm hardware. This node communicated with the robot's controllers and provided feedback on the arm's state. The driver utilized the `ros2_control` framework, which enabled precise joint control through trajectory interpolation. Additionally,

(a) Scenario 1 (b) Scenario 2 (c) Scenario 3

Fig. 19 Images from the load test video

Table 10 Results of the different load test scenarios

Scenario	Maximum Load
Scenario 1	365 g
Scenario 2	480 g
Scenario 3	305 g

joint state information was broadcast in real-time, allowing for accurate monitoring of the arm's movements.

The ROS 2 driver, combined with MoveIt 2, ensures smooth coordination between motion planning, trajectory execution, and real-time control, providing the G-Arm with the capabilities required for advanced robotic tasks.

The activity diagram of the driver is shown in Fig. 18, which outlines the process of receiving trajectory commands, moving joints, and publishing state updates. Although the provided demonstration launcher initiates these nodes, creating a custom launcher would be beneficial for more refined control and flexibility in the system.

6 Experiments

The robot arm's performance was thoroughly assessed through a series of tests under various conditions, including load capacity, maximum speed, and power consumption. The tests, documented with corresponding videos, are linked in the following footnotes: load capacity [11], maximum speed [12], and power consumption [13]. The results are analyzed with respect to the evaluation criteria outlined in the introduction.

6.1 Load capacity

Load capacity was evaluated to determine the robot arm's ability to lift and manipulate weights under various conditions. The tests were conducted in three distinct scenarios, considering both static and dynamic loads.

- **Scenario 1:** The arm lifted weights with its arm fully extended, testing static capacity at the most mechanically disadvantaged position.
- **Scenario 2:** The load was positioned closer to the base, evaluating the effect of reduced lever arm length on lifting performance.
- **Scenario 3:** The electromagnet tool, rated for a 3 kg load, was tested for its ability to lift ferromagnetic objects in static conditions.

In each scenario, the robot attempted to lift and hold the load statically before performing dynamic tests, moving the load through a standard trajectory to assess its effect on trajectory fidelity, precision, and motor performance. The tests identified the limits where excessive weight adversely impacted movement.

The results, summarized in Fig. 19 and Table 10, show that the arm lifted up to 365 g in Scenario 1 (fully extended) and 480 g in Scenario 2 (load near the base). The electromagnet, however, only lifted 305 g, falling short of its rated capacity. Despite this, it successfully manipulated small metallic objects during dynamic movements. While the arm handled

[11] https://youtu.be/McD7kLMKMo0

[12] https://youtu.be/37KK_hJk_tI

[13] https://youtu.be/2MfBabEWZNo

(a) Start of movement (b) End of movement

Fig. 20 Images from the speed test video

loads up to 150 g (e.g., a 100 g TV remote) at high speeds, heavier loads negatively impacted trajectory precision and introduced minor deviations in end-effector accuracy.

6.2 Maximum speed

Maximum speed was tested by evaluating the rotational velocity of each joint while carrying a 100 g load, simulating a typical operational scenario. The robot was mounted on a table to mitigate any base movement induced by inertia, and movements were recorded to measure average speed over defined angular ranges.

The base joint achieved a peak speed of 225°/s, with an average operational speed of 90°/s, considering acceleration and deceleration phases. Secondary joints, responsible for smaller angular movements, reached 120°/s. These speeds were sufficient for smooth and precise movements during manipulation tasks.

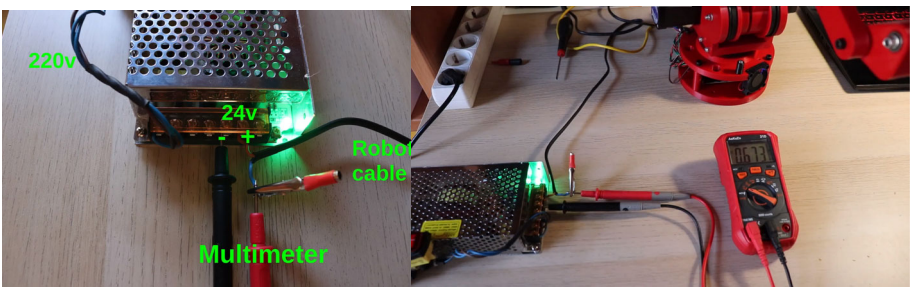
For comparison, the ABB IRB120 robot, priced around €20,000, achieves a base speed of 250°/s, which is slightly higher but comes at a significantly higher cost.

6.3 Power consumption

Power consumption was assessed under different operational scenarios to evaluate the robot's energy efficiency, particularly in the context of battery-powered systems like the Turtlebot 2 platform (Figs. 20 and 21).

Three scenarios were tested:

1. Slow movements: Simulating low-speed, deliberate operations.



(a) Multimeter in series (b) Test scenario

Fig. 21 Images from the power consumption test video

Table 11 Power consumption across various scenarios

Scenario	Average Consumption
Slow Movements	14.4W
Fast Movements	17W
Fixed Position	6.8W

2. Fast movements: Simulating rapid actions.
3. Fixed position: Representing standby power consumption to maintain motor positions.

The results, presented in Table 11, show that the power consumption for slow and fast movements was nearly identical, averaging 14.4W and 17W, respectively. The stationary consumption was significantly lower at 6.8W. These findings indicate that the robot is energy-efficient, making it suitable for mobile systems with power constraints.

7 Conclusions

This project successfully developed a cost-effective robotic arm for educational purposes, achieving key milestones in design, functionality, and deployment. Deployed in two courses at Rey Juan Carlos University-*Industrial Robotics* and *Software Architectures for Robots*-the arm provided students with hands-on experience in robotic manipulation, trajectory planning, and ROS 2 integration. The lightweight, 3D-printed arm demonstrated versatility, low power consumption (under 20W), and a manufacturing cost of €171, well within the €200 budget. Its integration with the ROS 2 ecosystem and MoveIt 2 framework enabled effective teaching and experimentation.

Feedback from both students and educators highlighted the arm's ease of use, its practical value in bridging theoretical concepts with hands-on learning, and its suitability for integration into course structures. Educators valued its role in enhancing student engagement and understanding of robotics. The feedback also pointed to areas for improvement, such as increasing robustness for long-term use and adding more advanced features.

Future work includes adding a fourth degree of freedom to the end effector, developing specialized tools and attachments, incorporating a simulation environment for testing, and creating custom firmware for improved low-level control. These improvements will further enhance the arm's capabilities and its value as an educational tool.

The success of this project demonstrates the potential of low-cost, hands-on robotics systems to transform robotics education. By providing an accessible and scalable solution, the robotic arm can be expanded for use in additional courses and fields, advancing the development of modern robotics curricula.

Acronyms

DOF *Degrees Of Freedom*
STEM *Science, Technology, Engineering and Mathematics*
DIY *Do It by Yourself*
CAD *Computer-Aided Design*
FDM *Fused Deposition Modeling*
ROS *Robot Operating System*
CNC *Control Numérico por Computadora*
PWM *Pulse Width Modulation*
DDS *Data Distribution Service*

SCARA	*Selective Compilant Assembly Robot Arm*
URDF	*Unified Robot Description Format*
XML	*eXtensible Markup Language*
UART	*Universal Asynchronous Receiver/Transmitter*
STL	*Standard Tessellation Language*

Author Contributions Conceptualization, J.V.; methodology, J.V.; software, V.P.; validation, J.V. and V.P.; formal analysis, J.V.; investigation, J.V.; resources, J.V.; data curation, J.V. and V.P.; writing—original draft preparation, J.V. and V.P.; writing—review and editing, J.V.; visualization, V.P.; supervision, J.V.; project administration, J.V.; funding acquisition, J.V. All authors have read and agreed to the published version of the manuscript.

Funding The authors declare that no financial support was received for this work.

Data Availability The datasets generated and/or analyzed during the current study are publicly available and can be accessed via the following link: https://github.com/RoboticsURJC/tfg-vperez/tree/280861172bce3b1c0cfbb155a434364ea68eeb30/src. No restrictions apply to the access or use of these data.

Declarations

Ethical Not applicable.

Competing interests The authors declare that there are no competing interests.

References

1. Adediran EM, Fadare DA, Falana A, Kazeem RA, Ikumapayi OM, Adedayo AS, Adetunla AO, Ifebunandu UJ, Fadare DA, Olarinde ES (2023) Uiarm i: Development of a low-cost and modular 4-dof robotic arm for sorting plastic bottles from waste stream. J Eur des Syst Automatises 56(1):97
2. Almeida P et al (2021) Robotics in stem education: A systematic review of teaching methods and applications. IEEE Access 9:133604–133623
3. Anderson J et al (2022) Affordable robots for education: Reducing costs in stem classes. Int J Robot Autom 20(3):102–115
4. Authors V (2023) A review of current techniques for robotic arm manipulation and mobile navigation. In IEEE Conference Publication
5. Bano S, Atif K, Mehdi S (2024) Systematic review: Potential effectiveness of educational robotics for 21st century skills development in young learners. Educ Inf Technol 29:11135–11153
6. Chavdarov I, Yovchev K, Naydenov B, Hrosinkov V (2024) 3d printed delta robot for educational purposes. In 2024 international conference on software, telecommunications and computer networks (SoftCOM), pp 1–6
7. Eaton M, Tanveer MH (2024) The development and implementation of a cost-effective educational robotic arm using ros-moveit. In 2024 IEEE integrated STEM education conference (ISEC), pp 1–4
8. Čehovin Zajc L, Rezelj A, Skočaj D (2023) Low-cost open-source robotic platform for education. IEEE Trans Learn Technol 16(1):18–25

9. Grubišić V, Crnokić B (2024) A systematic review of robotics' transformative role in education. In: Volarić T, Crnokić B, Vasić D (eds) Digital transformation in education and artificial intelligence application. Cham, Springer Nature Switzerland, pp 257–272
10. Krimpenis AA, Papapaschos V, Bontarenko E (2020) Hydrax, a 3d printed robotic arm for hybrid manufacturing. part i: Custom design, manufacturing and assembly. Procedia Manuf 51:103–108. 30th international conference on flexible automation and intelligent manufacturing (FAIM2021)
11. Lee R, Usaquen J, Martínez F (2024) Development of low-cost stereoscopic vision systems for educational robotics: A state of the art. Global J Eng Technol Adv 20(01):001–009. Received on 16 May 2024; Revised on 25 June 2024; Accepted on 28 June 2024
12. Martínez A et al (2023) Design and implementation of a robotic arm for a mocap system within extended educational mechatronics framework. Mach 11(3):281
13. Papapaschos V, Bontarenko E, Krimpenis AA (2020) Hydrax, a 3d printed robotic arm for hybrid manufacturing. part ii: Control, calibration and programming. Procedia Manuf 51:109–115. 30th international conference on flexible automation and intelligent manufacturing (FAIM2021)
14. Surynek P (2024) Real robot one (rr1): 3d printed robotic arm for teaching robotics engineering and robot control. In 2024 9th international conference on control and robotics engineering (ICCRE), pp 33–38
15. Tselegkaridis S, Sapounidis T (2021) Aerobot: A low-cost educational robot for stem education. J Educ Robot 9(2):56–68
16. Tselegkaridis S, Sapounidis T (2022) Exploring the features of educational robotics and stem research in primary education: A systematic literature review. Educ Sci 12:305
17. Uslu A et al (2022) Trends and research foci of robotics-based stem education: A systematic review. Int J STEM Educ
18. Weeraratne A, Subasinghage K (2024) Comparison of open-source robotics platforms for undergraduate education. In 2024 international research conference on smart computing and systems engineering (SCSE), vol7. pp 1–5