CEA Comité Español de Automática
XLVII JORNADAS DE AUTOMÁTICA CÓRDOBA 2026
UNIVERSIDADE DA CORUÑA

Jornadas de Automática

Open-source, portable, and low-cost educational kit for engineering control courses

Juan M. Gandarias[a,*], Vicente Arévalo-Espejo[a], Ricardo Vázquez-Martín[a], Carlos J. Pérez-del-Pulgar[a]

[a] *Institute for Mechatronics Engineering and Cyber-Physical Systems (IMECH.UMA), Systems Engineering and Automation Department, University of Málaga, 29071, Málaga, Spain*

To cite this article: Gandarias, J.M., Arévalo-Espejo, V., Vázquez-Martín, R., Pérez-del-Pulgar, C.J. 2026. Open-source, portable, and low-cost educational kit for engineering control courses.
Jornadas de Automática, 47. https://doi.org/10.17979/ja-cea.2026.47.13762

Resumen

La experimentación práctica es esencial en la enseñanza de la ingeniería automática; sin embargo, las plataformas tradicionales suelen ser costosas, poco portátiles o depender de software propietario, lo que limita su accesibilidad y escalabilidad. Este trabajo presenta un kit educativo open-source, portátil y de bajo coste para el aprendizaje práctico en asignaturas de control automático. La plataforma integra un sistema embebido basado en ESP32 con un motor de corriente continua con encoder, junto con una interfaz gráfica de usuario (GUI) en Python para la configuración de experimentos, la visualización en tiempo real y el almacenamiento de datos. El firmware emplea FreeRTOS para garantizar un muestreo determinista y múltiples modos de operación (bucle abierto, posición y velocidad). La GUI permite configurar controladores, aplicar referencias y exportar datos. Gracias a su bajo coste y a su naturaleza abierta, los estudiantes pueden montar el sistema y disponer de un equipo de prácticas propio, manteniendo la interacción con un sistema físico real incluso fuera del laboratorio.

Palabras clave: Educación en Control, Diseño de Control, Modelado, Identificación y Procesamiento de Señales, Sistemas de Control Lineal, Computación aplicada al Control.

Open-source, portable, and low-cost educational kit for engineering control courses

Abstract

Hands-on experimentation is essential in control engineering education; however, traditional laboratory platforms are often costly, non-portable, or rely on proprietary software, limiting their accessibility and scalability. This paper presents an open-source, portable, and low-cost educational kit for practical learning in control engineering courses. The kit integrates an ESP32-based embedded system with a direct-current (DC) motor equipped with an encoder, together with a Python graphical user interface (GUI) for experiment configuration, real-time visualization, and data logging. The firmware employs FreeRTOS to ensure deterministic sampling and supports multiple operating modes (open-loop, position, and velocity). The GUI enables users to configure controllers, apply reference signals, and export experimental data. Thanks to its low cost and open nature, students can assemble the system themselves and have a personal experimental kit, maintaining interaction with a real physical system even outside the laboratory.

Keywords: Control Education, Control Design, Modelling, Identification, and Signal Processing, Linear Control Systems, Computers for Control.

1. Introduction

Hands-on experimentation is a cornerstone of control engineering education, as it allows students to directly relate theoretical concepts to the behavior of real physical systems Carlson and Sullivan (1999). While simulation environments are essential for introducing control theory and fostering conceptual understanding, they are often insufficient on their own to

*Autor para correspondencia: jmgandarias@uma.es

convey practical issues such as sensor noise, actuator saturation, discretization effects, and implementation constraints. In addition, simulation-based tools are often perceived by students as hands-off activities, which may limit engagement and learning outcomes Sianez et al. (2010). For this reason, physical laboratory platforms continue to play a crucial role in undergraduate and graduate courses in control, automation, and mechatronics Jara et al. (2011).

Despite their pedagogical relevance, traditional laboratory setups present several practical limitations. Commercial educational platforms are typically expensive, bulky, and difficult to scale when large numbers of students are involved Bernstein (2005); Vazquez-Hurtado et al. (2025). In addition, many laboratory systems rely on proprietary software tools, which increase costs, restrict portability, and limit reuse beyond institutional facilities Dixon et al. (2002). Conversely, low-cost solutions based on microcontrollers often require students to engage in low-level programming, real-time scheduling, and debugging tasks that may introduce accidental complexity and divert attention from the main learning objectives related to system dynamics and control behavior Schmidt et al. (2006); Núñez (2024).

To address these challenges, numerous low-cost, open-source educational platforms have been proposed, particularly for robotics and control education. Devices such as haptic paddles Gandarias et al. (2016); Martinez et al. (2019), mobile robots Muñoz-Ramírez and Gómez-de Gabriel (2016); Gonzalez et al. (2017), and single-degree-of-freedom actuation systems Docekal and Golembiovsky (2018); Muñoz et al. (2022) have demonstrated their effectiveness for teaching dynamics and control concepts in project-based learning environments. Model-based approaches using tools such as Simulink have also been shown to reduce implementation effort and improve teaching efficiency Gandarias et al. (2017). Conversely, programming-centric approaches based on Arduino or similar platforms offer low cost and openness but impose a substantial cognitive and technical burden on students during laboratory sessions Muñoz-Ramírez and Gómez-de Gabriel (2016).

This paper presents an open-source, portable, and low-cost educational kit specifically designed for control engineering courses to deal with these limitations (see Fig. 1). All the code and detailed instructions are available in a public GitHub repository [1]. The proposed kit relies on widely available, low-cost, and open or off-the-shelf hardware components and includes detailed assembly instructions to ensure straightforward system integration and maintainability, preventing this from becoming a barrier for students. A fully developed, open-source firmware is provided that handles low-level tasks such as signal acquisition, real-time scheduling, and control execution. As a result, students are not required to engage in embedded programming, since the firmware can be easily uploaded following the provided guidelines, typically as a one-time setup process. In addition, a cross-platform, open-source Python Graphical User Interface (GUI) enables intuitive experiment configuration, real-time data visualization, controller selection and tuning, and data logging without reliance on proprietary software. This architecture allows students, even in the early stages of their degree, to build and use their own portable experimental kit, which can be reused across multiple courses, enabling them to focus on control and system dynamics concepts rather than implementation details. Hence, the kit we propose preserves interaction with a real system while reducing accidental complexity and improving accessibility, scalability, ease of repair, and cross-disciplinary applicability in educational contexts.

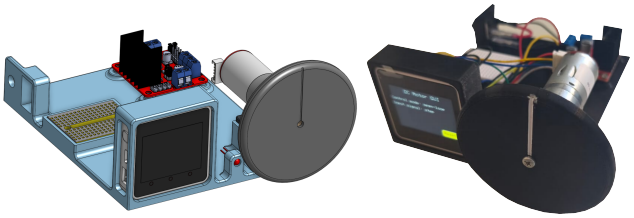

Figure 1: The proposed kit combines a DC motor-based plant, driven by an ESP32-based integrated controller, with a fully open-source, cross-platform graphical user interface (GUI), and requires only a single 3D-printed part for assembly. CAD design (left) and real experimental kit (right).

The remainder of this paper is organized as follows. Section 2 describes the proposed system architecture, including hardware and firmware design. Section 3 introduces the GUI and the communication protocol. Section 4 presents representative educational use cases and example experiments. Finally, Section 5 concludes the paper and outlines directions for future work.

2. System Overview

The proposed educational kit is a compact, portable, and self-contained unit for hands-on experimentation in control engineering courses. Its design follows a modular architecture that integrates a physical control plant, an embedded real-time controller, and a host-level graphical user interface. The complete hardware interaction and communication pipeline is presented in Fig. 2. This section provides an overview of the hardware and firmware components (left-hand side of the pipeline).

2.1. Hardware kit

The physical plant consists of a DC motor equipped with an incremental encoder, driven by an ESP32-based embedded system. Motor actuation is performed via bidirectional Pulse-Width Modulation (PWM) with an L298N H-bridge driver, enabling control of rotation direction and the input voltage applied to the motor. Encoder feedback provides high-resolution angular position measurements, from which velocity can be computed numerically by discrete differentiation.

An external power supply powers the actuation to ensure sufficient current delivery and stable operation. In the setup proposed in this work, a 12 V, 5 A power supply is used, enabling the system to operate reliably under varying loading conditions while remaining affordable and easy to implement. The remaining components are powered via a standard USB connection that is also used for communication.

[1] GitHub repository: `https://github.com/jmgandarias/dc_motor_kit`

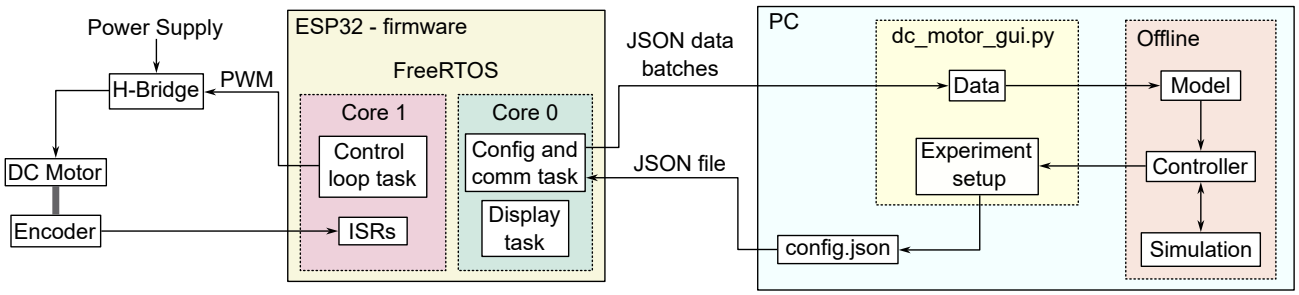

Figure 2: System pipeline showing the interaction between the embedded control architecture and the host graphical interface. The diagram illustrates the flow from experiment configuration and controller execution to real-time data acquisition and visualization.

The proposed kit is available in two configurations. The basic version is built around a standalone ESP32 microcontroller, offering a minimal, cost-effective setup for control experiments. The advanced version is based on the M5Core2 (see Fig. 1), which integrates the ESP32 along with additional components that may be useful for other related subjects but are not essential for the purpose described in this article. Furthermore, the kit allows the inclusion of additional elements such as an LED and a small breadboard, enabling simple extensions for related courses such as industrial electronics or informatics.

The Bill of Materials (BOM), including the list of components and their approximated costs at the time of writing, is provided in Table 1. The total cost of the kit ranges from 26.9 € for the simplest, cheapest version (ESP32) to 77.03 € for the most expensive, complete one (M5Core2).

Table 1: Bill of Materials (BOM). The components at the bottom are optional.

Components	Cost (€)
JGA25-370 DC motor	6.19
L298N H-bridge	1.76
ESP32-DevKitC-32 / M5Core2	3.69 / 51.39
Dupont AWG Cables	1.59
12V, 5A power supply	11.89
3D printed parts	1.78
Jack connector	0.32
Switch	1.03
Mini breadboard	1.04
LED	0.04

2.2. Firmware architecture

The embedded firmware is developed using the Arduino framework for the ESP32 and is structured around a multitasking, multicore architecture based on FreeRTOS Barry (2018) (see the ESP32-firmware block in Fig. 2). This design allows a clear separation of functionalities while maintaining deterministic timing behavior in the control loop.

The firmware is organized into three main tasks. The control task executes the real-time control algorithm in Core 1 with a configurable sampling period. It performs signal acquisition and actuator command generation, and supports multiple operating modes, including open-loop actuation, position control, and velocity control. Encoder signals are processed using Interrupt Service Routines (ISRs) in Core 1 to ensure accurate pulse counting. Angular position and velocity are computed from the encoder data, with optional filtering applied to reduce measurement noise.

A configuration and communication task runs on Core 0 and handles serial communication with the host computer, processes experiment commands, and updates control parameters at runtime. A third task, which also runs on Core 0, manages local interaction and visual feedback (available only in the M5Core2 version), facilitating standalone operation and system status monitoring. Output saturation and experiment time limits are implemented to ensure safe operation during student use.

The control loop runs independently on the embedded system with a configurable sampling period of up to 1 kHz, ensuring deterministic real-time performance. In contrast, communication over the serial interface occurs at a fixed rate of approximately 100 Hz, which is sufficient for visualization. Each transmitted batch, encoded as a JSON message, contains a sequence of time-stamped samples, allowing the host application to reconstruct the high-frequency system behavior despite the lower communication rate.

On the host side, the GUI parses these JSON messages and reconstructs the experiment's temporal evolution for real-time visualization and logging. This design decouples control execution from the communication layer, enabling high-frequency control loops to operate reliably while maintaining an efficient, robust data transmission pipeline.

3. Graphical User Interface and Communication Protocol

To support experiment configuration, execution, and analysis, a cross-platform GUI has been developed in Python (see Fig. 3), with its dependencies specified in the *requirements* file in the GitHub repository. The GUI is a key element of the proposed educational kit, providing students with intuitive access to the experimental kit while abstracting away low-level programming, communication, and implementation details. This section describes the interface's design objectives, the communication protocol, and the main interaction and data management features.

3.1. Design objectives

The GUI has been designed with a strong educational focus. Its primary objective is to enable students to interact with a real control system without requiring prior knowledge of embedded programming or communication protocols. The interface should therefore be intuitive, lightweight, and easy to deploy on standard personal computers.

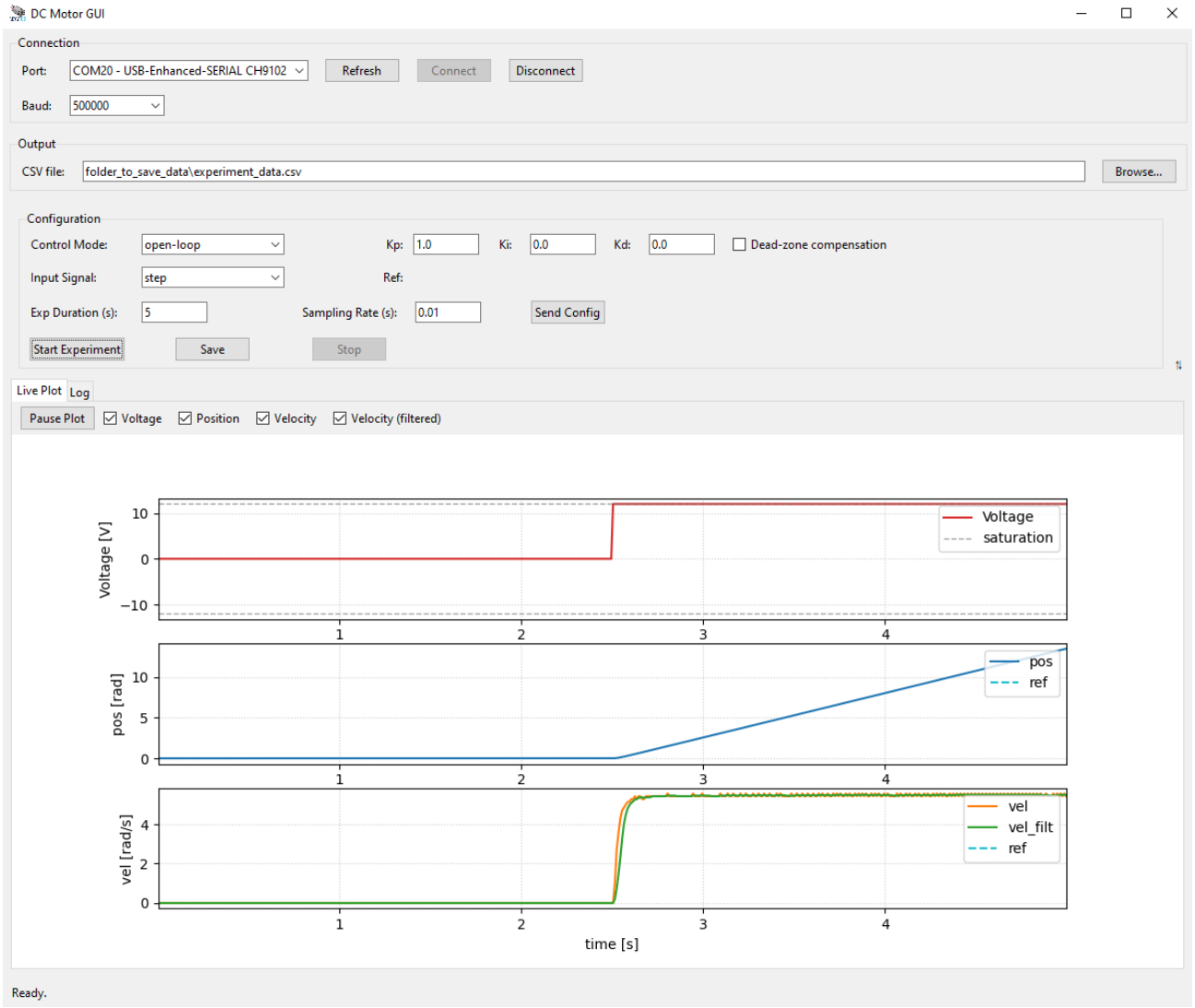

Figure 3: GUI of the proposed system. The upper section shows the serial communication panel, including port selection, baud rate configuration, and connection controls. Below it, the experiment configuration panel allows selection of the control mode, input signal, controller parameters (i.e., PID gains), experiment duration, and sampling time. The lower part of the interface displays real-time plots of the main system variables: the top plot shows the applied voltage with saturation limits, the middle plot shows angular position, and the bottom plot shows measured and filtered velocity. Logging options and variable selection checkboxes are also available for data recording and visualization.

In addition, the GUI is intended to provide immediate visual feedback on system behavior, allowing students to relate theoretical concepts—such as transient response, steady-state error, or the effect of controller parameters—to real experimental results. Finally, the interface must support reproducibility and assessment by enabling systematic experiment configuration and automatic data logging.

3.2. *Communication and experiment management*

Communication between the embedded system and the host computer is implemented via a USB serial interface. A lightweight text-based protocol is used, combining explicit command messages with structured configuration data encoded in JSON format. This approach allows experiment parameters—such as the sampling period, controller gains, reference signals, and experiment duration—to be dynamically modified without recompiling or reprogramming the firmware.

During operation, the firmware continuously transmits time-stamped variables (actuation, position, velocity, filtered velocity, and reference) to the host computer. This data stream enables real-time monitoring of system behavior and supports post-processing and analysis of experimental results.

To ensure efficient and reliable data transfer at different sampling rates, the firmware groups and serializes the measurements into compact JSON structures, which are then transmitted to the host periodically. This batching strategy reduces communication overhead, improves throughput, and prevents data loss when operating at high control frequencies.

3.3. *Data logging and offline analysis*

In addition to real-time visualization, the GUI supports automatic data logging throughout each experiment. Received data samples are stored locally and can be exported to standard file formats, such as CSV, at the end of an experiment or on demand. This feature enables students to perform offline analysis using external tools and to document their results in

laboratory reports or assignments. In addition, this feature allows students to perform an offline workflow using the data to obtain experimental models, design controllers, and work on simulations, as described below.

3.4. Experimental workflow for control design

The right-hand side of the diagram in Fig. 2 represents the host-level processing pipeline, which connects experiment configuration, real-time monitoring, and optional analysis tools. The proposed pipeline enables a complete experimental workflow that closely follows the typical control engineering methodology. Using the graphical interface, the student can first configure and launch an experiment on the real system, obtaining time-stamped measurements. These experimental data can then be exported and used to develop a mathematical model of the system, for instance, through system identification techniques.

Based on this model, the student can design a controller using standard tools, either through dedicated environments such as MATLAB or open-source alternatives such as the Python Control Systems Library Fuller et al. (2021). The designed controller can be validated in simulation before being implemented in the real system. Once validated, a new experiment can be executed by integrating the controller into the embedded firmware, allowing direct comparison between simulated and real responses.

This iterative process allows students to bridge the gap between theory and practice, verify that theoretical concepts are supported by experimental results, and reinforce their understanding through direct interaction with the physical system.

4. Educational Activities and Use Cases

The proposed educational kit has been designed to support a broad spectrum of learning activities in control engineering, ranging from introductory laboratory sessions to advanced project-based coursework. Its modularity and software flexibility allow the same kit to be reused across different academic levels and teaching methodologies, promoting continuity and scalability in control education.

4.1. Undergraduate-level activities

In undergraduate control courses, the kit can be primarily used to introduce fundamental concepts such as system dynamics, transient response, stability, and feedback control. Initial laboratory sessions typically focus on system familiarization through open-loop experiments, where students apply step or ramp inputs to the motor and observe the resulting position and velocity responses. These experiments also allow learners to identify practical phenomena such as actuator saturation, nonlinear effects, and measurement noise.

Subsequent sessions introduce closed-loop control. By implementing proportional or PID controllers and adjusting their gains through the GUI, students can directly observe how controller parameters affect key performance indicators such as overshoot, settling time, and steady-state error. The immediate visual feedback provided by real-time plots facilitates conceptual understanding and promotes active experimentation.

At the intermediate stage, students can also explore the differences between continuous-time and discrete-time control systems. By modifying the sampling period, they can analyze its impact on the dynamic behavior, including changes in stability, responsiveness, and noise sensitivity. Additionally, the platform allows students to observe practical nonlinear effects such as actuator saturation, as well as mitigation strategies including anti-windup mechanisms. The effect of filtering the derivative action can also be studied, highlighting the trade-off between noise attenuation and control performance.

In higher undergraduate courses, the kit can be extended to develop more advanced control strategies, such as cascade control or feedforward compensation. These approaches may require modifications to the embedded firmware and the GUI, providing students with additional insight into the implementation aspects of control systems.

Furthermore, the kit can also be used in closely related courses within the control engineering curriculum, such as industrial informatics or microcontroller programming. In these contexts, students can explore topics such as embedded programming, PWM signal generation, ISRs, digital controller implementation, and real-time system design, thereby reinforcing their understanding of both theoretical and practical aspects of engineering systems.

4.2. Usage scenarios in educational environments

The proposed kit supports several complementary use cases, making it a flexible tool across diverse teaching contexts and learning methodologies.

First, the kit can be used in a conventional manner as a laboratory setup for hands-on sessions in control engineering courses. In this context, students perform guided experiments to study system dynamics, controller design, and the real-time behavior of physical systems.

Second, thanks to its portability, the kit can be used by instructors during theoretical lectures to illustrate key concepts directly and interactively. For example, it can be used to demonstrate step responses, transient dynamics, or the effects of controller parameters such as PID gains, providing an immediate link between theoretical analysis and real-system behavior in the classroom.

Third, the low cost and accessibility of the components enable students to assemble and use the kit at home. This supports the development of assignments and practical activities beyond the laboratory environment, and facilitates their integration into Massive Open Online Courses (MOOCs). In this context, the kit can complement traditionally theory-oriented courses with hands-on experimentation, increasing student engagement and enhancing learning outcomes.

Finally, the open-source nature of both hardware and software enables project-based and self-directed learning. Students can independently explore the system, develop their own experiments, extend existing functionalities, or implement new features without direct supervision. This fosters autonomous learning and creativity, and allows the kit to evolve beyond its initial design, reaching wider user communities and expanding its educational potential.

5. Conclusions and Future Work

This paper has presented an open-source, portable, and low-cost educational kit designed to support hands-on experimentation in engineering control courses. The proposed solution integrates a physical DC motor plant controlled by an ESP32-based embedded system with a Python GUI that enables intuitive experiment configuration, real-time visualization, and automated data logging. This architecture preserves direct interaction with a real system while minimizing accidental complexity and avoiding dependence on proprietary software environments.

The kit has been conceived with a strong educational focus, enabling reuse across different academic levels and teaching methodologies. Its flexibility makes it suitable for guided laboratory sessions, exploratory activities, and project-based learning approaches, helping students connect theoretical concepts with experimental behavior and gain experience with real-world implementation issues.

At its current stage, the kit is primarily focused on classical control theory, based on linear time-invariant (LTI) systems and controller design using transfer functions, such as PID control. However, it can be extended towards modern control approaches in state-space form, allowing the study of concepts such as pole placement, state feedback, and integrator design. This evolution is planned as part of future developments.

At more advanced academic levels, the kit can be extended to support the implementation and evaluation of more sophisticated control strategies. This requires modifications at both the firmware and graphical interface levels to incorporate functionalities beyond the standard position and velocity control loops. In particular, extensions may include cascade control architectures, feedforward compensation strategies, or nonlinear PID controllers, enabling students to explore more advanced control schemes under realistic conditions. Furthermore, optimal control techniques such as Linear Quadratic Regulator (LQR) or Model Predictive Control (MPC) can also be addressed. Due to their higher computational requirements, these approaches may require dedicated firmware, potentially leveraging lightweight embedded optimization libraries such as *tinyMPC* Nguyen et al. (2024).

Future work will therefore focus on extending both the technical and educational capabilities of the kit. On the technical side, this includes integrating advanced control strategies, improving signal processing methods, and supporting computationally demanding algorithms under embedded constraints. On the educational side, future efforts will focus on incorporating the kit into remote and hybrid laboratory environments, as well as systematically assessing learning outcomes in real classroom settings.

Overall, the proposed kit provides a practical, scalable, and accessible tool for control engineering education, promoting the adoption of open, flexible, and reusable experimental kits that effectively bridge the gap between theory and real-world implementation.

Acknowledgements

This research was funded by the University of Málaga.

References

Barry, R., 2018. FreeRTOS Reference Manual: API functions and configuration options. Real Time Engineers Ltd., accedido: 28 de mayo de 2026.
URL: https://freertos.org

Bernstein, D. S., 2005. The quanser dc motor control trainer individual or team learning for hands-on control education-product review. IEEE Control Systems Magazine 25 (3), 90–93.
DOI: https://doi.org/10.1109/MCS.2005.1432604

Carlson, L. E., Sullivan, J. F., 1999. Hands-on engineering: learning by doing in the integrated teaching and learning program. International Journal of Engineering Education 15 (1), 20–31.
URL: https://www.ijee.ie/articles/Vol15-1/Ijee1041.pdf

Dixon, W. E., Dawson, D. M., Costic, B. T., De Queiroz, M. S., 2002. A matlab-based control systems laboratory experience for undergraduate students: toward standardization and shared resources. IEEE Transactions on Education 45 (3), 218–226.
DOI: https://doi.org/10.1109/TE.2002.1024613

Docekal, T., Golembiovsky, M., 2018. Low cost laboratory plant for control system education. IFAC-PapersOnLine 51 (6), 289–294, 15th IFAC Conference on Programmable Devices and Embedded Systems PDeS 2018.
DOI: https://doi.org/10.1016/j.ifacol.2018.07.168

Fuller, S., Greiner, B., Moore, J., Murray, R., van Paassen, R., Yorke, R., 2021. The python control systems library (python-control). In: 60th IEEE Conference on Decision and Control (CDC). IEEE, pp. 4875–4881.
DOI: https://doi.org/10.1109/CDC45484.2021.9683368

Gandarias, J. M., Kalhor, S. A., Gómez-de Gabriel, J. M., 2016. Diseño y uso de una paleta háptica para prácticas de teleoperación con simulink. Actas de las XXXVII Jornadas de Automática.
DOI: https://doi.org/10.17979/spudc.9788497498081.0286

Gandarias, J. M., Muñoz-Ramírez, A. J., Gómez-de Gabriel, J. M., et al., 2017. Uso del haptic paddle con aprendizaje basado en proyectos. Actas de las XXXVIII Jornadas de Automática.
URL: http://hdl.handle.net/2183/25764

Gonzalez, C., Alvarado, I., La Peña, D. M., 2017. Low cost two-wheels self-balancing robot for control education. IFAC-PapersOnLine 50 (1), 9174–9179.
DOI: https://doi.org/10.1016/j.ifacol.2017.08.1729

Jara, C. A., Candelas, F. A., Puente, S. T., Torres, F., 2011. Hands-on experiences of undergraduate students in automatics and robotics using a virtual and remote laboratory. Computers & Education 57 (4), 2451–2461.

Martinez, M. O., Nunez, C. M., Liao, T., Morimoto, T. K., Okamura, A. M., 2019. Evolution and analysis of hapkit: An open-source haptic device for educational applications. IEEE transactions on haptics 13 (2), 354–367.
DOI: https://doi.org/10.1109/TOH.2019.2948609

Muñoz, J., Monje, C. A., Mena, L., Balaguer, C., 2022. Educational low cost platform for control engineering. Actas de las XLIII Jornadas de Automática.
DOI: https://doi.org/10.17979/spudc.9788497498418.0295

Muñoz-Ramírez, A. J., Gómez-de Gabriel, J. M., 2016. Modelar o programar en prácticas de robótica. Actas de las XXXVII Jornadas de Automática.
DOI: 10.17979/spudc.9788497498081.0432

Nguyen, K., Schoedel, S., Alavilli, A., Plancher, B., Manchester, Z., 2024. Tinympc: Model-predictive control on resource-constrained microcontrollers. In: IEEE International Conference on Robotics and Automation (ICRA).
DOI: https://doi.org/10.1109/ICRA57147.2024.10610987

Núñez, E. B., 2024. Diseño de un plóter para el aprendizaje del modelado y control de motores cc. Actas de las XLV Jornadas de Automática (45).
DOI: https://doi.org/10.17979/ja-cea.2024.45.10900

Schmidt, D. C., et al., 2006. Model-driven engineering. Computer-IEEE Computer Society- 39 (2), 25.
URL: http://unbox.org/wisp/var/06miami/mmbse/01597083.pdf

Sianez, D. M., Fugère, M. A., Lennon, C. A., 2010. Technology and engineering education students' perceptions of hands-on and hands-off activities. Research in Science & Technological Education 28 (3), 291–299.
DOI: https://doi.org/10.1080/02635143.2010.502102

Vazquez-Hurtado, C., Sotelo, D., Sotelo, C., Frye, M., Martínez-Jiménez, F., Oropeza, R. S., Quiñones, J. I. G., 2025. Development of control engineering curriculum for advanced research and undergraduate education: A practical approach to bring theory closer to practice using quanser® products. In: 2025 ASEE Annual Conference & Exposition.
DOI: https://doi.org/10.18260/1-2--56284