Teaching Service Robotics with ROS and Unibotics web framework in Higher Education

Lucía Chen *, José M. Cañas †, David Roldán § and Diego Martín ¶
Universidad Rey Juan Carlos, Spain
Lía García-Pérez ‡
Universidad Complutense de Madrid, Spain
Florian Stöckl ‖, Silvan Müller ** and Marcus Strand ††
Baden-Wuerttemberg Cooperative State University, Germany

***Abstract*—Unibotics is an open web framework for robot programming and teaching robotics in Higher Education. It is based on ROS and supports both Gazebo simulated robots and real ones. This paper describes the latest improvements on Unibotics regarding Service Robotics. Firstly, a new exercise has been developed from scratch regarding the navigation of a robot inside a warehouse for moving goods. Two logistics robot models have been designed to mimic real Autonomous Mobile Robots (AMRs), and three reference solutions have been developed using the Open Motion Planning Library. They take into account the corresponding robot geometry and Ackermann dynamics constraints. Secondly, Unibotics has been successfully used in two different European universities and the teaching experiences are reported.**

***Index Terms*—Service Robots, Education Robotics, ROS, Logistics, Visual-Based Navigation**

I. Introduction

Service robotics has become a fundamental tool for transforming various industrial and commercial sectors, improving efficiency and quality of life. These robotic systems, designed to interact directly with humans and perform everyday tasks, are revolutionising areas such as healthcare [1], logistics [2] and home maintenance [3]. The implementation of service robots not only optimises operational processes and reduces costs, but also allows for greater customisation and adaptation to the specific needs of users. In a context of rapid technological evolution and increasing demand for innovative solutions, service robotics is positioned as an essential pillar for sustainable development and global competitiveness.

Teaching service robotics to future engineers at university is crucial to prepare a new generation of skilled professionals to meet the technological challenges of the 21st century. Including this discipline in the curriculum provides students not only with advanced technical knowledge, but also with the practical and creative skills needed to design, implement and optimise robotic systems intended to interact with people and assist in everyday tasks. It also promotes a comprehensive understanding of the ethics and societal implications of robotics, ensuring that future engineers are prepared to develop innovative solutions that improve the quality of life and promote sustainable development. As the demand for service robots grows exponentially in various sectors, university training in this area becomes an essential component to ensure competitiveness and technological leadership in the global market.

Service robots are miscellaneous, expensive and educational institutions often face budgetary constraints that make it difficult to acquire multiple units for classroom use. In addition, the complexity of these systems requires constant supervision and a high level of technical knowledge on the part of instructors, which can be a challenge in classrooms with many students.

This reality makes the use of advanced simulators [4], [5] and hybrid approaches [6] vitally important in the teaching of service robotics. Simulators allow students to practice and experiment in a virtual environment, faithfully replicating the characteristics and behaviours of real robots. In this way, testing and development of complex algorithms can be carried out without the risks and costs associated with the direct use of hardware. In addition, simulators facilitate equitable access to hands-on training, ensuring that all students have the opportunity to interact with advanced robotics technologies, regardless of economic constraints.

There are several private web-based online robotics training platforms, all of which use virtual laboratories. The Construct's Robot Ignite Academy is based on ROS, Gazebo and Jupyter and offers several short courses. Riders.ai offers online robotics courses with real-world applications. RobotBenchmark from Cyberbotics provides free access to a range of robotics exercises based on Webots simulations run in the cloud.

Robots Formation Control Platform (RFCP) is a web-based interactive environment for experimentation with mobile robots [7]. RFCP has an interactive multi-robot simulator, an experimental environment for working remotely with real robots, namely the low-cost moway educational robots. Another noteworthy initiative is the Robot Programming Network (RPN) [8], [9]. It extends existing remote robot laboratories with the flexibility and power of writing ROS code in a Web browser and running it on the remote robot on the server side with a single click.

An additional web learning framework is Unibotics [10], which was born offline, then became ROS-based [11] and then online [12]. It currently contains more than 20 exercises, 10 of which relate to different service robotics topics.

Second section of this paper briefly describes the Unibotics

robot programming website. Third section describes the available Service Robotics exercises at it, making emphasis in two of them: Follow Line exercise for autonomous driving and Amazon Warehouse exercise for robots in logistics. Fourth section summarizes two teaching experiences at two different European universities using part of those contents.

II. Unibotics web platform for teaching robotics

Unibotics[1] is an online robot programming website from the open source JdeRobot organization[2]. It is based in ROS middleware (currently ROS2 Humble) and Gazebo simulator (currently Gazebo 11). It aims to improve the practical tools for learning and teaching robotics, making emphasis on the software, on the programming of robotics algorithms (perception, planning, control, etc). It is also suited for distance learning, and a good complement to MOOCs on robotics.

Its main features are:

- Supports Python programming language.
- All from the browser: code edition, execution and debugging (Figure 1).
- Easy to install: all robotics dependencies are preinstalled in a container (a docker image).
- Multiplatform: it works with Linux, Windows and MacOS operating systems.
- Supports both simulated and real robots.

The Unibotics design is described in detail in [12], [13]. At runtime there are two main components: the web browser and a docker container. The browser acts as the graphical user interface (GUI) of the framework, allowing the user to select an exercise in which he/she has to solve a task by programming a robot. Once the user has selected an exercise, the GUI of that exercise is displayed, as shown in figure 1. This webpage allows the user to edit the source code of their robotics application and to view the application running. There are several widgets to display the simulated robotics world, the robot in action, a text console, some auxiliary widgets for debugging and specific widgets depending on each exercise.

Both the robot simulator and the robotics application itself run inside a Docker container named RoboticsBackend, which the user must download from a public repository[3] and launch before using the framework. It is the robotics runtime of the system, and all the robotics dependencies are preinstalled. Therefore, the user does not need to install them on the local computer. which greatly reduces the amount of time needed to start programming robots. Inside RoboticsBackend there is a Robot Application Manager (RAM) that acts as a bridge between the browser and the robotics applications inside. When the user executes code through the browser, this code is sent to the RAM, which runs it within RoboticsBackend. The robotics application sends the results of the execution of this code (images, data, simulation result, etc) back to the RAM and then to the browser in real time.

[1]https://unibotics.org
[2]https://jderobot.github.io/
[3]https://hub.docker.com/r/jderobot/robotics-backend/tags

Unibotics includes the online version of RoboticsAcademy [4], an open source collection of robotics academic challenges. Several exercises and courses have already been presented in [14], [15].

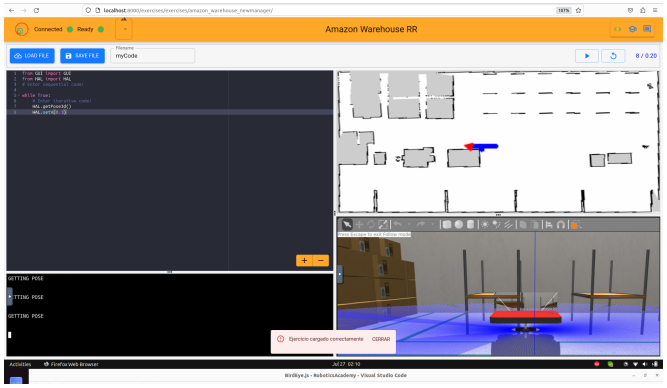

Fig. 1. Webpage for the Amazon Warehouse exercise in Unibotics

III. Service Robotics contents

One of the challenges when teaching Service Robotics is the large variety of robots, scenarios and application domains. Unibotics currently supports several of them [5].

Regarding cleaning robots, two exercises are provided: the basic and the localized vacuum cleaner. In both, the goal is to program the robot to clean an apartment in a given time interval. The larger the area covered, the better. The basic vacuum cleaner[6] mimics the low-end commercial ones, without good localisation. A pseudo random navigation algorithm with spirals and bump-and-go stages is a reference solution, typically implemented with a reactive Finite State Machine. The localized vacuum cleaner mimics the high-end commercial ones, which have good localization. They are typically programmed with a coverage algorithm such as BSA [16], which provides systematic movements and a more efficient cleaning.

One increasing type of service robots are drones [14]. They are successfully used in crop monitoring, search and rescue operations and infrastructure inspection. Regarding drones Unibotics provides several exercises, including Drone Rescue People [7] and Power Tower monitoring [8]. In Drone Rescue People exercise the drone helps to locate survivors in the sea after a plane crash in unknown coordinates, so the ER team may go directly to the survivor positions. The drone explores the accident area and looks for humans using its onboard camera.

Regarding social robots, Follow Person exercise[9] is provided. The students have to program a social robot endowed with a camera in order to follow a moving person inside a

[4]https://jderobot.github.io/RoboticsAcademy/
[5]https://jderobot.github.io/RoboticsAcademy/exercises/
[6]https://youtu.be/-Uwl4TQNVpU
[7]https://youtu.be/O8atiuEamp4
[8]https://youtu.be/IAGZmqNrmio
[9]https://www.youtube.com/watch?v=eSwLHWgYMsY

hospital scenario. Yolo deep learning perception is typically used to detect the person in the image and a local obstacle avoidance algorithm such as VFF is a reference navigation solution.

Unibotics also provides several exercises about Autonomous Driving: obstacle avoidance, global navigation [10], AutoParking and FollowLine. And regarding Logistics, it includes the Amazon Warehouse exercise. We are going to describe with some details these last two exercises, as they were used in the teaching experiences of section IV.

A. Follow Line exercise

In this exercise[11] the robot is a Formula1 car with an onboard camera, steering wheel and throttle. Several Formula1 circuits are provided (MonteCarlo, Nürburgring, MontMeló...), with a red line in the middle of the road to simplify the perception. The challenge for students is to program the navigation software of the car to follow the red line and complete the circuit lap in the shortest time.

Two Python programming API are provided: HAL-API and GUI-API. HAL-API is the Hardware Abstraction Layer, it includes functions such as `HAL.getImage()` to get the image (BGR8), `HAL.setV(velocity)` to set the linear speed and `HAL.setW(velocity)` to set the angular velocity. GUI-API provides the `GUI.showImage(image)`, so the student may show the output of her/his perception and debug it.

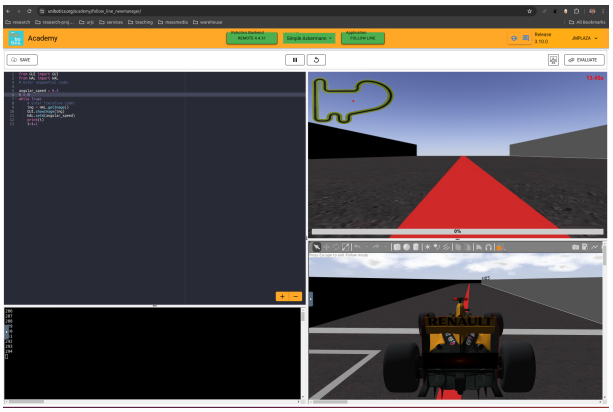

Fig. 2. Webpage for the Follow Line exercise in Unibotics.

A reference solution is a PID visual control algorithm in a reactive infinite loop. It involves a color filter and image processing for identifying the red line in the frame, and a PID control for commanding the steering speed and the linear speed for each iteration.

B. Amazon Warehouse exercise

The challenge in this exercise is to program a logistics robot for moving goods inside a warehouse. The goods are stored in shelves. Two robots (Figure 5) and two warehouses (Figure 3) are provided. In the warehouse there are several shelves (Figure 4), boxes and obstacles. First robot is an holonomic one (Figure 5 (right)) which can turn in the spot and includes a lifting platform. Second robot is a rectangular one, with Ackermann dynamics (Figure 5 (left)), which is inspired on a real AMR at BMW factories.

The Python HAL-API provides `HAL.getPose3d()`, `HAL.setV(velocity)`, `HAL.setW(velocity)` and two functions for the lifting platform: `HAL.lift()` and `HAL.putdown()`. The GUI-API provides `GUI.showPath(array)` to show a path on the map, as a 2D array containing the sequence of waypoints, and `GUI.getMap(url)` to get the warehouse map as a 2D image.

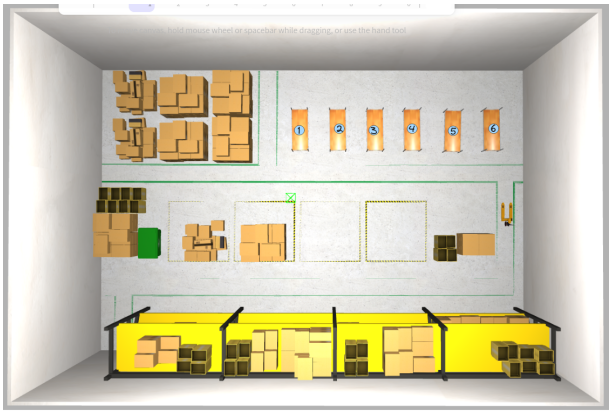

Fig. 3. Warehouse #1 for Amazon logistics exercise.

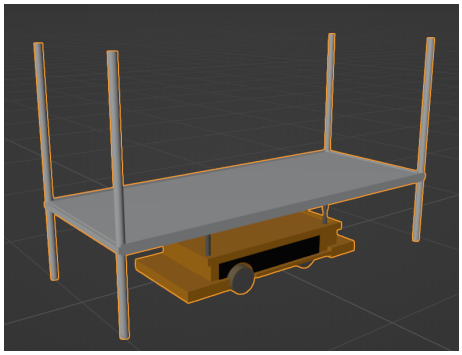

Fig. 4. Shelf in the Amazon Warehouse exercise in Unibotics.

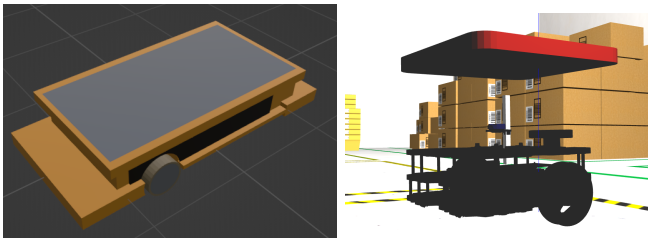

Fig. 5. Provided robots for Amazon Warehouse exercise in Unibotics.

[10] https://jderobot.github.io/RoboticsAcademy/exercises/AutonomousCars/global_navigation

[11] https://jderobot.github.io/RoboticsAcademy/exercises/AutonomousCars/follow_line

The reference solutions for this exercise use the Open Motion Planning Library [12] [17], as it is a requirement and an academic goal. Three solutions have been developed: (a) using pure spatial path planning assuming circular or square geometry of the robot; (b) using spatial path planning with rectangular geometry (Figure 6); and (c) planning with control constraints [18] (Ackermann) and rectangular geometry (Figure 7).

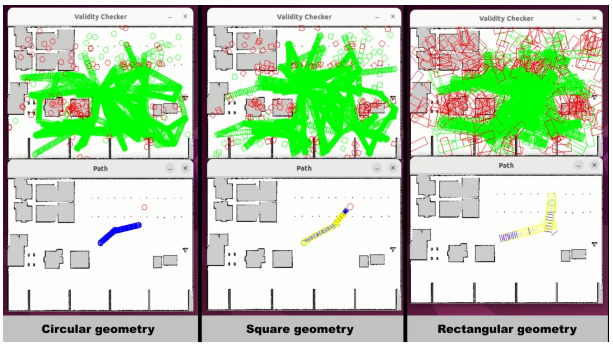

Fig. 6. Spatial path planning with different robot geometries

In (a) and (b) the state space is the XY world, and the state validity checking tests whether or not a pose in the map is free of collision for the robot body with its geometry. Several planning algorithms are available in OMPL (PRM, RRT, Batch Informed Trees, SPARS, etc.) and they find the optimal path from initial position to destination. The geometry of the robot is different in the path towards the shelf and with the shelf onboard, and that has to be taken into account. Beyond the path planning the students have also to program the path execution, a position based controller. A video of typical execution is publicly available [13].

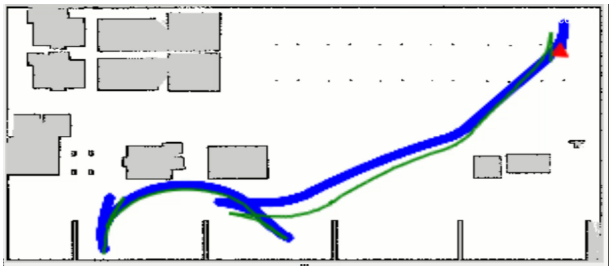

Fig. 7. Planned and achieved path when considering motion constraints.

In (c) the planner delivers a temporal sequence of commands to be sent to the robot motors. Figure 7 shows an illustrative example and a video of a typical execution is also publicly available [14].

IV. Teaching Experiences

The service robotics contents of Unibotics have been recently used in two different European universities in the 2023-2024 course. Those teaching experiences are briefly reported in this section.

A. Baden-Wuerttemberg Cooperative State University (DHBW in Germany)

This institution of higher education follows a dual education system in which students are employed by a company while pursuing their studies. The combination of theoretical instruction and practical experience prepares students effectively for their future careers. At the Karlsruhe campus, undergraduate Computer Science students can select the elective course 'Applications of Robotics' in their second year. This course introduces to Robot Operating System (ROS), various robotic systems, including industrial and collaborative robots, as well as mobile robots and manipulators. The course is offered annually, with 13 students enrolled in 2024. Due to the diverse backgrounds of these students—who are employed in different companies—their prior knowledge of robotics varies significantly. This heterogeneity poses a challenge in designing laboratory exercises that are both accessible and sufficiently challenging for all participants [19]. The course comprises 36 h of lectures and 39 h of self-study. In Karlsruhe, these hours are distributed across six sessions, two of which focused on the Unibotics platform. During these sessions, students worked exclusively on the 'Follow Line' exercise. They were organized into groups of two to three and engaged in a competitive format. For instructors, the Unibotics platform provided a straightforward setup, offering compatibility with nearly all classroom devices, including Windows laptops with adequate hardware specifications. On macOS, Docker Desktop was recommended to mitigate performance issues.

For the students' feedback they were asked to rate the following statements on a scale of 1 (="I strongly disagree") to 5 (="I strongly agree"):

S1 I had prior knowledge for this exercise.
S2 The exercise was at an appropriate level.
S3 I have learnt something new.
S4 I would have liked to get a deeper insight into the topic.

In addition to S1 they were asked what prior knowledge they had.

Figure 8 shows a visual representation of the evaluated students' feedback forms. 62% had no prior knowledge for the exercise. The five students with prior knowledge mentioned topics such as *OpenCV*, Python, PID and line following with *LEGO* robots. For 85% of the students the exercise was at an appropriate level and 77% learned something new. More than the half of them even want to deepen the topics of the "Follow Line" exercise. Some additional comments on the exercise:

- 'The embedded IDE could be better' ⇒ They were missing syntax highlighting and auto-completion
- Simulation errors occurred on relatively low spec hardware ⇒ No current hardware
- Some students prefer real hardware and real robots to simulation ⇒ Connecting Unibotics to real hardware in the lab

[12] https://ompl.kavrakilab.org/
[13] https://youtu.be/EVt9vYqEoDg
[14] https://youtu.be/0PO4S8Omn30

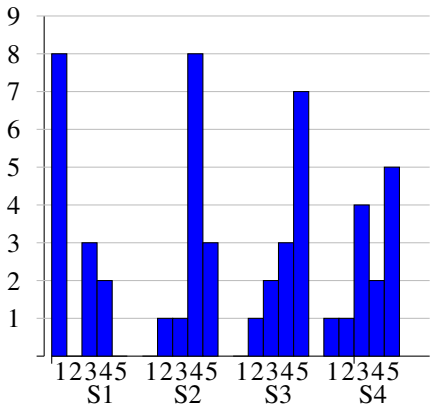

Fig. 8. Visual representation of DHBW students' feedback for "Follow Line" exercise.

There were no complaints about the setup or the instructions.

Overall, the students liked the course and the Unibotics "Follow Line" exercise. They engaged with the platform and came up with really good solutions in a fairly short time. It is ideal for group work, but can also be solved individually.

B. Universidad Rey Juan Carlos

This university offers a Degree on Robotics Software Engineering, which inludes a course on Service Robotics on the fourth year. 25 undergrad students participated in this pilot study from the Service Robotics course, with 4 exercises: "Localized Vacuum Cleaner", "Drone Rescue People", "AutoParking" and "Amazon warehouse". This course lasts 13 weeks, 4 class hours a week (2h for theory classes and 2h for practical classes with Unibotics), and their focus is on the robotics algorithms and the final applications, not in the ROS topics management which are taught in other courses. All the students had ROS prior knowledge, anyway.

Fig. 9. URJC students doing the AutoParking exercise in Unibotics.

Several minor problems arouse along the course, mainly with the new GUI, and they were solved in two working days at most, but the users were frequently required to update their RoboticsBackend images.

The students rated this course using Unibotics with a score of 4.30 out of 5. This ranking is above the mean of the Robotics Software Engineering degree, which is 3.87 out of 5.

V. Conclusions

The last improvements on Unibotics robot programming website regarding Service Robotics have been presented. First, a new exercise about robotics in logistics has been created and integrated. It is based on real Amazon warehouses where a fleet of robots moves goods inside the warehouse in a very fast and efficient way. The corresponding world models and robot models have been developed for Gazebo simulator, including an Ackermann steering robot. Three reference solutions based on Open Motion Planning Library for path planning were programmed.

In addition, two teaching experiences with Unibotics and Service Robotics have also been described. One with Computer Science students in a German dual university. And another one with students of the Degree on Robotics Software Engineering in a Spanish university. Both were successful, and the presented robotics framework provided an easy setup and probed to be truly multiplatform, as it worked fine for students with Linux operating system, MS-Windows or even MacOS.

As future lines, we aim to provide not only the current simple Python interfaces to students (HAL-API and GUI-API) but also to provide the raw interfaces, the robot ROS-topics. This way Unibotics could also be used not only for teaching robotics algorithms, but also for teaching the ROS middleware itself, including programming the robotics application directly with ROS topics and services. In addition, it may help to expand the scope of the platform, which could be used both by Computer Science students with no prior ROS background and by Robotics Engineering students with some experience on ROS middleware.

Another future line is to migrate from Gazebo 11 to the more recent LTS release Gazebo Harmonic. It is more efficient and it will allow a fluent execution even in computers with limited resources.

Acknowledgments

This research was partially funded by the UNIBOTICS-GAM project Ref. TED2021-132632B-I00, from Proyectos de Transición Ecológica y Transición Digital 2021 call Spain. Authors also appreciate the help of Google for improving RoboticsAcademy through the Google Summer of Code program since 2017.

References

[1] J. Holland, L. Kingston, C. McCarthy, E. Armstrong, P. O'Dwyer, F. Merz, and M. McConnell, "Service robots in the healthcare sector," *Robotics*, vol. 10, no. 1, p. 47, 2021.

[2] J. A. Gonzalez-Aguirre, R. Osorio-Oliveros, K. L. Rodriguez-Hernandez, J. Lizárraga-Iturralde, R. Morales Menendez, R. A. Ramirez-Mendoza, M. A. Ramirez-Moreno, and J. d. J. Lozoya-Santos, "Service robots: Trends and technology," *Applied Sciences*, vol. 11, no. 22, p. 10702, 2021.

[3] G. A. Zachiotis, G. Andrikopoulos, R. Gornez, K. Nakamura, and G. Nikolakopoulos, "A survey on the application trends of home service robotics," in *2018 IEEE international conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2018, pp. 1999–2006.

[4] R. Tellez, "A thousand robots for each student: Using cloud robot simulations to teach robotics," in *Robotics in Education*, M. Merdan, W. Lepuschitz, G. Koppensteiner, and R. Balogh, Eds. Cham: Springer International Publishing, 2017, pp. 143–155.

[5] G. Lopez-Nicolas, A. Romeo, and J. J. Guerrero, "Simulation tools for active learning in robot control and programming," in *2009 EAEEIE Annual Conference*, 2009, pp. 1–6.

[6] U. Dakeev, R. R. Pecen, F. Yildiz, I. I. Basith, S. M. Obeidat, and L. E. Sowell, "Development of virtual reality robotics laboratory simulation," in *2022 ASEE Zone IV Conference*, no. 10.18260/1-2–44729. Vancouver: ASEE Conferences, May 2022, https://peer.asee.org/44729.

[7] E. Fabregas, G. Farias, S. Dormido-Canto, M. Guinaldo, J. Sánchez, and S. Dormido Bencomo, "Platform for teaching mobile robotics," *Journal of Intelligent & Robotic Systems*, vol. 81, pp. 131–143, 2016.

[8] E. Cervera, P. Martinet, R. Marin, A. A. Moughlbay, A. P. Del Pobil, J. Alemany, R. Esteller, and G. Casan, "The robot programming network," *Journal of Intelligent & Robotic Systems*, vol. 81, no. 1, pp. 77–95, 2016.

[9] G. A. Casañ and E. Cervera, "The experience of the robot programming network initiative," *Journal of Robotics*, vol. 2018, 2018.

[10] D. Roldán-Álvarez, J. M. Cañas, D. Valladares, P. Arias-Perez, and S. Mahna, "Unibotics: open ros-based online framework for practical learning of robotics in higher education," *Multimedia Tools and Applications*, vol. 83, no. 17, pp. 52 841–52 866, 2024.

[11] D. Roldán-Álvarez, S. Mahna, and J. M. Cañas, *A ROS-based Open Web Platform for Intelligent Robotics Education*, ser. International Conference on Robotics in Education (RiE). Springer, 2021, pp. 243–255.

[12] D. Roldán-Álvarez, J. M. Cañas, D. Valladares, P. Arias-Perez, and S. Mahna, "Unibotics: open ros-based online framework for practical learning of robotics in higher education," *Multimedia Tools and Applications*, vol. 83, no. 17, pp. 52 841–52 866, 2023.

[13] D. Roldán-Álvarez, S. Mahna, and J. M. Cañas, "A ros-based open web platform for intelligent robotics education," in *Robotics in Education: RiE 2021 12*. Springer, 2022, pp. 243–255.

[14] J. M. Cañas, D. Martín-Martín, P. Arias, J. Vega, D. Roldán-Álvarez, L. García-Pérez, and J. Fernández-Conde, "Open-source drone programming course for distance engineering education," *Electronics*, vol. 9, no. 12, p. 2163, 2020.

[15] J. M. Cañas, E. Perdices, L. García-Pérez, and J. Fernández-Conde, "A ros-based open tool for intelligent robotics education," *Applied Sciences*, vol. 10, no. 21, p. 7419, 2020.

[16] E. Gonzalez, O. Alvarez, Y. Diaz, C. Parra, and C. Bustacara, "BSA: A complete coverage algorithm," in *proceedings of the 2005 IEEE International Conference on Robotics and Automation*. IEEE, 2005, pp. 2040–2044.

[17] I. A. Şucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library," *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, December 2012, https://ompl.kavrakilab.org.

[18] Z. Kingston, M. Moll, and L. E. Kavraki, "Exploring implicit spaces for constrained sampling-based planning," *Intl. J. of Robotics Research*, vol. 38, no. 10–11, pp. 1151–1178, Sep. 2019.

[19] F. Stöckl, S. Müller, and M. Strand, "Tailored Robotics Lab Sessions - Accommodating Undergraduate Students with Diverse Skill Levels," *15th International Conference on Robotics in Education (RiE 2024)*, Apr. 2024.