applied sciences

MDPI

Article

A Free Simulation Environment Based on ROS for Teaching Autonomous Vehicle Navigation Algorithms

Marco Antonio Chunab-Rodríguez [1], Alfredo Santana-Díaz [1], Jorge Rodríguez-Arce [1,2,*], Emilio Sánchez-Tapia [3,4] and Carlos Alberto Balbuena-Campuzano [1]

[1] Escuela de Ingeniería y Ciencias, Tecnologico de Monterrey, Ave. Eugenio Garza Sada 2501, Monterrey 64849, Mexico; marcochunabr@gmail.com (M.A.C.-R.); asantana@tec.mx (A.S.-D.); carlos_balbuena@tec.mx (C.A.B.-C.)

[2] Facultad de Ingeniería, Universidad Autónoma del Estado de México, Ciudad Universitaria Cerro de Coatepec s/n, Toluca 50110, Mexico

[3] CEIT-Basque Research and Technology Alliance (BRTA), Manuel Lardizabal 15, 20018 Donostia, Spain; esanchez@ceit.es

[4] Tecnun, University of Navarra, Manuel Lardizabal 13, 20018 Donostia, Spain

* Correspondence: jrodrigueza@uaemex.mx or jorge_arce@tec.mx

Abstract: In recent years, engineering degree programs have become fundamental to the teaching of robotics and incorporate many fundamental STEM concepts. Some authors have proposed different platforms for teaching different topics related to robotics, but most of these platforms are not practical for classroom use. In the case of teaching autonomous navigation algorithms, the absence of platforms in classrooms limits learning because students are unable to perform practice activities or cannot evaluate and compare different navigation algorithms. The main contribution of this study is the implementation of a free platform for teaching autonomous-driving algorithms based on the Robot Operating System without the use of a physical robot. The authors present a case study using this platform as a teaching tool for instruction in two undergraduate robotic courses. Students evaluated the platform quantitatively and qualitatively. Our study demonstrates that professors and students can carry out different tests and compare different navigation algorithms to analyze their performance under the same conditions in class. In addition, the proposed platform provides realistic representations of environments and data visualizations. The results claim that the use of simulations helps students better understand the theoretical concepts, motivates them to pay attention, and increases their confidence.

Keywords: mobile robotics; autonomous driving; Robot Operating System (ROS); educational robotics; educational innovation; professional education; higher education

check for updates

Citation: Chunab-Rodríguez, M.A.; Santana-Díaz, A.; Rodríguez-Arce, J.; Sánchez-Tapia, E.; Balbuena-Campuzano, C.A. A Free Simulation Environment Based on ROS for Teaching Autonomous Vehicle Navigation Algorithms. *Appl. Sci.* **2022**, *12*, 7277. https://doi.org/10.3390/app12147277

Academic Editors: Dimitris Mourtzis and Anja Richert

Received: 15 June 2022
Accepted: 18 July 2022
Published: 20 July 2022

1. Introduction

STEM is the acronym for Science, Technology, Engineering, and Mathematics. In recent years, this educational area has grown due to the fact that STEM graduates are in high demand in the job market. For example, in the United States, there are 26 million jobs that require knowledge in these areas [1]. Robotics is fundamentally rooted in STEM education. Robotics can be described as a discipline that addresses a class of mechatronic systems called robots, capable of performing different industrial, scientific, and commercial applications. This discipline has become a critical part of industrial mechatronic systems in recent years. That is why professional degree programs such as mechatronics engineering, electronic engineering, and computer engineering have become fundamental to teaching robotics and incorporate many of the basic STEM concepts [2].

Due to constant changes in technologies and robotics in different industries, adequately preparing future engineers is critical. To carry this out, teachers must develop and implement innovative ways to teach concepts and use new technologies to optimize educational processes [3]. In education, when information is transmitted with the ideal didactic tools, the knowledge acquired will be retained longer than the standard methods typically used [4].

One innovative way of teaching robotics concepts and offering practice activities is by using robotics platforms. However, these platforms present various challenges in terms of accessibility, cost, and flexibility. An accessible platform is helpful for obtaining or replacing components if necessary and aiding students in education. Likewise, the flexibility of these educational tools allows for their adaptation to different teaching methodologies and curricula. Finally, cost is a significant consideration. In many cases, it is not feasible for educational institutions with many students to have invest in more than one platform to serve the needs of numerous classes [5]. Consequently, to develop a tool to perform simulations in different environments and evaluate various autonomous-driving navigation algorithms in the classroom ought to be an objective.

The main objective of this work is to propose a platform for simulating autonomous driving algorithms that helps students improve their academic performance. Using the proposed platform, teachers and students can analyze and compare various existing navigation algorithms under the same conditions in a classroom setting. The contribution is a free platform and scenarios that teachers and students can download as an academic tool to perform autonomous driving laboratory practices and reinforce the theoretical concepts.

The proposed platform has some advantages for teaching autonomous driving. For example, robotics teachers will be able to focus on this topic, and students will not be distracted by other activities such as platform design, robot design, programming, among others. Similarly, students can carry out practical simulations and compare the results of different navigation algorithms in real environments under the same conditions. In order to demonstrate how teachers could use this free platform in the classroom, the authors present a case study comparing two local route planning algorithms for three different scenarios: a hospital, a warehouse, and a laboratory. At the end of this activity, the students evaluated the proposed platform quantitatively and qualitatively, these evaluations were designed to study the impact of the platform on the student's academic performance and to identify the improvements of its use in the class. Finally, conclusions and future work are outlined.

1.1. Teaching Robotics

Several platforms have been developed in previous research to teach various robotics topics and concepts. Each platform has different components, tools, and learning objectives. One of the tools that teachers use most for teaching robotics is LEGO™. The main advantages of this platform include its accessibility and the wide variety of kits, which can be adapted to different projects. Martínez et al. developed a platform using LEGO NXT™ to teach reinforcement learning (RL), which can address a wide variety of problems related to robotics [6]. Likewise, Rosillo et al. developed the first platform combining three technological tools: LEGO™ Kits, Matlab™, and Simulink™. This platform rearranges the code in Matlab™, translates it into C++ code with Simulink™ and transfers it to the LEGO™ robots via Wi-Fi. The authors developed a second educational robotics tool using Matlab and ROS. This framework develops and simulates complex robots [7].

One of the most widely used physical platforms is TurtleBot [8]. The authors describe how this platform is used in courses taught at the Katholieke Universiteit Leuven in Belgium, providing an overview of the main functionalities, and suggesting improvements to reduce student learning curves. They wrote the curriculum and reported the learning results from two courses that used the platform, which were positive according to student feedback.

Another technological resource for teaching robotics is virtual reality (VR). VR allows students to perform simple electronic laboratory experiments safely using relevant tools, instruments, components, and virtual applications. Using virtual scenarios and study materials allows students to learn by participating in a distance learning process and introduces other Industry 4.0 concepts such as the digital twin, Big Data, cloud computing, IoT (Internet of Things), and cybersecurity. Another significant advantage of using VR in teaching is in eliminating the risk of potentially harmful experimental strategies as well as minimizing cost and time. An example is the Virtual Mechatronics Laboratory (ViMeLa) project, developed by the Lodz University of Technology, the Ss. Cyril and Methodius University in Skopje, the University of Tartu, and the University of Pavia, which implemented the teaching and learning of electronics in higher education institutions [9]. Their results demonstrate that VR is of great assistance to the process of education and facilitates the acquisition of knowledge by putting theory into practice.

In addition, there exist platforms that provide excellent content, activities, and tools for learning robotics. One such platform is Universal Robots Academy, an open-access platform containing a collection of exercises and activities to learn robotics and electronics, with a focus on engineering. The activities found on this platform cover topics such as mobile robots, industrial robotics, and even drones [10]. One of the most exciting perspectives in teaching robotics is video games, which serve as comprehensible learning tools. Robles and Quintero [4] developed a platform with various interactive video games that store all the information generated by a player and then decoded, analyzed, and evaluated it using an intelligent system. At the end of this analysis, the system displays all the statistical information calculated during the interaction of users with the game and provides suitable suggestions to reinforce the topics covered in the games. The platform aims to develop high school students' skills in mathematics; however, these platforms can teach innovative and interactive robotics concepts dynamically.

Finally, these types of platforms allow for distance learning and are particularly useful for situations such as the COVID-19 pandemic, which is the reason why many schools suspended their face-to-face educational activities and began teaching online. These platforms allow laboratory practices to be carried out and, thus, reinforce theoretical knowledge. Practical activities can be carried out in virtual laboratories or simulations. Technology in teaching has increased due to the pandemic. Specifically, robotics and digital teaching platforms have been critical for students to learn various concepts and subjects. On the other hand, incorporating these platforms into the classroom remains a challenge due to their cost, availability, and learning curve. In addition, many of them have to be redesigned to meet specific learning goals.

1.2. Autonomous Driving in Robotics

Autonomous driving is a method that consists of operating a vehicle without active driver control. It involves multiple variables for the proper operation of the mobile robot such as reading sensors, following a trajectory, maintaining a safe distance from other vehicles or obstacles, and speed control, among others.

Because this subject has paved the way for thousands of applications in daily life, industry, commerce, and more, research related to autonomous driving has increased in recent years. STEM courses have garnered more interest in different topics related to robotics, such as teaching and applying autonomous driving models. However, developing teaching strategies that facilitate knowledge acquisition in an innovative way to put concepts and theory into practice has proven challenging for robotics teachers [11].

One of the relevant autonomous vehicle components is sensors. In autonomous vehicles, these include cameras, LiDAR radar, sonar, global positioning systems (GPS), an inertial measurement unit (IMU), and wheel odometry. The sensors collect data to be analyzed by the computer of the vehicle to control its direction, speed, and braking. However, sensor fusion is also applied in autonomous vehicles, combining data from disparate sources to assemble information coherently. The resulting information is more reliable when these sources or sensors are used individually. This approach is essential when combining different types of data. For example, it is crucial to have a camera to clone human vision in autonomous vehicles. However, it is preferable to obtain distance data from sensors such as LiDAR or radar. Therefore, the combination of camera sensors and

LiDAR or radar data is essential. Moreover, combining LiDAR and radar data provides quantifiable information on the distance between an obstacle and the vehicle, or the distance from different objects in the environment [12].

Likewise, another crucial consideration in operating autonomous vehicles is the algorithms that process sensor data for decision making. Levinson et al. [13] developed a series of algorithms for a commercial vehicle. The vehicle had to perform the following tasks: unsupervised laser calibration, mapping and localization, object recognition, and trajectory planning. Unsupervised laser calibration required a multi-beam laser to retrieve the optimal parameters for each beam's orientation and distance-response function. This method allows for the simultaneous calibration of tens or hundreds of beams, each with specific parameters. In addition, the extrinsic position of the sensor in relation to the robot's coordinate frame is recovered. Importantly, no specific calibration objective is required; it is based solely on the weak assumption that points in space tend to be on contiguous surfaces.

For mapping and location tasks, data from GPS, IMU, and LiDAR sensors can generate a high-resolution terrain map of the vehicle's location. The maps used in vehicles are grid-based orthographic projections of LiDAR remittance responses on the ground plane.

Regarding object recognition, each LiDAR scan was first segmented using depth information. The segments were inserted into a standard Kalman filter that includes position (x, y) and velocity (x′, y′) in their variable states. The path classification was executed by applying two separate boosting classifiers: one indicating the object's shape in each path frame and another using motion descriptors along the entire path. These predictions were combined using a discrete Bayes filter. The test results were favorable since the vehicle drove along routes autonomously for hundreds of miles under different conditions. However, the driver must always be present for safer driving, considering that some driving must also be performed manually upon the occasion [10].

1.3. Algorithms for Autonomous Driving

In education, the different platforms designed and tested to teach various robotics topics include electronic platforms such as, LEGO™, Arduino, and Raspberry Pi. In particular, the LEGO™ platform has been used to teach robotics, but it has some limitations regarding autonomous driving subjects. The LEGO™ control unit (Brick module) can be programmed using standard LEGO™ block-based software or other open-source programming languages. However, block-based software requires considerable programming experience, limiting the complexity of the program and the learning ability of the algorithms.

On the other hand, vision-based control is critical for autonomous driving. For example, the LEGO™ platform lacks a suitable camera. Furthermore, its hardware is not powerful enough to process images in real-time. Otto et al. [14] designed a platform to teach Robotics using the LEGO™ platform combined with Raspberry Pi, Matlab™, Simulink™, a low-cost webcam, and an external power source. Unlike the standard LEGO™ platform, this new design provides enough computing power to evaluate image data in real-time. The operation starts when the Raspberry Pi emulates the UART communication protocol on the LEGO™ platform and transmits the continuously calculated control inputs. A Pi-sensor controller transfers control input to LEGO™ motors. Finally, Matlab™ and Simulink™ allow for the implementation of the autonomous driving algorithms on the Raspberry Pi card and robot control. The main advantage of this system is the possibility to add several sensors. The authors outline the limitations of their system (such as the lack of a camera) for teaching autonomous driving.

Today, there are various navigation algorithms with distinct characteristics and functions. When teaching, it is pertinent to compare the different algorithms to analyze their behavior and performance, but it is not always possible because the scenarios and test conditions are not the same.

Naotunna and Wongratanaphisan [15] conducted a performance analysis of three navigation algorithms: Dynamic-Window Approach (DWA), Timed Elastic Band (TEBand), and Elastic Band (EBand). The authors compare the performance of these three local planners using ROS software. The experiment was conducted in a selected area, approximately 145 m^2 of the first floor of the main building at the School of Engineering of Chiang Mai University. The performance of the three algorithms was analyzed by navigating the robot on four pre-defined trajectories. For each algorithm, the robot performed 20 tests divided into two sections on the path. The first ten tests were conducted in an unobstructed scenario, and the subsequent ten tests in the same scenario with obstacles. The study concluded that the tracking accuracy values of the DWA and TEBand algorithms are higher than those of the EBand algorithm. Likewise, considering the average time taken to complete the journey, the results showed that the DWA algorithm took less time to complete the route than the TEBand algorithm. Although the EBand algorithm generates the shortest route, the maximum speed generated is less than the maximum speed set. Finally, the position and orientation errors measured at the target location showed that the EBand algorithm has greater precision, and the TEBand algorithm had the maximum error deviation. However, the disadvantages of this research for teaching are that students require access to the real robotic platform and a large open space to perform tests with the robot.

Pimentel and Aquino [16] compared four different local route planning algorithms: Base Local Planner, DWA, TEBand, and EBand. The experiments were carried out using a front laser sensor and a rear laser sensor. A 3D-depth camera was added for the evaluations. Two scenarios were used in these tests: the first, a simulated scenario that evaluated navigation in a static environment without obstacles; and the second, a simulated scenario that evaluated navigation with static 3D objects outside the range of the sensors. After performing the corresponding tests, the authors noted that the DWA and Base Planner algorithms presented the worst results, despite being the most used in the ROS environment. At the end of this study, the authors demonstrate that ROS can be used to compare the performance of different autonomous algorithms without requiring a physical mobile robot or the space to perform the tests. In this way, ROS could be used as a learning tool in the classroom to put theoretical knowledge into practice.

However, many of these platforms must be implemented by students during class, which distracts them from assignments that do not meet the learning objectives. Additionally, the absence of platforms in class limits learning because students cannot carry out autonomous driving practice activities; thus, they cannot compare the performance of different algorithms. Another important consideration is that many of the platforms are physical. Although universities acquire them, sometimes students cannot access large spaces to perform different tests. As a consequence, there is a challenge to develop a learning tool to run simulations in different environments and evaluate different navigation algorithms in real time.

Therefore, this study proposes the use of ROS as a tool that students can utilize in class to perform different practices and reinforce learning robot implementation in virtual scenarios. The advantage of using classroom simulations is that students can compare the performance of different navigation algorithms under the same conditions.

2. Materials and Methods

2.1. Using ROS to Teach Robotics

There are multiple platforms currently used to teach robotics at different educational levels. Such platforms are used for teaching mobile robotics [8,17], STEAM concepts [18], control engineering robotics [19], manipulators [20,21], programming skills [22,23], smart sensors [24], robot vision [25], and survival and behavior analysis [26], among others. There is a wide range of hardware platforms that support robotics courses and laboratory practices. As for the different programming environments, Matlab™ [7] and C [27] are the prevailing options in classrooms. In recent years, free open-source packages have emerged, which allow for the implementation of virtual scenarios for the simulation of mobile robots. Such is the case of Robot Operating System (ROS) [28].

Figure 1 shows the basic operation of ROS. The Robot Operating System (ROS) is an open-source working environment initially developed to be installed on a Linux operating system. It is used for programming and robot simulations. ROS is organized by execution units (called nodes), which communicate by publishing/subscribing messages, offering services, and/or executing actions. ROS includes connectivity with physical and real hardware (for example sensors or actuators) and algorithm implementation of various control types, navigation, map creations, etc. [29]. Its Gazebo module (simulation) and RVIZ (ROS-VIsualiZer) complete a flexible framework that allows teaching with simulated elements and minimal effort to transfer and test developments on real hardware.

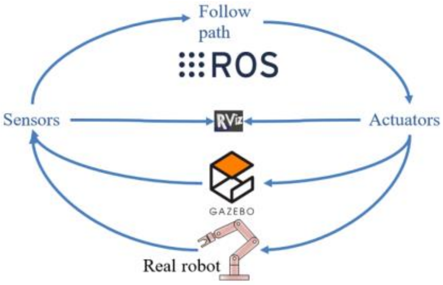


Figure 1. ROS is an open-source working environment, which includes connectivity with physical and real hardware (actuators and sensors). Gazebo (run 3D robot simulations) and RVIZ (allows a 3D visualization) modules complete a flexible framework that allows simulations and visualization in combination with ROS.

Gazebo is a 3D environment for running robot simulations in combination with ROS. In this way, users can test algorithms, design and test mobile robots, and train an AI (Artificial Intelligence) system using virtual representations of realistic scenarios. This offers the ability to simulate robot movements accurately and efficiently in complex indoor and outdoor environments [30]. The Gazebo graphical user interface (GUI) can be observed in Figure 2.

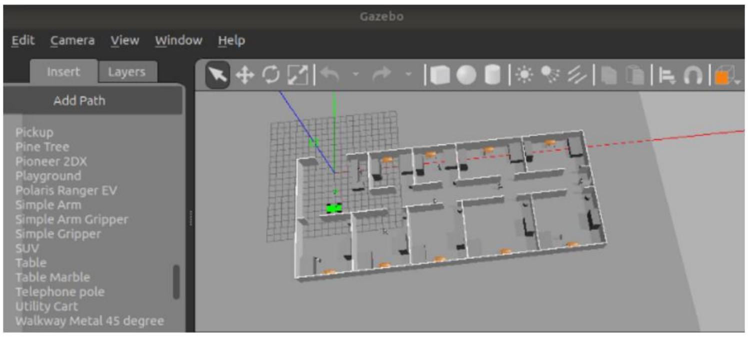
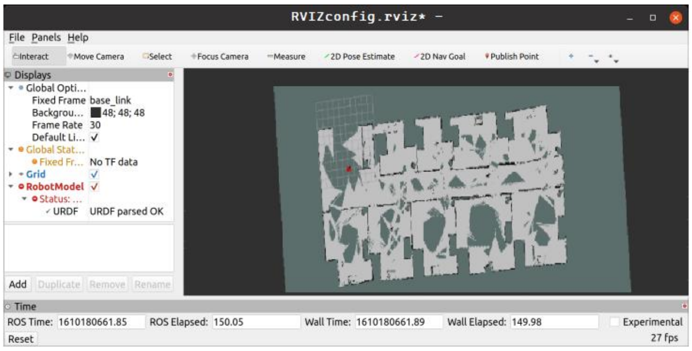


Figure 2. Gazebo main screen. Gazebo can run 3D robot simulations.

ROS Visualization (RVIZ) is a 3D visualization tool for ROS applications; its main screen is shown in Figure 3. RVIZ displays the mobile robot model, captures robot sensor data, and reproduces the captured data. It can display data from cameras, lasers, and 3D and 2D devices as images and point clouds [28].

It is more beneficial to use ROS to teach robotics because it has functions for hardware abstraction, different device drivers, communication among multiple machine processes, and testing and visualization tools. However, the primary feature of ROS lies in how the software runs and communicates, allowing for the design and implementation of the algorithms without knowing how the hardware operates in detail [31]. This section reviews some of the systems in which the ROS platform has been used in robotics teaching.



Global Opti...
Fixed Frame base_link
Backgrou... 48; 48; 48
Frame Rate 30
Default Li... ✓
Global Stat...
Fixed Fr... No TF data
Grid ✓
RobotModel ✓
Status: ...
URDF URDF parsed OK

ROS Time: 1610180661.85 ROS Elapsed: 150.05 Wall Time: 1610180661.89 Wall Elapsed: 149.98 □ Experimental
27 fps

Figure 3. RVIZ main screen. This tool allows a 3D visualization for ROS applications.

Niu et al. [32] developed a toolkit for teaching mobile robotics through the Matlab™ and Simulink™ package add-on using ROS and the Gazebo simulator to improve learning efficiency in physical robotic simulation environments. Access to virtual sensors and actuators has been included in this toolkit, and communication details are hidden to allow students to focus on programming and debugging their autonomous driving algorithms. The basic architecture of the platform consists of three main software blocks: Matlab™ and Simulink™, Robot Operating System (ROS), and Gazebo. The algorithms developed in Matlab™ and Simulink™ interact with ROS, which connects Matlab™ and the Gazebo simulator.

An activity was assigned to a group of robotics students to evaluate the platform. In this activity, students used LiDAR sensor data and the robot's odometry data to avoid it colliding with different obstacles independently and to make it arrive at a pre-determined final position. This activity was evaluated by measuring the time taken to complete a trajectory between the initial and final positions. The time obtained by each student was compared to find the best time [32].

The EUROPA platform (Educational ROS Robot Platform) continues the tendency to apply ROS in education. This platform consists of a low-cost, two-wheeled robot with differential traction for which its central controller is a Raspberry Pi 3 B+ card. It is perfectly scalable and adapts to different educational levels and curricula. Its design allows programming with tools according to the educational levels and course curricula. EUROPA uses the ROS platform as its primary communication and control software. Raspberry Pi controls the hardware (e.g., motors and other actuators) and collects data from the odometry sensors and camera. All the controllers responsible for controlling the two direct current motors, the servomotors, the ultrasonic sensor, and the LIDAR are installed on Raspberry Pi. Finally, Raspberry Pi also hosts several Python scripts that act as nodes. This platform is currently being evaluated in secondary schools in Central Macedonia (Greece), where a pilot program is being run [5].

Cañas et al. [31] developed another educational kit for teaching robotics called Robotics-Academy Design. This kit includes a set of independent exercises that propose specific problems regarding Autonomous Robotics. The student must develop the algorithms and program the robot to solve the problem correctly. Its main components are the Python programming language, ROS Middleware, the Gazebo simulator, and a one-code template per exercise (Python ROS node and Jupyter Notebooks). This kit was used and validated in several undergraduate engineering courses at Rey Juan Carlos University in Spain. More than 150 students participated in this study, and more than 95% of them positively evaluated its use in surveys. It was also validated in a study involving 221 pre-university students in which the quantitative analysis performed shows that the Robotics-Academy tool had a positive impact on the learning outcomes of students compared to the control group.

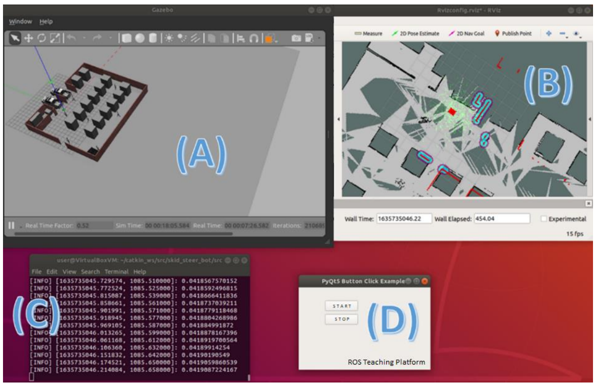
Previous studies have reported favorable results for ROS as an educational tool [7,31]. This explains its use among professors who design and implement tools such as virtual laboratories for learning robotics courses, solving the limitations of some educational institutions, and solving the problem that some students do not have access to a platform or physical spaces.

Nevertheless, to the authors' knowledge, most of these platforms are focused on general topics of robotics and users cannot run simulations under the same conditions. Moreover, these previous platforms have not been used to teach autonomous driving algorithms or evaluated to study their impact on the students' academic performance.

2.2. Development of a Free Platform for Teaching Autonomous Driving Algorithms

The platform in the present work uses the ROS and Gazebo simulator, together with the mapping algorithms and global and local route planning libraries. This free platform allows students to perform simulations practically without designing or implementing a robot or scenarios from scratch. Consequently, the student can focus on the learning of autonomous driving theory and put it into practice.

The proposed platform has an interface for exchanging messages. Therefore, the ROS and Gazebo programs can communicate with each other during the ongoing process. For example, the data processed by the LiDAR sensors are sent to the ROS server for further processing and to move the motors of the virtual mobile robot. During the simulations, Gazebo and RVIZ in this teaching platform provided students with realistic representations of the environments used and the movements of the mobile robot (see Figure 4). The proposed free platform incorporates three different scenarios and a mobile robot model. Students can evaluate the performance of different local route planning algorithms. These algorithms can be native to ROS and is user generated. To change algorithms in simulations, students only have to select the name of the algorithm they want to use. Similarly, to modify the scenario, students must only select the name of the scenario they want to use.



```
[INFO] [1635735045.729574, 1085.510000]: 0.0418567570152
[INFO] [1635735045.772524, 1085.525000]: 0.0418592496815
[INFO] [1635735045.815087, 1085.539000]: 0.0418666411836
[INFO] [1635735045.858661, 1085.561000]: 0.0418737039211
[INFO] [1635735045.901991, 1085.571000]: 0.0418779118468
[INFO] [1635735045.918945, 1085.577000]: 0.0418804268986
[INFO] [1635735045.969105, 1085.587000]: 0.041884991872
[INFO] [1635735046.013265, 1085.599000]: 0.0418878167396
[INFO] [1635735046.061168, 1085.612000]: 0.0418919780564
[INFO] [1635735046.106360, 1085.632000]: 0.04189914254
[INFO] [1635735046.151832, 1085.642000]: 0.04190190549
[INFO] [1635735046.174521, 1085.650000]: 0.0419059860539
[INFO] [1635735046.214084, 1085.658000]: 0.0419087224167
```

Figure 4. The proposed platform running a simulation: Gazebo render map (**A**), RVIZ planning map (**B**), data of robot measurements in real time (**C**), and user GUI (**D**).

All steps for platform installation and instructions for using the available scenarios and algorithms are in a GitHub repository at the following link: http://github.com/marcochunab/skid_steer_bot (accessed on 17 July 2022).

2.3. Design of the Real Closed Scenarios

As mentioned in previous sections, this work focuses on the ROS platform for teaching autonomous driving algorithms in robotics courses. One of the main advantages of using this tool in the classroom is that students can test and compare different algorithms under the same conditions: for example, the same scenario. The proposed platform could be a teaching support tool (virtual laboratory) for teachers and students to perform autonomous driving practices in robotics courses without accessing physical platforms or test spaces.

As an initial proposal, this work considers three virtual workspaces with different conditions and characteristics for testing and comparing autonomous driving algorithms. These free scenarios are available to the academic and scientific community for teaching or research applications.

Reports collected by the International Federation of Robotics (IFR) indicate that, between 2016 and 2019, there was an increase in the use and demand for service robots in medicine, agriculture, logistics, and defense compared to previous years. Based on these data, health, logistics (supply chain), and industry were chosen for the creation of the three virtual scenarios on the ROS platform [33]. Figure 5 summarizes the procedure for implementing the platform and scenarios.

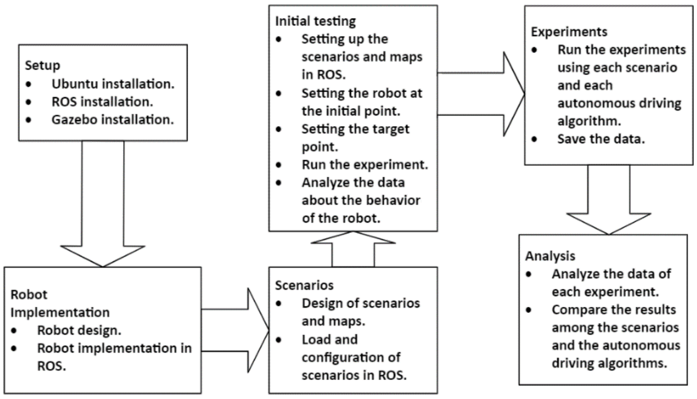

Setup
• Ubuntu installation.
• ROS installation.
• Gazebo installation.
Initial testing
• Setting up the scenarios and maps in ROS.
• Setting the robot at the initial point.
• Setting the target point.
• Run the experiment.
• Analyze the data about the behavior of the robot.
Experiments
• Run the experiments using each scenario and each autonomous driving algorithm.
• Save the data.
Robot Implementation
• Robot design.
• Robot implementation in ROS.
Scenarios
• Design of scenarios and maps.
• Load and configuration of scenarios in ROS.
Analysis
• Analyze the data of each experiment.
• Compare the results among the scenarios and the autonomous driving algorithms.

Figure 5. The methodology used for the development and implementation of the platform and virtual scenarios.

The Gazebo tool was used for the scenario design. The first proposed scenario is a hospital healthcare floor. In this scenario, a mobile robot can help carry out activities where direct human contact must be avoided, for example, during the current COVID-19 pandemic. In this scenario, the following spaces were considered: a reception area, nine medical care rooms, and a corridor connecting the rooms with the reception area (there are nine entrances to medical care rooms). In each room, there are different obstacles with enough space for the robot to circulate between them. In Figure 6, one observes the reception area (A), the rooms (B), and the corridor (C). The total area is 15 m wide and 45 m long (675 m^2). To simulate a realistic hospital in Gazebo, we used models with pre-existing people placed throughout the scenario. Movements were not counted independently.

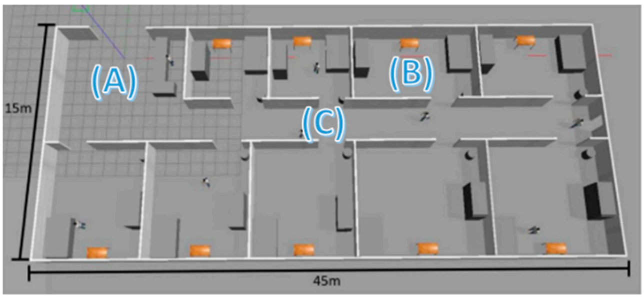


Figure 6. Map of the scenario #1—hospital: reception (**A**), room (**B**), and corridor (**C**).

The second proposed scenario is a loading and unloading warehouse. Here, a mobile robot can be helpful for carrying out logistics activities that require the transport of items between different areas. This scenario had an unloading area and a loading area, as well as a series of blocks aligned in two rows in the central part of the scenario, simulating the existing products in the warehouse. People were also scattered in different places to simulate the flow of personnel. Figure 7 shows the loading area (A), the unloading area (B), and the storage area (C). The total area is 20 m wide and 25 m long (500 m^2).

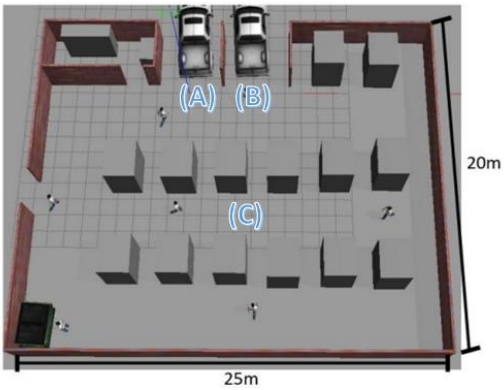


Figure 7. Map of the scenario #2—warehouse unloading (**A**), loading (**B**), and storage areas (**C**).

Unlike the hospital scenario, the warehouse has more open spaces and wider corridors around the blocks. However, there are also reduced or narrow corridors between each block. The aisles are wide enough for the robot to navigate in all cases. On the other hand, this scenario does not have entrances to different rooms that the robot must enter.

The final proposed scenario is an industrial manufacturing laboratory. The laboratory has a central area with several machines in the surrounding area and a warehouse. Figure 8 shows the machine zones (A), (B), (C), (D), and a warehouse space (E). The area in total is 20 m long and 25 m wide (500 m^2).

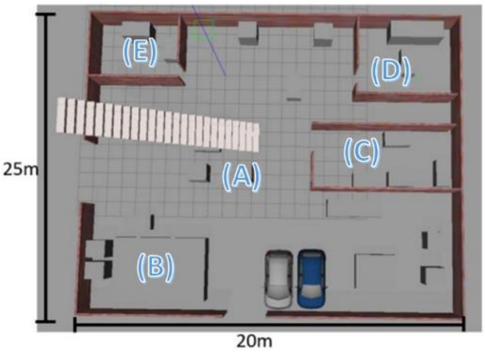


Figure 8. Map of the scenario #3—Industrial manufacturing area: machine zone 1 (**A**), machine zone 2 (**B**), machine zone 3 (**C**), machine zone 4 (**D**), and warehouse (**E**).

Table 1 summarizes the differences that exist. among the proposed scenarios. Additionally, two different paths or trajectories were established in each scenario to be evaluated; one trajectory called "simple path" and another called "complex path". On the simple path, there are less than 15.0 m between the initial position and the target point. Furthermore, there are not obstacles interfering with the movements of the mobile robot. On the complex path, there are more than 15.0 m between the initial and final position. The complex path includes various obstacles that the mobile robot must evade autonomously to reach the target point.

Table 1. Differences among the three scenarios.

Scenario	Wide Hallways	Narrow Hallways	Entrances/Exits	Open Areas
Hospital	X	√	√	X
Warehouse	√	√	X	√
Laboratory	√	X	X	√

A virtual mobile robot with four wheels, front steering, and dimensions of 65 cm long and 46 cm wide was implemented for testing, as shown in Figure 9. It has two sensors for mapping; the first is the LiDAR sensor, which determines the distance from an object using a laser emitter. The second sensor is a stereoscopic camera that is not used for autonomous navigation or environmental mapping; instead, its purpose is for the user to visualize the virtual environment in the current robot position.

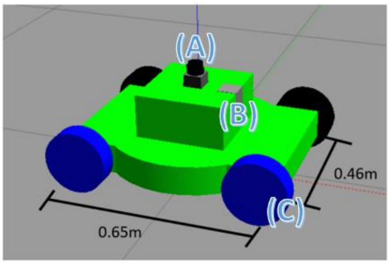


Figure 9. A virtual mobile robot with four wheels used in the simulations with the following components: LiDAR sensor (**A**), stereoscopic camera (**B**), and encoders (**C**).

Finally, the odometry data are generated from the encoders and the Gazebo platform, which provide the position of X and Y coordinates locating the robot in the virtual scenario. Figure 10 shows a flowchart of the steps to perform a simulation.

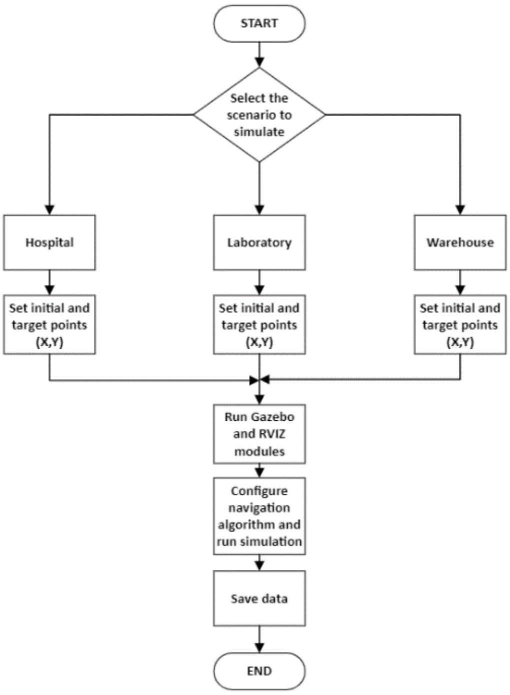

START
Select the scenario to simulate
Hospital
Laboratory
Warehouse
Set initial and target points (X,Y)
Set initial and target points (X,Y)
Set initial and target points (X,Y)
Run Gazebo and RVIZ modules
Configure navigation algorithm and run simulation
Save data
END

Figure 10. Steps to run a simulation. First, the users have to select the scenario. After, the initial and target point must be set. Finally, the users have to configure the algorithm to run.

3. Results

The objective of this case study is to describe the use of the platform in the classroom setting to perform the analysis and comparison of different autonomous driving algorithms. In addition, two evaluations were performed: quantitative and qualitative. For this study, two local route planning algorithms, the Time Elastic Band (TEBand) and the Elastic Band (EBand), were chosen as case studies because previous research by other authors obtained the best performance results by using these two algorithms [15]. These algorithms were implemented on the proposed platform to run a set of simulations using three scenarios (hospital, warehouse, and industrial manufacturing area) with simple and complex trajectories. The following experimental groups were tested:

- SP-TEBand → Simple path using the TEBand algorithm;
- SP-EBand → Simple path using the EBand algorithm;
- CP-TEBand → Complex path using the TEBand algorithm;
- CP-EBand → Complex path using the EBand algorithm.

This study was conducted during two undergraduate robotics courses (spring semester 2022). The study was divided into two phases, and both courses followed the same protocol. The first phase consisted of an explanation of the theoretical concepts about autonomous driving algorithms. In this phase, the professor used a traditional teaching method with oral presentations in combination with PowerPoint slides. This part included a written test in which students had to answer a set of questions about the theoretical concepts of robotics.

The second phase of this study continued with a demonstration of how the proposed platform works. In a classroom setting, the professor ran a set of simulations to show students the performance parameters of each algorithm in each scenario under the same conditions. Thirty tests were performed with each algorithm for each of the two paths in each scenario. In each test, the robot traveled along the route autonomously from the initial position to the target point. The obstacles were always static in each scenario. This educational approach allowed students to learn and compare the performance of different autonomous driving algorithms in order to reinforce theoretical knowledge. This part also included a written test in which students had to answer questions about the theoretical concepts. The simulations were performed on a personal computer with an Intel™ Core™ i7 processor, 8 GB of RAM, and an NVIDIA™ GeForce™ GTX 1050 graphics card.

Three performance indicators were measured during this study:

1. Total distance: This refers to the total distance that the robot travels from the initial position until it stops at the target point. This indicator was measured in meters. The real-time location of the robot during the simulation to calculate the total distance was used with the data calculated by the virtual odometry system (see Figure 11).
2. Time: This is the value in seconds that the robot requires to proceed from the initial position to the target point of the planned route. Time stops when the robot reaches the final position (see Figure 12).
3. Mean error: This value is measured in meters. It represents the measured error with respect to the final position of the robot and the target point (in other words, how close the mobile robot is to the endpoint). This indicator is presented in meters. The odometry information was used to know the coordinates of the robot at the end of the trajectory. The distance between the location of the robot and the target point was calculated using the following formula: $d = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2}$, where X_2 and Y_2 are the endpoint coordinates and X_1 and Y_1 are the coordinates of the initial position of the robot (see Figure 13).

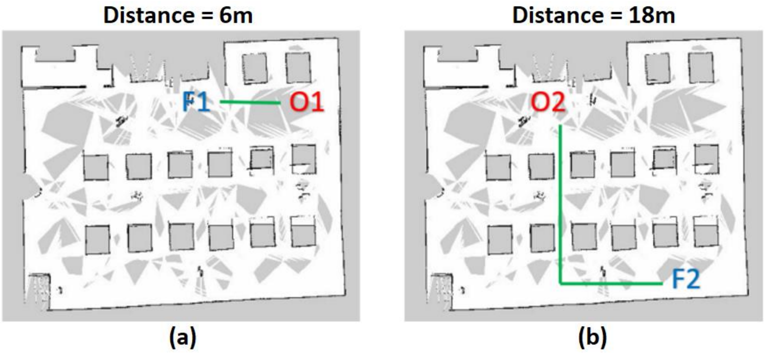


Figure 11. Example of paths in the same scenario: simple path (**a**) and complex path (**b**).

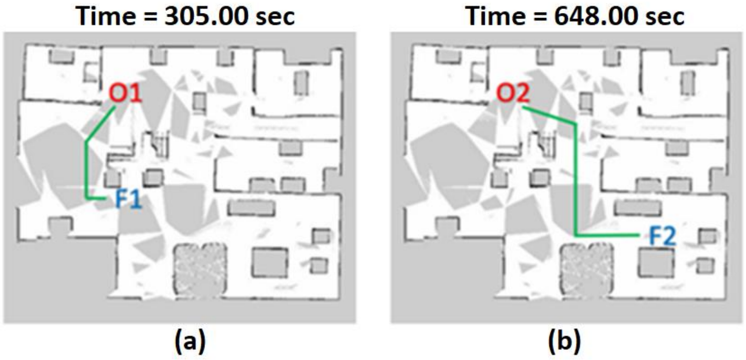


Figure 12. Example of the time for different paths. In this example, in the simple path (**a**), the robot requires less time to arrive to the target point in comparison with the complex path (**b**).

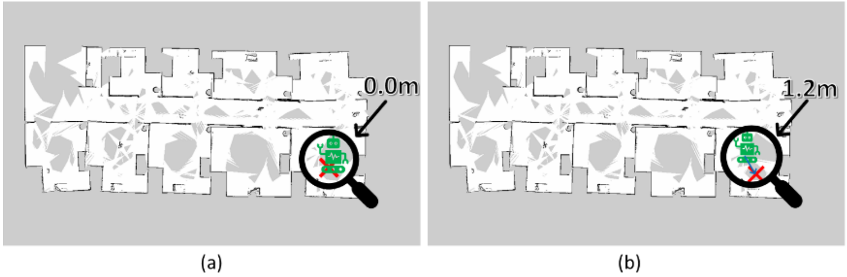


Figure 13. Examples of the mean error value for different paths. In this example, in the simple path (**a**), the distance between the robot and the target point is less than in the complex path (**b**).

Students were asked to perform the set of simulations in each scenario during the class session. At the beginning of the activity, they were given the datasheet shown in Table 2 for recording and further comparison of their simulation results.

Table 2. Example of sheet report for results obtained.

Scenario:____________________						
Path:	Path 1:__________			Path 2:__________		
Parameters:	Distance	Time	Error	Distance	Time	Error
Algorithm 1	______	______	______	______	______	______
Algorithm 2	______	______	______	______	______	______

The teacher and students can perform a detailed analysis and data comparison in each scenario of the autonomous driven algorithms. For this analysis, *t*-tests, a statistical study comparing two means of two independent groups of data, are recommended. This test determines if there is a statistically significant difference between the two means of both groups when the *p*-value is equal to or less than 0.05.

Table 3 summarizes the data obtained after the simulations were run in each scenario on both paths. These results depend on the initial and final position from the trajectory; in this case, these positions were the same for all the tests. It illustrates the average speed, calculated using the time and distance traveled, the time it takes the vehicle to finish the route, and the error with respect to the target point.

Figure 14 shows the comparison between the results of the written exams of phase 1 (traditional method) and phase 2 (ROS and simulations). These results show that students obtained a higher grade in phase 2 when the professor used the proposed platform and the simulations in class.

Table 3. Summary of simulation data obtained.

Scenarios	Hospital			Warehouse			Laboratory		
Parameters	**Speed (m/s)**	**Time (s)**	**Error (m)**	**Speed (m/s)**	**Time (s)**	**Error (m)**	**Speed (m/s)**	**Time (s)**	**Error (m)**
Simple path TEBand	0.08	305.00	0.18	0.08	179.80	0.09	0.07	245.80	0.23
Simple path EBand	0.08	267.00	0.16	0.06	340.00	0.13	0.06	465.00	0.15
Complex path TEBand	0.10	648.00	1.03	N/A	N/A	N/A	0.10	238.00	0.22
Complex path EBand	0.13	436.00	1.19	0.15	143.68	0.89	0.07	377.00	0.12

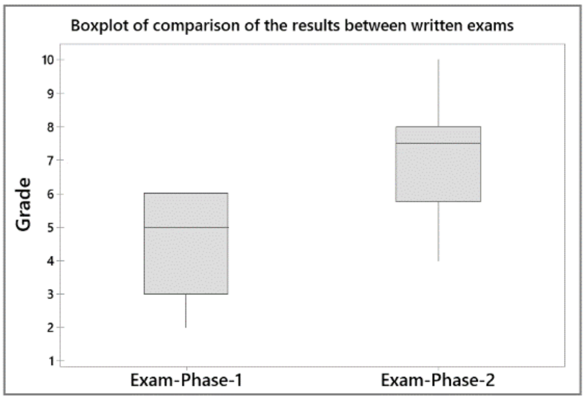


Figure 14. Comparison between the results of the written exams of phase 1 and 2.

In addition, at the end of both stages of this study, participants had to evaluate a set of affirmations in order to assess the usability of the platform using a 5-point Likert Scale. The affirmations are listed below:

A1. I think there is a difference in my learning when using ROS's simulations in class.
A2. The use of simulations helped me to better understand the theoretical concepts.
A3. The use of simulations helped me understand the advantages and disadvantages of each algorithm.
A4. The use of simulations motivated me to pay attention in class.
A5. The use of simulations made the class session more interesting.
A6. I think that the platform used to run the simulations is simple and easy to use.
A7. I feel more confident now that I have learned by using the simulations in class.
A8. I think that the simulations should always be used as a support tool in class.

Table 4 summarizes the results of the usability evaluation. In this case, a value of 1 means “totally agree”, while a value of 5 means “totally disagree”.

Table 4. Summary of the results of the usability evaluation.

5-Point Likert Scale	A1	A2	A3	A4	A5	A6	A7	A8
1	62%	77%	79%	63%	67%	23%	68%	73%
2	21%	7%	10%	15%	22%	21%	14%	15%
3	0%	11%	0%	11%	0%	35%	16%	0%
4	17%	5%	11%	11%	11%	3%	2%	12%
5	0%	0%	0%	0%	0%	18%	0%	0%

4. Discussion

Figure 15 shows the results from virtual scenario #1—the hospital floor in which the mobile robot using algorithm TEBand required longer periods of time to complete the trajectories, 14.23% and 48.62% slower than when using the EBand algorithm. In terms of accuracy, the EBand algorithm was 11.40% more accurate than the TEBand algorithm on the short path. However, the TEBand algorithm was 15.46% more accurate than the EBand algorithm on the long path. The statistical analysis demonstrates that, in the case of the simple path, there is no statistically significant difference between TEBand and EBand groups in terms of the time taken to complete the trajectory and their accuracy but not in the case of the complex path ($p < 0.05$).

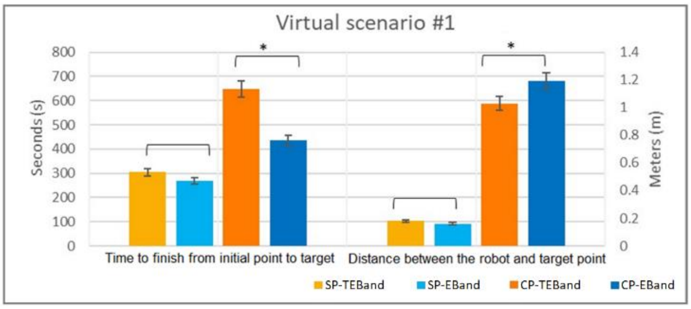


Figure 15. Comparison of time to finish and error among experimental groups using the hospital scenario (* $p < 0.05$).

Figure 16 shows the results from virtual scenario #2—the warehouse, where the robot using the EBand algorithm required longer times to cover the short route, 89.09% slower than the TEBand algorithm. In terms of accuracy, the TEBand algorithm was 36.56% more accurate than the EBand algorithm on the simple path. On the other hand, on the complex path, the TEBand algorithm could not complete the trajectories in this scenario. The statistical analysis demonstrates that in the case of the simple path there is a statistically significant difference ($p < 0.05$) between TEBand and EBand groups in terms of the time taken to complete the trajectory and their accuracy.

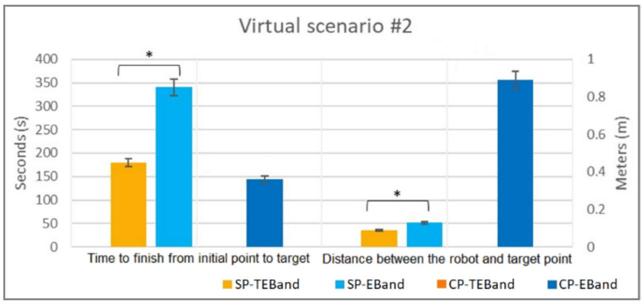


Figure 16. Comparison of time to finish and error among experimental groups using the warehouse scenario (* $p < 0.05$).

Figure 17 shows the results from virtual scenario #3—the industrial manufacturing area. The EBand algorithm took the longest times in both the short and long paths, 82.50% and 58.40% slower than the TEBand algorithm, respectively. In terms of accuracy, the EBand algorithm was 56.85% more accurate than the TEBand algorithm on the short path. Similarly, the Eband algorithm was 97.69% more accurate than the TEBand algorithm on the long path. The statistical analysis demonstrates that there is a statistically significant difference between all study groups ($p < 0.05$) in terms of the time taken to complete the trajectory and their accuracy.

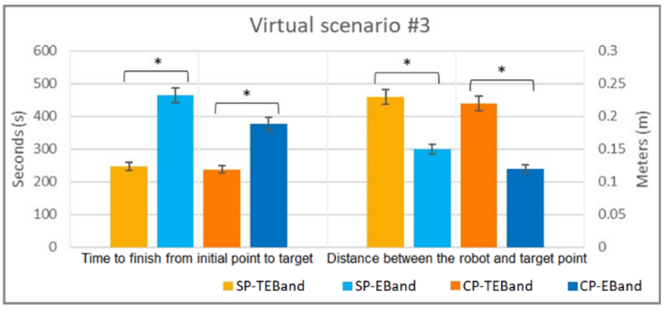


Figure 17. Comparison of time taken to finish and error among experimental groups in the industrial manufacturing scenario (* $p < 0.05$).

Regarding the performance of the students, Figure 14 shows that there is a difference in the results of the written exams. Moreover, the *t*-test demonstrates that there is a statistically significant difference ($p < 0.05$) between phase 1 and phase 2. In other words, in the case of phase 2 (when professor uses the simulations in class), this educational approach helps improve the performance of students in comparison to when professor uses a traditional teaching method. This fact demonstrates that the main objective of this work has been met, the proposed platform helps students in improving their academic performance.

Figure 18 shows that, in general, the use of simulations helps better understand the theoretical concepts, the advantages and disadvantages of each algorithm, makes the class session more interesting, motivates students to pay attention, and makes students feel more confident. More than 85% of the students suggest that this educational approach should always be used in class in order to improve their learning process. Nevertheless, some students think that the use of the platform is not simple and easy. This fact suggests that it is necessary to consider an initial session in which the teacher familiarizes the students with the use of the platform and its functions so that the tool is not distracting or does not impair the learning process.

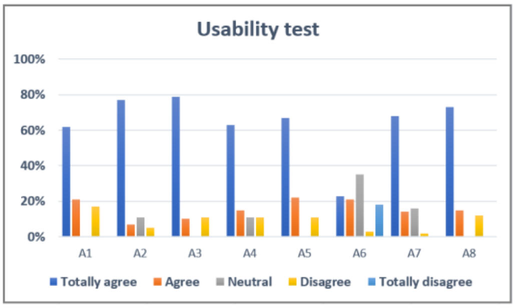


Figure 18. Comparison of the results of the usability test. In general, the use of simulations helps the learning process, but the professor should make sure that the students understand how to use the platform.

From the point of view of the professor and the students, this platform is a valuable tool for those who teach subjects related to the design and implementation of autonomous vehicles. In addition, freed from the responsibility of implementing the tools, the students can focus on the learning algorithms and autonomous driving techniques in order to analyze different variables that affect their operation.

An advantage of this project is that by using the ROS environment, students can perform different tests and analyze the performance of the algorithms to compare the data. Using the proposed platform, students can evaluate and compare the performance of different algorithms in different scenarios and different trajectories. The students have to set the trajectory (start point and target point), and at the end of the simulation, the platform provides the performance results of the algorithm (distance, time, and error). By comparing the performance values, students can identify the advantages and limitations of each algorithm in each scenario and, thus, select the one with the best performance to solve the given problem.

Similarly, students can implement different scenarios since it is only necessary to create them in Gazebo and perform the mapping using ROS libraries. In addition, different autonomous navigation algorithms can be used, which may be native to ROS, created by third parties, or user created. The main limitation of this study is that the platform was tested in two undergraduate robotic courses. More experiments would be recommended in order to identify limitations and improve the platform. Moreover, it is necessary to analyze how the proposed platform allows for the adaptation of different teaching methodologies and curricula. In addition, the platform should be improved to provide more realistic robot models, more measurements, and include more local route planning algorithms.

5. Conclusions

STEAM education is essential in engineering education for the development of competencies in undergraduate students. In the case of robotics, the research community has proposed different platforms for teaching; nevertheless, these platforms present challenges in terms of accessibility, cost, and flexibility for its use in the classroom. One complication of teaching autonomous driving is that students require access to the real robotic platform and free space to perform tests with the robot. This study describes a free platform to teach autonomous driving algorithms based on ROS software.

In order to exemplify the benefits that the proposed platform can provide in teaching, the authors conducted a case study using this platform with three different virtual scenarios in which two local route planning algorithms were evaluated and compared. In class, this platform was used to run simulations and to put theory into practice. Each algorithm was analyzed by using different variables such as total distance, time trajectory, and mean error. The students' quantitative and qualitative evaluations of the proposed platform show that this educational approach helps the students' learning process, reinforces theoretical knowledge, and motivates students in class. In addition, these results are a first step to demonstrating that the limitations of other platforms can be solved using the proposed platform in which students can compare the performance of different navigation algorithms under the same conditions.

Overall, the results claim that the use of ROS software allows students to perform autonomous driving simulations in a practical manner, without designing or implementing a physical robot. In addition, the platform facilitates the implementation of simulations and could help students focus on learning concepts and theory related to autonomous driving. The authors propose the continuation of this project as future work and suggest the design and implementation of a physical platform that students can build easily to verify simulation results in a real environment.

Author Contributions: Conceptualization, M.A.C.-R. and J.R.-A.; formal analysis, M.A.C.-R. and J.R.-A.; methodology, M.A.C.-R. and A.S.-D.; visualization, E.S.-T.; writing—original draft, M.A.C.-R. and J.R.-A.; writing—review and editing, A.S.-D., E.S.-T. and C.A.B.-C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors acknowledge the financial support of Writing Lab, Institute for the Future of Education, Tecnologico de Monterrey, Mexico, in the production of this work. Marco Antonio Chunab-Rodríguez acknowledges the Mexican National Council for Science and Technology (CONACYT) for the scholarship CONACYT CVU 876197 to carry out this research.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Benitti, F.B.V.; Spolaôr, N. How Have Robots Supported STEM Teaching? In *Robotics in STEM Education: Redesigning the Learning Experience*; Khine, M.S., Ed.; Springer International Publishing: Cham, Switzerland, 2017; pp. 103–129, ISBN 978-3-319-57786-9.
2. Eguchi, A. Bringing Robotics in Classrooms. In *Robotics in STEM Education: Redesigning the Learning Experience*; Khine, M.S., Ed.; Springer International Publishing: Cham, Switzerland, 2017; pp. 3–31. ISBN 978-3-319-57786-9.
3. Verner, I.M.; Cuperman, D.; Reitman, M. Exploring Robot Connectivity and Collaborative Sensing in a High-School Enrichment Program. *Robotics* **2021**, *10*, 13. [CrossRef]
4. Robles, D.; Quintero, C.G.M. Intelligent System for Interactive Teaching through Videogames. *Sustainability* **2020**, *12*, 3573. [CrossRef]
5. Karalekas, G.; Vologiannidis, S.; Kalomiros, J. EUROPA: A Case Study for Teaching Sensors, Data Acquisition and Robotics via a ROS-Based Educational Robot. *Sensors* **2020**, *20*, 2469. [CrossRef] [PubMed]
6. Martínez-Tenor, Á.; Cruz-Martín, A.; Fernández-Madrigal, J.-A. Teaching Machine Learning in Robotics Interactively: The Case of Reinforcement Learning with Lego® Mindstorms. *Interact. Learn. Environ.* **2019**, *27*, 293–306. [CrossRef]
7. Rosillo, N.; Montés, N.; Alves, J.P.; Ferreira, N.M.F. A Generalized Matlab/ROS/Robotic Platform Framework for Teaching Robotics. In Proceedings of the Robotics in Education, Vienna, Austria, 10–12 April 2019; Merdan, M., Lepuschitz, W., Koppensteiner, G., Balogh, R., Obdržálek, D., Eds.; Springer International Publishing: Cham, Switzerland, 2020; pp. 159–169.
8. Amsters, R.; Slaets, P. Turtlebot 3 as a Robotics Education Platform. In Proceedings of the Robotics in Education, Vienna, Austria, 10–12 April 2019; Merdan, M., Lepuschitz, W., Koppensteiner, G., Balogh, R., Obdržálek, D., Eds.; Springer International Publishing: Cham, Switzerland, 2020; pp. 170–181.
9. Kamińska, D.; Zwoliński, G.; Wiak, S.; Petkovska, L.; Cvetkovski, G.; Barba, P.D.; Mognaschi, M.E.; Haamer, R.E.; Anbarjafari, G. Virtual Reality-Based Training: Case Study in Mechatronics. *Technol. Knowl. Learn.* **2021**, *26*, 1043–1059. [CrossRef]
10. Agnihotri, A.; O'Kelly, M.; Mangharam, R.; Abbas, H. Teaching Autonomous Systems at 1/10th-Scale: Design of the F1/10 Racecar, Simulators and Curriculum. In Proceedings of the 51st ACM Technical Symposium on Computer Science Education, New York, NY, USA, 11–14 March 2020; Association for Computing Machinery: New York, NY, USA, 2020; pp. 657–663, ISBN 978-1-4503-6793-6.
11. Costa, V.; Rossetti, R.J.F.; Sousa, A. Autonomous Driving Simulator for Educational Purposes. In Proceedings of the 2016 11th Iberian Conference on Information Systems and Technologies (CISTI), Gran Canaria, Spain, 15–18 June 2016; pp. 1–5.
12. Kocić, J.; Jovičić, N.; Drndarević, V. Sensors and Sensor Fusion in Autonomous Vehicles. In Proceedings of the 2018 26th Telecommunications Forum (TELFOR), Belgrade, Serbia, 20–21 November 2018; pp. 420–425.
13. Levinson, J.; Askeland, J.; Becker, J.; Dolson, J.; Held, D.; Kammel, S.; Kolter, J.Z.; Langer, D.; Pink, O.; Pratt, V.; et al. Towards Fully Autonomous Driving: Systems and Algorithms. In Proceedings of the 2011 IEEE Intelligent Vehicles Symposium (IV), Baden-Baden, Germany, 5–9 June 2011; pp. 163–168.
14. Otto, S.; Schmitt, A.; Dücker, D.; Seifried, R. Teaching Vision-Based Control for Autonomous Driving with Lego Mindstorms EV3, Raspberry Pi and Simulink. *Proc. Appl. Math. Mech.* **2018**, *18*, e201800008. [CrossRef]
15. Naotunna, I.; Wongratanaphisan, T. Comparison of ROS Local Planners with Differential Drive Heavy Robotic System. In Proceedings of the 2020 International Conference on Advanced Mechatronic Systems (ICAMechS), Hanoi, Vietnam, 10–13 December 2020; pp. 1–6.
16. Pimentel, F.; Aquino, P. Performance Evaluation of ROS Local Trajectory Planning Algorithms to Social Navigation. In Proceedings of the 2019 Latin American Robotics Symposium (LARS), 2019 Brazilian Symposium on Robotics (SBR) and 2019 Workshop on Robotics in Education (WRE), Rio Grande, Brazil, 23–25 October 2019; pp. 156–161.
17. Fabregas, E.; Farias, G.; Dormido-Canto, S.; Guinaldo, M.; Sánchez, J.; Dormido Bencomo, S. Platform for Teaching Mobile Robotics. *J. Intell. Robot Syst.* **2016**, *81*, 131–143. [CrossRef]
18. Naya, M.; Varela, G.; Llamas, L.; Bautista, M.; Becerra, J.C.; Bellas, F.; Prieto, A.; Deibe, A.; Duro, R.J. A Versatile Robotic Platform for Educational Interaction. In Proceedings of the 2017 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), Bucharest, Romania, 21–23 September 2017; Volume 1, pp. 138–144.
19. Farias, G.; Fabregas, E.; Peralta, E.; Vargas, H.; Dormido-Canto, S.; Dormido, S. Development of an Easy-to-Use Multi-Agent Platform for Teaching Mobile Robotics. *IEEE Access* **2019**, *7*, 55885–55897. [CrossRef]
20. González-García, S.; Rodríguez-Arce, J.; Loreto-Gómez, G.; Montaño-Serrano, V.M. Designing a Teaching Guide for the Use of Simulations in Undergraduate Robotics Courses: A Pilot Study. *Int. J. Interact. Des. Manuf.* **2019**, *13*, 923–933. [CrossRef]
21. Manzoor, S.; Ul Islam, R.; Khalid, A.; Samad, A.; Iqbal, J. An Open-Source Multi-DOF Articulated Robotic Educational Platform for Autonomous Object Manipulation. *Robot. Comput.-Integr. Manuf.* **2014**, *30*, 351–362. [CrossRef]
22. Alers, S.; Hu, J. AdMoVeo: A Robotic Platform for Teaching Creative Programming to Designers. In *Learning by Playing. Game-Based Education System Design and Development, Proceedings of the International Conference on Technologies for E-Learning and Digital Entertainment, Banff, AB, Canada, 9–11 August 2009*; Chang, M., Kuo, R., Kinshuk, Chen, G.-D., Hirose, M., Eds.; Springer: Berlin/Heidelberg, Germany, 2009; pp. 410–421.
23. Plaza, P.; Sancristobal, E.; Carro, G.; Castro, M.; Blázquez, M.; Muñoz, J.; Álvarez, M. Scratch as Educational Tool to Introduce Robotics. In *Teaching and Learning in a Digital World, Proceedings of the International Conference on Interactive Collaborative Learning, Budapest, Hungary, 27–29 September 2017*; Auer, M.E., Guralnick, D., Simonics, I., Eds.; Springer International Publishing: Cham, Switzerland, 2018; pp. 3–14.

24. D'Ademo, N.; Lui, W.L.D.; Li, W.H.; Sekercioglu, A.; Drummond, T. EBug—An Open Robotics Platform for Teaching and Research. In Proceedings of the 2011 Australasian Conference on Robotics and Automation, Melbourne, Australia, 2–4 December 2011.
25. Vega, J.; Cañas, J.M. PiBot: An Open Low-Cost Robotic Platform with Camera for STEM Education. *Electronics* **2018**, *7*, 430. [CrossRef]
26. Pitt, J.N.; Strait, N.L.; Vayndorf, E.M.; Blue, B.W.; Tran, C.H.; Davis, B.E.M.; Huang, K.; Johnson, B.J.; Lim, K.M.; Liu, S.; et al. WormBot, an Open-Source Robotics Platform for Survival and Behavior Analysis in C. Elegans. *Geroscience* **2019**, *41*, 961–973. [CrossRef] [PubMed]
27. Farias, G.; Fabregas, E.; Peralta, E.; Torres, E.; Dormido, S. A Khepera IV Library for Robotic Control Education Using V-REP. *IFAC-PapersOnLine* **2017**, *50*, 9150–9155. [CrossRef]
28. Fairchild, C.; Harman, T.L. *ROS Robotics by Example*; Packt Publishing: Birmingham, UK, 2016; ISBN 978-1-78217-519-3.
29. Documentation—ROS Wiki. Available online: http://wiki.ros.org/ (accessed on 14 June 2022).
30. Gazebo. Available online: https://gazebosim.org/home (accessed on 14 June 2022).
31. Cañas, J.M.; Perdices, E.; García-Pérez, L.; Fernández-Conde, J. A ROS-Based Open Tool for Intelligent Robotics Education. *Appl. Sci.* **2020**, *10*, 7419. [CrossRef]
32. Niu, Y.; Qazi, H.; Liang, Y. Building a Flexible Mobile Robotics Teaching Toolkit by Extending MATLAB/Simulink with ROS and Gazebo. In Proceedings of the 2021 7th International Conference on Mechatronics and Robotics Engineering (ICMRE), Budapest, Hungary, 3–5 February 2021; pp. 10–16.
33. IFR World Robotics Report. 2016. Available online: https://ifr.org/ifr-press-releases/news/world-robotics-report-2016 (accessed on 14 June 2022).