engineering proceedings

MDPI

Proceeding Paper

Teaching Aspects of ROS 2 and Autonomous Vehicles †

Ernő Horváth [1,*], Gergő Ignéczi [2], Norbert Markó [1], Rudolf Krecht [1] and Miklós Unger [1]

[1] Vehicle Industry Research Center, Széchenyi István University, H-9026 Győr, Hungary; marko.norbert@ga.sze.hu (N.M.); krecht.rudolf@ga.sze.hu (R.K.); miklos.unger@ga.sze.hu (M.U.)
[2] Robert Bosch Engineering Center, Széchenyi István University, Gyömrői út 104, H-1103 Budapest, Hungary; gergo.igneczi@ga.sze.hu
* Correspondence: herno@ga.sze.hu
† Presented at the Sustainable Mobility and Transportation Symposium 2024, Győr, Hungary, 14–16 October 2024.

Abstract: The advancement of autonomous vehicles (AVs) has brought forth a substantial need for effective education in robotic operating systems, particularly ROS 2, which serves as the backbone for many autonomous vehicle (AV) applications. This paper explores the academic approach and instructional methodologies tailored for teaching ROS 2 in the context of autonomous vehicle technology. It highlights the critical components and architecture of ROS 2, emphasizing its modularity, real-time communication capabilities, and robust ecosystem which make it ideal for AV development. Through a detailed curriculum outline, we describe hands-on learning activities, simulation-based exercises, and project-driven modules that facilitate deep understanding and practical skills acquisition. The effectiveness of these teaching methods is evaluated through a mixed-methods study involving student feedback, performance assessments, and project outcomes. Results indicate significant improvements in student comprehension and proficiency in both ROS 2 and autonomous vehicle systems. This research contributes to the body of knowledge by providing a comprehensive framework for educators to effectively teach ROS 2, thereby fostering the next generation of engineers proficient in developing and deploying autonomous vehicle technologies.

Keywords: ROS2; robotics; self-driving

check for updates

Citation: Horváth, E.; Ignéczi, G.; Markó, N.; Krecht, R.; Unger, M. Teaching Aspects of ROS 2 and Autonomous Vehicles. *Eng. Proc.* **2024**, *79*, 49. https://doi.org/10.3390/engproc2024079049

Academic Editors: András Lajos Nagy, Boglárka Eisinger Balassa, László Lendvai and Szabolcs Kocsis-Szürke

Published: 6 November 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The Digital Education Action Plan is a renewed European Union (EU) policy initiative that also prioritizes STEM education (science, technology, engineering, and mathematics) [1]. Autonomous vehicles and robotics which are emerging STEM fields are int the focus of our research group with ROS 2, which serves as the backbone for these systems.

The learning curve of ROS 2 (Robot Operating System 2) is considered steep due to several interrelated factors that collectively create a challenging environment for newcomers [1–6]. Firstly, ROS 2 is a complex system that integrates a multitude of tools, libraries, and conventions designed to facilitate the creation of sophisticated and robust robot behaviors. This complexity, while providing powerful capabilities, can be overwhelming for beginners who must grasp various components such as nodes, topics, services, actions, and parameters. Moreover, ROS 2 operates on a distributed system architecture where multiple nodes communicate over a network. Understanding this architecture, including the setup and management of inter-node communication, requires knowledge of network programming and concurrent computing, both of which are advanced topics that can be daunting for novices. Additionally, ROS 2 uses the Data Distribution Service (DDS) as its default middleware, which adds another layer of abstraction. DDS introduces concepts like Quality of Service (QoS) settings that must be understood and properly configured, further complicating the learning process.

The real-time requirements of many ROS 2 applications, especially in fields like autonomous vehicles, add an additional layer of complexity. Achieving real-time performance

necessitates an understanding of issues related to latency, jitter, and determinism, which are typically addressed in real-time systems. Beginners may find it difficult to determine which components to use and how to assemble them into a functional autonomous vehicle or robotic system. Additionally, the educational resources available for ROS 2, though growing, are still limited compared to more established technologies. Finally, the effective learning of ROS 2 often necessitates hands-on experience with physical or simulated robots. Setting up and configuring the necessary hardware and software environments can be technically demanding, requiring a good understanding of robotics hardware, sensors, and actuators. This practical aspect of learning adds to the overall complexity, making the learning curve of ROS 2 steep and challenging for many newcomers.

Considering the aforementioned points, our team decided to develop a uniquely tailored ROS 2 course specifically focused on autonomous vehicles and robotics. The course encompasses C++ and Python based ROS 2 programming. We have open-sourced this course and made it available to anyone interested in learning.

2. Related Courseware

There is a diverse range of ROS 2 courseware available, each tailored to different aspects of robotics. Some courses concentrate on aerial robotics and flying drones, while others specialize in robotic arms or communication systems. Our specific focus, however, is on ground robotics and autonomous vehicles. To address this, we developed our own courseware designed to focus on these areas. Additionally, where licensing permitted, we incorporated elements from existing courseware to enhance our curriculum and ensure comprehensive coverage of relevant topics. Table 1 summarizes the educational resources that were cited or shaped our syllabus.

Table 1. Course materials that influenced or were referenced in our syllabus.

University/Institution	Department/Project	People	License
MIT (Massachusetts Institute of Technology)	RACECAR project, F1/10 car model for Gazebo	Assoc. Prof. Dr. Tufan Kumbasar et al.	MIT license
University of Virginia	F1/10 Crew	Dr. Madhur Behl and Varundev Suresh Babu	GPL-3.0 license
ETH Zürich	Programming for Robotics—ROS	Péter Fankhauser, Dominic Jud, Martin Wermelinger, Prof. Dr. Marco Hutter	N/A
TU München (Technical University of Munich)	Institute of Automotive Technology, "Autonomous Driving Software Engineering"	Prof. Dr.-Ing. Lienkamp and Phillip Karle	GPL-3.0 license
TH Ingolstadt	Autonomous Vehicle Engineering Faculty of Electrical Engineering and Computer Science	Prof. Dr. Martin Ebert et al.	N/A
Stanford University	Introduction to Robotics	Prof. Oussama Khatib et al.	N/A
Stanford University	CS231n: Convolutional Neural Networks for Visual Recognition	Andrej Karpathy	MIT license
Óbuda University	Antal Bejczy Center for Intelligent Robotics	Tamás D. Nagy and Péter Galambos	CC BY-NC-SA 4.0
Autoware Foundation	Software and documentation	N/A	Apache License 2.0
Antonio Mauro Galiano	ros2_pid_library	N/A	MIT license
Budapest University of Technology	Control theory	József Bokor and Péter Gáspár	CC BY-NC-ND 3.0

3. The Developed Syllabus and Hands-On Learning

The syllabus (Table 2) is structured to guide students progressively through key concepts and practical applications of ROS 2, starting with foundational knowledge and advancing toward complex system integrations like perception, simulation, and real-time control. Each module is designed to offer a balanced blend of lectures, hands-on exercises, and project-based learning to ensure that students not only comprehend the theoretical aspects of ROS 2 but also gain substantial practical experience.

Table 2. Description of our syllabus.

Module	Description	Tools/Software	Guidance
1. Introduction	Introduction to autonomous vehicles and robots (without ROS 2)	Terminal, VS Code, Foxglove Studio	Introduce the typical sensory data, visualize it with Foxglove studio and get familiar with the terminal.
2. ROS 2 Concepts and Install	Overview of ROS 2 architecture, installation, and configuration.	ROS 2 Humble	Step-by-step installation guide on Linux/Windows, troubleshooting, and verifying setup via ROS 2 CLI commands.
3. Sensing	Integration of sensors into ROS 2 for AV applications.	ROS 2 drivers, LIDAR, camera, GPS.	Guide students through sensor integration, subscribing to data, and visualizing sensor inputs in rviz2/Foxglove.
4. ROS 2 Advanced	Advanced ROS 2 topics including QoS, node lifecycle, and multi-threading.	DDS for QoS, rclcpp and rclpy for node management.	Teach how to adjust QoS settings for performance in unreliable network conditions, focusing on real-time use cases.
5. Transformations	Coordinate transformations in AVs using TF2.	tf2_ros library, rviz2 for visualization.	Explain the importance of transformations for sensor alignment and robot navigation, with practical transformation examples.
6. Perception	Object detection and environment perception in AVs using ROS 2.	OpenCV, PCL libraries, machine learning models	Guide students through using perception libraries, applying them to real-world AV tasks like obstacle detection.
7. Simulation	Use of simulators like Gazebo to test AV algorithms.	Gazebo (Ignition)	Teach students how to create and customize virtual environments, focusing on AV simulation in Gazebo.
8. Planning	Path planning for AVs using navigation algorithms.	ROS 2 Nav2, A*, RRT.	Show students how to configure planners and controllers, adapting them to specific AV tasks and real-time environments.
9. Control	Vehicle control (low-level actuators, high-level path following) in ROS 2.	ROS 2 Control package, PID controllers.	Teach students how to tune control parameters for different robot types, ensuring smooth and accurate path following in Gazebo simulation.
10. AI	Introduction to AI methods in AVs for decision-making and perception.	PyTorch integrated with ROS 2.	Provide step-by-step guidance on setting up AI models and integrating them with ROS 2 for enhanced AV performance.
11. Safety, V&V	Safety aspects, verification, and validation of AV systems.	ROS 2 testing frameworks	Guide students in using formal verification methods and testing frameworks to ensure system safety and reliability.

4. Selected Aspects of the Syllabus and Experiments

The syllabus our team constructed follows the patterns of the popular course structures mentioned in the previous chapter. Although our special focus and previous teaching experience slightly altered these patterns. In the following a few notable examples will be introduced.

According to our team's previous experience there are some parts of robotics which engineers and computer science student understand very differently. One of the simplest examples of the mentioned phenomenon are the kinematic equations for vehicle motion in 2 dimensions (see Equation (1)). These equations describe the motion of the autonomous vehicle or robot, in terms of its position and velocity. The equation describes the connection between (x,y), which is the position of the vehicle, θ, which is the heading angle, v, which is the linear velocity, ϕ, which is the steering angle, and L, which is the wheelbase of the vehicle.

$$\begin{aligned} \dot{x} &= v\cos(\theta) \\ \dot{y} &= v\sin(\theta) \\ \dot{\theta} &= \tfrac{v}{L}\tan(\phi) \end{aligned} \tag{1}$$

For an engineer, developing code from this equation is relatively straightforward; however, it presents challenges for certain students. Figure 1 displays the same equation, slightly modified in discreet domain and introduced with the Δt sampling time. From this form it is much easier to create the corresponding C++ and Python code. Although it may seem like a minimal didactic adjustment, this approach significantly enhances student understanding.

$$\begin{aligned} x' &= x + v\cos(\theta)\Delta t \\ y' &= y + v\sin(\theta)\Delta t \\ \theta' &= \theta + v/L\tan(\phi)\Delta t \end{aligned}$$

(**a**)

```
x+=v*math.cos(theta)*dt
y+=v*math.sin(theta)*dt
theta+=(v/L)*math.tan(phi)*dt
```

(**b**)

```
x+=v*std::cos(theta)*dt;
y+=v*std::sin(theta)*dt;
theta+=(v/L)*std::tan(phi)*dt;
```

(**c**)

Figure 1. Motion equations (**a**) in the discrete time domain and the corresponding ROS 2 Python (**b**) and ROS 2 C++ (**c**) code snippets.

Another altered pattern of the compared to the popular course structure is that they tend to use own-developed robots [1,3,5,6]. Our approach was in that situation rather buy than build. Our approach is to use simulator as much as possible and at the end of the semester, students who are persistent enough can use a real robot. In the syllabus Ignition Gazebo, now known simply as Gazebo is used. Gazebo is a powerful simulation tool widely used in ROS 2 teaching. This simulator is quite lightweight our course is designed in a way that students could run simulations even without a powerful GPU. In this course 4 standard F1/10 Traxxas Robot [7–9] and 12 Wheeltec Ackermann Robot is used. At the end of the semester the simulation-based work could be continued on a real-world robot.

The mentioned robots have roughly the same dimensions, similar sensors (Figure 2), but a slightly different mechanical build. Wheeltec could be considered as a low-cost version of the F1/10 car, here most of the algorithms and teaching materials apply. With this approach we not only got strong teaching materials [10–12] for F1/10 car, but also, we could adopt it for the Wheeltec car too.

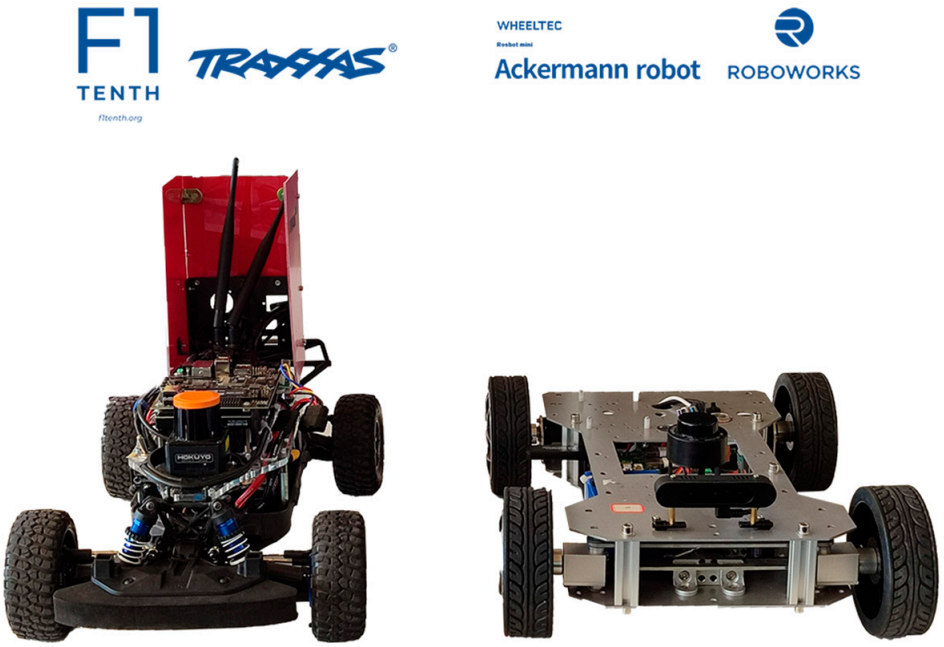


Figure 2. Side-by-side scale comparison of F1/10 Ackermann robot and Wheeltec Roboworks Rosbot-mini Ackermann robot.

5. Discussion

The rise of autonomous vehicles (AVs) has generated a strong demand for effective education in robotic operating systems, particularly ROS 2, which plays a crucial role in numerous AV applications. This paper examines academic approaches and instructional methodologies for teaching ROS 2 in the context of AV technology. It highlights ROS 2's critical components, modularity, real-time communication capabilities, and robust ecosystem, making it ideal for AV development. The paper outlines a part of the curriculum featuring hands-on activities, and project-driven modules to enhance understanding and practical skills. The effectiveness of these teaching methods is assessed through a mixed-methods study involving student feedback, performance assessments, and project outcomes, showing substantial improvements in student comprehension and proficiency. This research offers a comprehensive framework for educators to teach ROS 2 effectively, preparing the next generation of engineers for AV technology development and deployment.

Our team has accumulated one semester of experience with the Hungarian version of the course. The English version is nearing completion. Both versions are available online and are open source.

Author Contributions: Conceptualization, E.H. and G.I.; methodology, R.K.; software, M.U. and G.I.; resources, G.I., E.H. and N.M.; data curation, R.K.; writing—original draft preparation, E.H. All authors have read and agreed to the published version of the manuscript.

Funding: The research was supported by the European Union within the framework of the National Laboratory for Autonomous Systems. (RRF-2.3.1-21-2022-00002).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Source code and data available on https://sze-info.github.io/avr (accessed on 19 July 2024).

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

References

1. Karalekas, G.; Vologiannidi, 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]
2. 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]
3. Roldán-Álvarez, D.; Cañas, J.M.; Valladares, D.; Arias-Perez, P.; Mahna, S. Unibotics: Open ROS-based online framework for practical learning of robotics in higher education. *Multimed. Tools Appl.* **2024**, *83*, 52841–52866. [CrossRef]
4. Karaca, M.; Yayan, U. ROS based visual programming tool for mobile robot education and applications. *arXiv* **2020**, arXiv:2011.13706.
5. Araújo, A.; Portugal, D.; Couceiro, M.S.; Rocha, R.P. Integrating Arduino-based educational mobile robots in ROS. *J. Intell. Robot. Syst.* **2015**, *77*, 281–298. [CrossRef]
6. Raudmäe, R.; Schumann, S.; Vunder, V.; Oidekivi, M.; Nigol, M.K.; Valner, R.; Masnavi, H.; Singh, A.K.; Aabloo, A.; Kruusamäe, K. ROBOTONT–Open-source and ROS-supported omnidirectional mobile robot for education and research. *HardwareX* **2023**, *14*, e00436. [CrossRef] [PubMed]
7. O'Kelly, M.; Sukhil, V.; Abbas, H.; Harkins, J.; Kao, C.; Pant, Y.V.; Bertogna, M. F1/10: An open-source autonomous cyber-physical platform. *arXiv* **2019**, arXiv:1901.08567.
8. Tătulea-Codrean, A.; Mariani, T.; Engell, S. Design and simulation of a machine-learning and model predictive control approach to autonomous race driving for the f1/10 platform. *IFAC-PapersOnLine* **2020**, *53*, 6031–6036. [CrossRef]
9. 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, Portland, OR, USA, 11–14 March 2020; pp. 657–663.
10. Betz, J.; Zheng, H.; Liniger, A.; Rosolia, U.; Karle, P.; Behl, M.; Mangharam, R. Autonomous vehicles on the edge: A survey on autonomous vehicle racing. *IEEE Open J. Intell. Transp. Syst.* **2022**, *3*, 458–488. [CrossRef]
11. Evans, B.D.; Betz, J.; Zheng, H.; Engelbrecht, H.A.; Mangharam, R.; Jordaan, H.W. Bypassing the Simulation-to-reality Gap: Online Reinforcement Learning using a Supervisor. In Proceedings of the 2023 21st International Conference on Advanced Robotics (ICAR), Abu Dhabi, United Arab Emirates, 5–8 December 2023; pp. 325–331.
12. Evans, B.D.; Betz, J.; Zheng, H.A.; Engelbrecht, H.A.; Mangharam, R.; Jordaan, H.W. High-performance Racing on Unmapped Tracks using Local Maps. *arXiv* **2024**, arXiv:2401.17732.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.