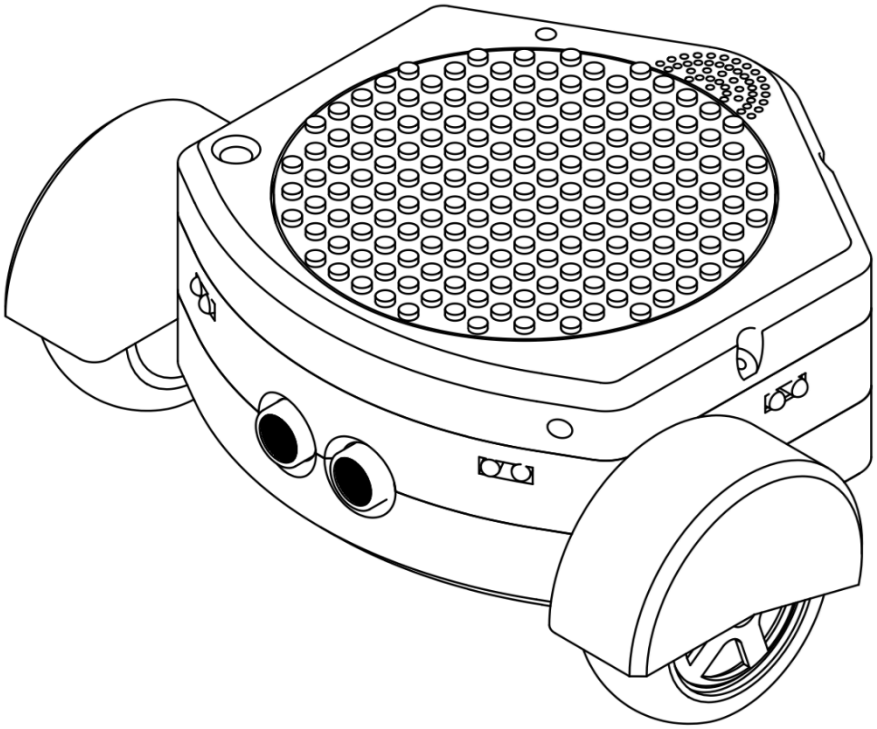
S.T.E.P.S.
Robotics and Education

Design and Development of S.T.E.P.S., the New Generation of Open Source Educational Robot

July 2025

Co-funded by the European Union



Co-funded by
the European Union

Disclaimer

Funded by the European Union. Views and opinions expressed are, however, those of the authors only and do not necessarily reflect those of the European Union or the Agence Erasmus+ France / Education Formation. Neither the European Union nor the Agence Erasmus+ France / Education Formation can be held responsible for them.

Information

Project	STEM Teaching and Education Platform for Students – S.T.E.P.S.
Project N°	2023-1-FR01-KA220-HED-000165711
Work Package / Project Result number	Work package n°2 - S.T.E.P.S. educational robot : Design and development of the open source educational robot / PR2
Date	July 2025
Type of Document	Report
Language	English

Consortium

Co-funded by the European Union

S.T.E.P.S.
Robotics and Education

Index

1. Executive Summary 6

1.1. Overview of the S.T.E.P.S. project 6

1.2. Objectives of the New Design 7

1.3. Key Innovations and Improvements 8

1.4. Target Audience and Use Cases 10

2. Background and Inspiration 13

2.1. Overview of FOSSBot: The Open-Source Educational Robot 13

2.2. Limitations and Challenges of FOSSBot 15

2.3. Desk Research Insights from 5 Countries 16

2.4. Initial Aims: Simplification and Usability 18

3. Design Philosophy and Goals 21

3.1. Core Design Principles 21

3.2. Focus on Open-Source and Accessibility 21

3.3. Simplification of Assembly and Usage 22

3.4. Modularity and Customizability 23

3.5. Alignment with Educational Standards 24

4. Development Process 25

4.1. Conceptualization and Ideation 25

4.2. Iterative Design and Prototyping 25

4.3. Incorporating Feedback from Desk Research 26

4.4. Collaboration with Open-Source Communities 26

4.5. Challenges and Solutions 27

5. Technical Design of the new FOSSBot 28

5.1. Mechanical Design 28

5.2. Electronics Design 31

5.3 Software Architecture and Connectivity 33

6. Key Features and Innovations 35

6.1. Simplified Assembly Process 35

6.2. Enhanced Durability and Usability 36

6.3. Support for Multiple Programming Levels 36

6.4. Expandable and Customizable Modules 36

Co-funded by
the European Union

6.5. Cost-Effective and Open-Source Design 37

7. Comparison with FOSSBot 37

7.1. Design and Structural Improvements 37

7.2. Enhanced Educational Features 38

7.3. Usability and Accessibility Comparison 38

7.4. Cost and Manufacturing Efficiency 38

8. Conclusion 40

8.1. Summary of Achievements 40

8.2. Impact on Educational Robotics 40

8.3. Call to Action for Educators and Developers 41

References 42

1. Executive Summary

1.1. Overview of the S.T.E.P.S. project

S.T.E.P.S. (STEM Teaching and Education Platform for Students) is a project designed to democratize access to robotics and STEM education, utilizing open-source technologies and free software to make educational resources more accessible and customizable for students and educators from various backgrounds. Leveraging open-source solutions and tools, as well as the best practices of STEM education across several countries, it aims to make robotics and programming accessible to students and their educators.

At its core, S.T.E.P.S. provides **low-cost, customizable educational robots** that empower students and educators to explore robotics and programming in a hands-on manner. By leveraging open-source principles, the project ensures that its tools and resources are freely available, enabling widespread adoption even in underserved communities. Complementing the hardware, S.T.E.P.S. includes manuals, guidelines, sample programs, etc., offering free access to high-quality STEM learning resources, tutorials, and project ideas. This combination of physical and digital tools creates a comprehensive ecosystem for learning and innovation.

The project brings together experts from diverse fields, including robotics, education, software development, and sustainability, to collaboratively design tools, guidelines, and educational content tailored to a wide range of learners. By addressing Erasmus+ priorities, S.T.E.P.S. promotes digital transformation, inclusion, and diversity in education, ensuring that no one is left behind in the rapidly evolving digital landscape. The initiative is designed to support pre- and in-service secondary school teachers or trainers who aim to teach STEM and robotics to beginners, as well as students from all backgrounds, particularly those in secondary schools and beyond, who are eager to learn robotics and programming. Additionally, S.T.E.P.S. engages educational institutions, authorities, and policymakers to foster systemic change, while also involving parents or guardians who wish to support their children's STEM education.

The project also welcomes community members, including local businesses and organizations, to contribute to STEM education initiatives in their areas, and it actively collaborates with open-source communities and developers to enhance the educational robot and MOOC platform. Furthermore, S.T.E.P.S. encourages primary school teachers to integrate STEM into their curriculum and invites university students or recent graduates to volunteer as mentors or instructors, creating a vibrant ecosystem of learning and collaboration. To align with global sustainability goals, the project incorporates green practices, such as using sustainable materials in robot production and adopting energy-efficient design principles. Through these efforts, S.T.E.P.S. not only advances STEM education but also embodies a forward-thinking, socially responsible approach that benefits learners, educators, and communities alike.

Co-funded by
the European Union

After performing an overview of open source software and hardware solutions and based on the previous experience of the team on designing educational robots, we decided to rely on the open source robot FOSSBot, and instead of designing a new robot from scratch, we decided to upgrade it.

1.2. Objectives of the New Design

The design of S.T.E.P.S. is driven by clear, measurable objectives aimed at making STEM education more accessible, engaging, and effective. These objectives focus on simplifying assembly and usage, enhancing educational value, promoting open-source collaboration, and ensuring affordability. Each objective is supported by specific design improvements and key performance indicators (KPIs) to track progress and success.

1. Simplify Assembly and Usage

One of the primary goals of S.T.E.P.S. is to reduce the complexity of building and operating the robot, making it accessible to beginners, including students and educators with limited technical expertise. To achieve this, the robot's body and electronic components have been redesigned to allow for an **easier assembly process**. This includes modular components that snap together without the need for specialized tools, as well as clear, step-by-step instructions. Additionally, the project aims to **reduce 3D printing time and material usage** by optimizing the design for efficiency. Quantitative KPIs, such as the **percentage reduction in printing materials and time**, will be tracked to measure success. User feedback on the ease of assembly will also be collected through the LTTA workshops, ensuring continuous improvement based on real-world experiences.

2. Enhance Educational Value

S.T.E.P.S. is designed to provide a **versatile platform that supports multiple programming levels**, catering to users with varying skill levels. Beginners can start with the kindergarten mode which contains **only 4 movement buttons** and continue with **block-based coding** (using Blockly). The user interface (UI) is intuitive and adaptable, ensuring a seamless learning experience. To measure the success of this objective we will gather qualitative feedback through **online surveys** on the usability of the UI and its educational value. Additionally, the robot will feature **new sensors**, to expand its capabilities and applications.

3. Promote Open-Source Collaboration

S.T.E.P.S. is committed to fostering a **vibrant open-source community** where users can contribute to the development and improvement of the robot. By extending FOSSBot and generating FOSSBot 2.0, we will be able to encourage collaboration among educators, students, developers, and hobbyists. To further assist users who do not have the ability to print and assemble the robot immediately, we also provide a simulated version of the robot developed using CoppeliaSim. The digital twin allows users to simulate and test

Co-funded by
the European Union

the robot in a virtual environment, further lowering barriers to entry. The simulated version and all design files, software, and documentation of the new robot are **freely available through the FOSSBot GitHub page** (https://github.com/eellak/fossbot) Qualitative feedback from the community will be collected to assess the relevance and impact of the open-source approach.

4. Ensure Affordability

To enable **widespread adoption**, particularly in schools and underserved communities, the new FOSSBot is designed to be **cost-effective**. The optimized design reduces material and production costs, while the use of 3D printing allows for local manufacturing, further lowering expenses. Affordability will be measured through quantitative KPIs, such as the **percentage reduction in material costs**, and qualitative feedback from educators and institutions on the robot's accessibility.

The success of FOSSBot redesign will be evaluated through a combination of **quantitative and qualitative KPIs**. Quantitative metrics include the **percentage reduction in printing materials and time**, the **number of new sensors and expansion capabilities**, the **number of programming tools supported by the UI**, and the **number of downloads and users of the digital twin**. Qualitative KPIs will focus on user feedback, collected through **online surveys** and assessing the ease of assembly, usability of the UI, educational value, and relevance to curricula. These metrics will ensure that the project remains aligned with its goals and continuously improves based on user needs.

By focusing on these objectives and tracking progress through well-defined KPIs, S.T.E.P.S. aims to create a transformative educational tool that empowers learners, supports educators, and fosters innovation in STEM education.

1.3. Key Innovations and Improvements

S.T.E.P.S. introduces a range of **key innovations and improvements** over the original FOSSBot, now referred to as **FOSSBot 2.0**. These advancements are designed to enhance functionality, usability, and accessibility, making the robot a more versatile and effective tool for STEM education. Below are the major innovations that define FOSSBot 2.0:

1. Integration of Raspberry Pi

One of the most significant upgrades in FOSSBot 2.0 is the **integration of a Raspberry Pi** as the main processing unit. This powerful microcontroller not only supports a wide range of sensors and peripherals but also allows the robot to function as a **web server**, hosting its own programming platform. This means users can program the robot directly through a web interface, eliminating the need for additional software installations. The Raspberry Pi's versatility opens up new possibilities for advanced projects, such as data

Co-funded by
the European Union

logging, machine learning, and IoT applications, making FOSSBot 2.0 a future-proof educational tool.

2. Wireless Connectivity

FOSSBot 2.0 introduces **Wi-Fi connectivity**, enabling users to connect to the robot wirelessly. This eliminates the need for cables or USB connections to transfer programs, making the robot more convenient and user-friendly. Wireless connectivity also allows for real-time control and monitoring, enhancing the interactive learning experience. Students can program and control the robot from their laptops, tablets, or smartphones, fostering a more flexible and engaging learning environment.

3. New PCB Design

The robot features a **new printed circuit board (PCB)** that integrates multiple circuits, significantly reducing the number of cables and connections required. This streamlined design not only simplifies assembly but also improves reliability by minimizing potential points of failure. The new PCB also includes **JST clips** for easy sensor attachment and replacement, making it easier for users to customize and upgrade the robot.

4. Modular Plastic Design

FOSSBot 2.0 incorporates a **modular design for its 3D-printed plastic components**, which minimizes assembly time and simplifies maintenance. The modular approach allows users to easily replace or upgrade individual parts without disassembling the entire robot. Additionally, the careful placement of sensors and cables reduces the overall size of the robot and optimizes printing time, making it more efficient to produce.

5. Simulation in CoppeliaSim

To lower the barrier to entry, FOSSBot 2.0 includes a **digital twin simulation in CoppeliaSim**. This allows users to program and test the robot in a virtual environment without the need for physical hardware. The simulation is particularly useful for educators and students who may not have immediate access to the robot, providing a no-cost way to explore its programming philosophy and capabilities.

6. LEGO-Compatible Top

The robot's top panel is designed to be **LEGO-compatible**, enabling users to attach additional components and accessories. This feature encourages creativity and experimentation, allowing students to build custom extensions and integrate the robot with other LEGO-based projects. The top panel is also **attached with magnets**, making it easy to open and access the internal components for maintenance or upgrades.

7. Battery Compatibility and Accessibility

FOSSBot 2.0 supports **rechargeable lithium batteries** as well as other commonly available battery types, ensuring flexibility and accessibility for users with different resources. The battery compartment features a **removable bottom cover**, making it easy to replace batteries without disassembling the robot. This design choice enhances

Co-funded by the European Union

usability and ensures the robot can be powered by whatever batteries are readily available.

8. Robust and Durable Design

The robot's **sturdy and durable design** ensures it can withstand the rigors of classroom use. The materials and construction are optimized for longevity, reducing the need for frequent repairs or replacements. This durability makes FOSSBot 2.0 a reliable tool for educators and students alike.

9. Additional Sensors and Expandability

FOSSBot 2.0 includes **more sensors** than its predecessor, expanding its capabilities for educational projects. The robot also features **I2C connectivity**, allowing for the easy integration of additional components and peripherals. This expandability ensures that the robot can grow with the user's skills and interests, supporting more advanced projects as learners progress.

10. Improved Software Connectivity

The software architecture of FOSSBot 2.0 has been enhanced to improve **connectivity and compatibility** with various programming tools and platforms. This ensures a seamless experience for users, whether they are working with block-based coding environments like Blockly or text-based languages like Python and JavaScript.

11. Compact Size

The overall size of FOSSBot 2.0 has been reduced, making it more portable and easier to handle. Despite its smaller footprint, the robot retains all its functionality and expandability, ensuring it remains a powerful tool for STEM education.

These innovations collectively make FOSSBot 2.0 a **cutting-edge educational robot** that is easier to use, more versatile, and more accessible than ever before. By addressing the limitations of the original FOSSBot and incorporating user feedback, S.T.E.P.S. has created a tool that empowers learners and educators to explore the exciting world of robotics and programming.

1.4. Target Audience and Use Cases

S.T.E.P.S. is designed to cater to a diverse audience, ensuring that its benefits reach a wide range of users across educational and community settings. The robot and its accompanying resources are tailored to meet the needs of various target groups, each with unique goals and requirements.

1. Educators and Trainers

Pre- and in-service secondary school teachers or trainers form a core part of S.T.E.P.S.'s target audience. These educators are often tasked with introducing STEM and robotics

Co-funded by
the European Union

to students at a beginner level but may lack the tools or confidence to do so effectively. S.T.E.P.S. addresses this challenge by providing an easy-to-assemble robot, comprehensive lesson plans, and a user-friendly programming interface. Teachers can use the robot to teach fundamental concepts in robotics, programming, and engineering, while also fostering problem-solving and critical thinking skills. Additionally, primary school teachers who wish to incorporate STEM education into their curriculum can leverage S.T.E.P.S. to introduce younger students to basic robotics concepts in an engaging and age-appropriate manner.

2. Students and Learners

Students from all backgrounds, particularly those in secondary schools and beyond, are a key focus of S.T.E.P.S. The robot is designed to be accessible to beginners, allowing students with no prior experience in robotics or programming to get started quickly. At the same time, its advanced features and expandable capabilities provide opportunities for more experienced learners to explore complex concepts and projects. By supporting multiple programming levels—from block-based coding for beginners to text-based programming for advanced users—S.T.E.P.S. ensures that students can progress at their own pace. The robot's hands-on nature makes it an ideal tool for project-based learning, enabling students to apply theoretical knowledge to real-world challenges.

3. Institutions, Policymakers, and Community Stakeholders

Educational institutions and authorities play a crucial role in the adoption and implementation of STEM education initiatives. S.T.E.P.S. provides these stakeholders with a cost-effective, scalable solution that aligns with curriculum standards and promotes digital transformation. Policymakers can use the project as a model for integrating robotics and programming into national or regional education frameworks. Beyond formal education, community members, including local businesses and organizations, can support STEM education initiatives by sponsoring S.T.E.P.S. kits or hosting workshops. This collaborative approach helps build a strong ecosystem of learning and innovation at the community level.

4. Open-Source Contributors and Mentors

S.T.E.P.S. actively engages open-source communities and developers to contribute to the ongoing development and improvement of the robot and its accompanying MOOC platform. By making all design files, software, and documentation freely available, the project encourages collaboration and innovation. Additionally, university students or recent graduates can volunteer as mentors or instructors, sharing their expertise with younger learners and gaining valuable teaching experience. This not only enriches the learning experience for students but also fosters a sense of community and shared purpose among contributors.

Co-funded by
the European Union

Through its inclusive design and versatile applications, S.T.E.P.S. serves as a powerful tool for empowering learners, supporting educators, and engaging communities in the exciting world of STEM and robotics.

Co-funded by
the European Union

2. Background and Inspiration

2.1. Overview of FOSSBot: The Open-Source Educational Robot

FOSSBot is an educational robot that uses open software and hardware and can be employed at all levels of education. FOSSBot has been developed collaboratively by Harokopio University of Athens and GFOSS, and is an extension of "GSOC 2019 - A DIY robot kit for educators" https://github.com/eellak/gsoc2019-diyrobot. As a DIY (Do It Yourself) robot, FOSSBot is designed to be easily built, disassembled, and reassembled by users, making the construction process itself an integral part of the educational experience. The robot's components are intentionally simple and affordable, with electronic parts that are commercially available at low cost and 3D-printable plastic parts, ensuring accessibility for schools, educators, and hobbyists.

Figure 1. The top-view and internal aspect of FOSSBot's initial design

FOSSBot is equipped with a variety of sensors and features that make it a powerful tool for teaching robotics and programming. Its sensors include an ultrasonic distance sensor, battery sensor, accelerometer, gyroscope, odometers, IR receiver, line detection sensors, and light sensors, enabling a wide range of educational activities and experiments. The robot also boasts interaction features such as a speaker, front RGB LED, and a top cover for attaching additional components. Practical design elements, like a hole in the front for marker or pencil attachment and a special pulling loop, further enhance its usability in classroom settings. Additionally, FOSSBot is powered by rechargeable batteries, making it an eco-friendly and cost-effective solution for hands-

Co-funded by
the European Union

on learning. These features,combined with its open-source nature, make FOSSBot a highly adaptable and engaging platform for STEM education.

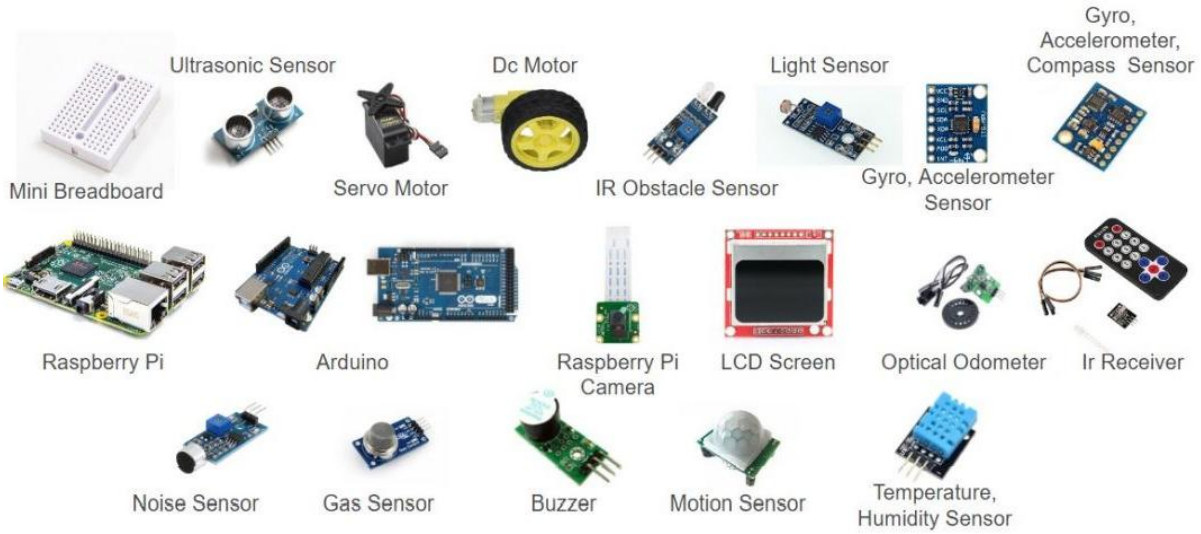


Figure 2. The list of FOSSBot electronics

FOSSBot is operated on a modular software stack that enables a variety of operation and programming modes, seamless orchestration through its administration GUI, and straightforward hardware control via a Python-based software library that functions as the robot's operating system. The stack integrates several interconnected components, including Google Blockly for block-based programming, Python Jupyter for interactive coding, and Python Flask, which hosts the administration and supervisor GUI. At its core, FOSSBot relies on a Python library that abstracts hardware control, allowing users to easily command the robot's movements, sensors, and actuators. This core library is accessible to all programming and operation modules, ensuring a unified and flexible development environment. The administration container oversees the robot's normal operation, enabling users to modify default parameters and maintain the system. While FOSSBot's current capabilities focus on basic 2D movement and sensor data collection, future development aims to expand its functionality with advanced features such as path planning and computer vision. Additionally, the system supports no-coding and block-based coding modes through a middleware interface that wraps core functionalities into REST API calls, ensuring accessibility for users of all skill levels. This modular and extensible architecture positions FOSSBot as a versatile platform for educational robotics.

Co-funded by
the European Union

Figure 3. The programming stack of FOSSBot

FOSSBot offers four distinct programming modes, each tailored to different skill levels and educational needs. The first mode, No-Coding Mode, is designed for users with little or no programming experience, such as pre-school children. In this mode, the robot is operated through a simple graphical interface with buttons that control basic movements like forward, backward, and rotational steps. Educators can customize this interface by adding new buttons linked to pre-defined or block-based scripts, enabling activities like line-following or sound playback. This mode not only introduces young learners to robotics but also allows educators to adjust the complexity of tasks based on the students' progress. The second mode, Block-Based Coding Mode, is ideal for primary school students and is built on the open-source Google Blockly platform. Here, users create programs by connecting coding blocks that represent commands, such as movement, sensor interactions, and LED control. These blocks are mapped to Python methods in the core FOSSBot library, allowing students to control the robot's hardware without writing code. The block-based scripts can be saved, modified, and even linked to buttons in the No-Coding Mode, providing a seamless transition between modes. This approach fosters computational thinking and problem-solving skills in a visual and intuitive way. Together, these two modes make FOSSBot accessible to beginners while gradually preparing them for more advanced programming challenges.

2.2. Limitations and Challenges of FOSSBot

Despite its innovative design and educational potential, the original FOSSBot faced several limitations and challenges that hindered its accessibility and usability, particularly for beginners and educators with limited technical expertise. One of the most significant issues was the **complexity of assembly**, which required users to manually connect multiple electronic components without the aid of a printed circuit board (PCB). This not only increased the risk of wiring errors but also made the assembly process time-consuming and intimidating for those unfamiliar with electronics. Additionally, the **3D printing of plastic parts** was both time-intensive and material-heavy, further delaying the readiness of the robot for classroom use. These factors

Co-funded by
the European Union

combined to create a steep learning curve, limiting FOSSBot's adoption in educational settings where simplicity and ease of use are critical.

Another challenge was the **lack of modularity and user-friendly design** in the original FOSSBot. The absence of a unified PCB meant that cables and connections were exposed and prone to disconnection, making the robot less durable and harder to maintain. Furthermore, the assembly process required specialized tools and a certain level of technical skill, which posed a barrier for educators and students who were not familiar with electronics or robotics. The **long printing and assembly times** also made it difficult to integrate FOSSBot into time-constrained classroom activities, reducing its practicality as an educational tool. Additionally, the **battery charging circuit** added to the cost and complexity of the robot, as it required specific components and careful assembly, further limiting its accessibility for schools with limited budgets or technical resources.

A notable usability challenge was the **initial connection process**, which required users to connect to FOSSBot as a hotspot. While this allowed for wireless interaction, the setup process was not always intuitive, especially for younger students or educators unfamiliar with networking concepts. This created a barrier to entry, as users had to navigate multiple steps to establish a connection before they could begin programming or controlling the robot. Moreover, while FOSSBot offered **multiple programming modes**, not all of them were equally useful or relevant for its primary target audience of primary and secondary education. For example, the **Python programming mode**, which provided low-level control over the robot, was often too advanced for younger students, while the **no-coding and block-based modes** were more aligned with their skill levels. This mismatch in programming modes highlighted the need for a more focused and streamlined approach to ensure that the robot's features were both accessible and educationally relevant for its intended users.

These challenges underscored the importance of redesigning FOSSBot to address these pain points, leading to the development of **FOSSBot 2.0**, which incorporates a new PCB, modular design, simplified battery compatibility, and a more intuitive connection process. By focusing on the needs of primary and secondary education, FOSSBot 2.0 aims to overcome these limitations and make STEM education more inclusive, engaging, and effective.

2.3. Desk Research Insights from 5 Countries

The development of S.T.E.P.S. and the redesign of FOSSBot were informed by extensive desk research conducted across five countries: **France, Greece, Poland, Spain, and Turkey**. This research, which included a review of educational policies, curricula, and best practices, highlighted both the opportunities and challenges of integrating educational robotics (ER) and STEM education into school systems. A key resource in this research was the book chapter that has been published by the S.T.E.P.S. consortium in a Springer Palgrave book. The chapter was titled *"Theoretical and Practical Coherence of Integrated STEM Education and Educational Robotics: Review and Analysis of Good Practices in Europe"*, co-authored by experts from the S.T.E.P.S. project. This chapter

Co-funded by
the European Union

analyzed ten best practices in ER and STEM education across Europe, identifying common trends, challenges, and areas for improvement.

The desk research analyzed **50 good practices** across the five European countries, focusing on four key parameters: **teaching methodologies**, **promotion of co-education**, **types of robots used**, and **integration of STEM areas**. The findings revealed that **cooperative and collaborative learning** was the most prevalent teaching methodology, used in 46 out of 50 practices, followed by **constructionism** (23 practices) and **project-based learning (PBL)** (21 practices). These methodologies emphasize interactive, hands-on, and group-based learning, which are highly effective in engaging students and fostering critical thinking. However, the research also highlighted a significant gap in **co-education**, with 88% of practices failing to address gender equality. This underscores the need for more initiatives to encourage girls to pursue STEM studies and ensure inclusivity in robotics education.

In terms of **robotic resources**, the analysis showed a preference for **custom-built robots** and **Lego Mindstorms kits**, each used in 15 practices, while **Arduino boards** were utilized in 6 practices. This indicates a trend toward both low-cost, customizable solutions and commercially available kits, depending on the educational context. The integration of **STEM disciplines** varied, with **technology and engineering** being the most commonly integrated areas (over 90% of practices), followed by **science** (70%) and **mathematics** (62%). This suggests that while robotics naturally aligns with technology and engineering, there is room for improvement in integrating mathematics and science more cohesively into ER activities.

The research also examined the good practices by country, revealing distinct trends and preferences. For example:

- **France** favored **self-built robots** using low-cost materials and emphasized **cooperative learning** and **constructionism**. However, it lagged in integrating mathematics and science into its practices.
- **Spain** and **Greece** showed a strong preference for **Lego Mindstorms kits** and **project-based learning**, with Spain also leading in the promotion of co-education.
- **Poland** stood out for its **diverse use of robot types** and its focus on **co-education**, with four practices explicitly addressing gender equality.
- **Turkey** and **Greece** demonstrated robust integration of all four STEM areas, though Turkey faced challenges in teacher training and resource accessibility.

One of the most significant findings was the **lack of compulsory integration of ER into national curricula** across Europe. While countries like Spain and France have made strides in incorporating robotics into their educational frameworks, ER is often relegated to extracurricular activities, weekend workshops, or summer camps. For example, Spain's 2022 Royal Decree introduced a compulsory "computational thinking, automation, and robotics" block in secondary education, but similar mandates are rare in other countries. In Greece, robotics is taught within the context of informatics and supported by initiatives like the "Skills Workshops" program, but it remains optional. This inconsistency highlights the need for a more standardized approach to ER integration,

Co-funded by
the European Union

ensuring that all students, regardless of location or socio-economic status, have access to robotics education.

The research also revealed **common barriers to ER adoption**, including high equipment costs, lack of trained teachers, and insufficient infrastructure. For instance, while France's 2015 Digital Plan for Education accelerated the integration of robotics, many schools still struggle with the cost of commercial robot kits and the need for specialized laboratories. Similarly, in Turkey, despite significant advancements in STEM education through initiatives like TÜBİTAK and TEKNOFEST, challenges such as unequal access to resources and inadequate teacher training persist. These findings underscored the importance of developing **low-cost, open-source solutions** like FOSSBot, which can be easily replicated and adapted to different educational contexts.

Another critical insight was the **need for interdisciplinary and project-based learning** in STEM education. The reviewed best practices emphasized the importance of integrating at least two STEM disciplines in a cohesive manner, using methodologies like inquiry-based learning and the engineering design process. For example, Poland's educational policy promotes an interdisciplinary approach that combines sciences and arts, fostering creativity and critical thinking. Similarly, Greece's recent investment in robotics and STEM equipment, funded by the EU's National Recovery and Resilience Plan, aims to enhance hands-on, collaborative learning experiences. These findings informed the design of S.T.E.P.S., which emphasizes modularity, expandability, and the use of open-source tools to support diverse educational activities.

The research also highlighted the **gender gap in STEM and robotics education**, with a notable lack of initiatives aimed at encouraging girls to pursue STEM studies. While countries like Turkey have made efforts to promote co-education and gender equality in STEM, more work is needed to address this issue across Europe. This insight reinforced the need for inclusive design and educational content that appeals to all students, regardless of gender.

Finally, the desk research underscored the **importance of teacher training and support** in successfully implementing ER and STEM education. Many of the challenges identified, such as the lack of robotics teachers and the complexity of integrating new technologies into the classroom, can be addressed through targeted professional development programs. The S.T.E.P.S. project addresses this need by providing comprehensive resources, including a MOOC platform and a digital twin simulation, to support educators in using FOSSBot effectively.

In summary, the desk research provided valuable insights into the current state of ER and STEM education in Europe, highlighting both the progress made and the challenges that remain. These findings directly informed the design and development of S.T.E.P.S., ensuring that the project addresses real-world needs and promotes inclusive, accessible, and effective STEM education for all.

2.4. Initial Aims: Simplification and Usability

The redesign of FOSSBot into FOSSBot 2.0 was driven by a set of clear initial aims focused on simplifying assembly and usage, enhancing educational value, and ensuring affordability. These aims were informed by the limitations of the original FOSSBot, as well

Co-funded by
the European Union

as insights from desk research across five European countries, which highlighted the need for a more accessible and user-friendly educational robot. The overarching goal was to create a tool that could be easily adopted by educators and students, regardless of their technical expertise or access to resources.

Simplifying Assembly and Usage

One of the primary objectives was to reduce the complexity of building and operating the robot, making it accessible to beginners, including students and educators with limited technical skills. The original FOSSBot required users to manually connect multiple electronic components without a printed circuit board (PCB), leading to a time-consuming and error-prone assembly process. To address this, FOSSBot 2.0 incorporates a new PCB design that integrates multiple circuits, significantly reducing the number of cables and connections. This not only simplifies assembly but also improves the robot's durability and reliability. Additionally, the modular design of the 3D-printed plastic parts minimizes assembly time and allows users to easily replace or upgrade individual components without disassembling the entire robot. These improvements ensure that FOSSBot 2.0 can be quickly and efficiently assembled, even by users with no prior experience in robotics or electronics.

Enhancing Educational Value

Another key aim was to enhance the educational value of FOSSBot by providing a platform that supports multiple programming levels, from beginners to advanced users. The original FOSSBot offered four programming modes, but not all were equally useful for its target audience of primary and secondary education. FOSSBot 2.0 addresses this by refining its no-coding and block-based coding modes, which are ideal for younger students, while also supporting more advanced programming in Python for older students. The integration of Raspberry Pi as the main processing unit further enhances the robot's capabilities, enabling advanced projects such as data logging, machine learning, and IoT applications. Additionally, the inclusion of new sensors and peripherals, such as cameras and I2C-compatible components, expands the robot's functionality, allowing for a wider range of educational activities and experiments.

Ensuring Affordability

To make FOSSBot 2.0 accessible to schools and underserved communities, the redesign focused on reducing costs without compromising functionality. The optimized design minimizes material and production costs, while the use of 3D printing allows for local manufacturing, further lowering expenses. The robot's battery compatibility was also simplified, supporting rechargeable lithium batteries as well as other commonly available battery types. This ensures that users can power the robot with whatever resources are readily available, reducing the financial burden on schools and educators.

Addressing Usability Challenges

The original FOSSBot faced usability challenges, particularly with its initial connection process, which required users to connect to the robot as a hotspot. While this allowed for wireless interaction, the setup process was not always intuitive, especially for younger students or educators unfamiliar with networking concepts. FOSSBot 2.0

Co-funded by
the European Union

addresses this by streamlining the connection process and providing a more user-friendly interface for programming and controlling the robot. The inclusion of a digital twin simulation in CoppeliaSim further lowers the barrier to entry, allowing users to program and test the robot in a virtual environment before working with the physical hardware.

Promoting Inclusivity and Co-Education

Insights from the desk research highlighted the gender gap in STEM and robotics education, with a notable lack of initiatives aimed at encouraging girls to pursue STEM studies. FOSSBot 2.0 addresses this by incorporating inclusive design principles and educational content that appeals to all students, regardless of gender. The robot's LEGO-compatible top panel and modular design encourage creativity and experimentation, making it an engaging tool for both boys and girls. Additionally, the project emphasizes co-education by providing resources and activities that promote gender equality in STEM education.

Supporting Teacher Training and Professional Development

Finally, the redesign of FOSSBot 2.0 recognizes the importance of teacher training and support in successfully implementing ER and STEM education. Many of the challenges identified in the desk research, such as the lack of robotics teachers and the complexity of integrating new technologies into the classroom, can be addressed through targeted professional development programs. The S.T.E.P.S. project provides comprehensive resources, including a MOOC platform and digital twin simulation, to support educators in using FOSSBot effectively. These resources ensure that teachers have the knowledge and tools they need to integrate robotics into their curriculum and engage students in hands-on, project-based learning.

By focusing on these initial aims, FOSSBot 2.0 overcomes the limitations of its predecessor and provides a more accessible, versatile, and effective tool for STEM education. The redesign ensures that the robot is not only easier to assemble and use but also more aligned with the needs of educators and students, making it a valuable resource for promoting inclusive and engaging STEM learning experiences.

Co-funded by
the European Union

3. Design Philosophy and Goals

The design of FOSSBot 2.0 is guided by a clear philosophy and set of goals aimed at creating an accessible, versatile, and effective educational tool for STEM learning. These principles ensure that the robot meets the needs of diverse users, from students and educators to hobbyists and researchers, while promoting inclusivity, affordability, and innovation in robotics education.

3.1. Core Design Principles

The core design principles of FOSSBot 2.0 are rooted in simplicity, accessibility, and educational value, ensuring that the robot is both easy to use and highly effective as a teaching tool. At its heart, FOSSBot 2.0 is designed to lower the barriers to entry for robotics and STEM education, making it accessible to students and educators with varying levels of technical expertise. This is achieved through a modular and intuitive design, which simplifies assembly and operation, and a user-friendly programming interface that supports multiple skill levels, from no-coding modes for beginners to advanced Python scripting for more experienced users. By prioritizing ease of use, FOSSBot 2.0 ensures that even those with no prior experience in robotics can quickly get started and engage in meaningful learning activities.

In addition to accessibility, FOSSBot 2.0 emphasizes durability, reliability, and adaptability. The robot is built to withstand the rigors of classroom use, with robust materials and a streamlined design that minimizes wear and tear. Its open-source architecture encourages collaboration and innovation, allowing users to customize and expand its capabilities to suit their specific needs. Furthermore, the robot's alignment with educational standards ensures that it can be seamlessly integrated into existing curricula, supporting a wide range of STEM learning objectives. These core principles guide every aspect of FOSSBot 2.0's design, making it a versatile and impactful tool for fostering creativity, critical thinking, and problem-solving skills in learners of all ages.

3.2. Focus on Open-Source and Accessibility

FOSSBot 2.0 is built on the foundation of open-source hardware and software, ensuring that all design files, code, and documentation are freely available to the public. This approach not only reduces costs but also fosters a collaborative community where educators, students, and developers can contribute to the robot's development and improvement. By making the robot's components 3D-printable and its software openly accessible, FOSSBot 2.0 eliminates the financial and technical barriers often associated with educational robotics. This democratizes access to robotics education, enabling schools and individuals with limited budgets to participate in STEM learning. Additionally, the digital twin simulation in CoppeliaSim provides a no-cost entry point for users who may not have immediate access to the physical robot, further enhancing accessibility.

Co-funded by
the European Union

The open-source nature of FOSSBot 2.0 also encourages innovation and customization. Users can modify the robot's design, add new features, or integrate it with other platforms, such as Raspberry Pi or Arduino, to suit their specific educational needs. This flexibility ensures that FOSSBot 2.0 remains relevant and adaptable to evolving educational trends and technologies. Moreover, the project's MOOC platform offers free access to STEM learning resources, tutorials, and project ideas, supporting educators and students in their robotics journey. By embracing open-source principles, FOSSBot 2.0 not only makes robotics education more accessible but also empowers users to take ownership of their learning experience, fostering a culture of collaboration and innovation.

3.3. Simplification of Assembly and Usage

One of the primary goals of FOSSBot 2.0 is to simplify the assembly and usage process, making the robot accessible to users of all skill levels. The original FOSSBot required manual wiring and complex assembly, which posed a significant barrier for beginners. FOSSBot 2.0 addresses this challenge with a new PCB design, with names to all connections and an integration of multiple circuits that reduces the number of cables and connections. This streamlined design not only simplifies assembly but also improves the robot's reliability by minimizing potential points of failure. Additionally, the modular plastic components are designed to snap together easily, eliminating the need for specialized tools and reducing assembly time. These improvements ensure that even users with no prior experience in robotics or electronics can quickly and efficiently assemble the robot.

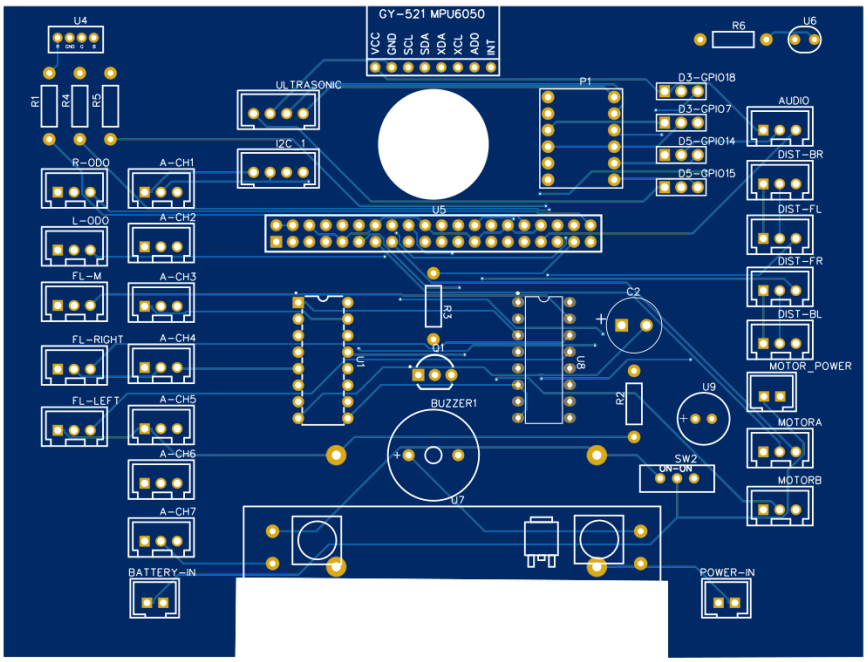


Figure 4. The design of the new PCB

Beyond assembly, FOSSBot 2.0 focuses on simplifying the user experience through intuitive programming interfaces and streamlined operation. The robot supports no-coding and block-based coding modes, which are ideal for younger students and

Co-funded by
the European Union

beginners, as well as more advanced programming options like Python for older students. The administration GUI provides a user-friendly interface for controlling the robot and modifying default parameters, while the wireless connectivity eliminates the need for cables or USB connections, making the robot easily connectable to the local network and more convenient to use. These features ensure that FOSSBot 2.0 is not only easy to assemble but also straightforward to operate, enabling educators and students to focus on learning and experimentation rather than technical setup. By prioritizing simplicity and usability, FOSSBot 2.0 removes the intimidation factor often associated with robotics, making STEM education more approachable and engaging for all.

3.4. Modularity and Customizability

FOSSBot 2.0 is designed with modularity and customizability at its core, ensuring that the robot can be easily adapted to meet the diverse needs of educators and students. The hardware design is divided into three main modular parts: the base, the middle section, and the top cover. The base houses the motors, wheels, and battery compartment, providing the robot with mobility and power. The middle section contains the main electronics, including the Raspberry Pi, sensors, and the new integrated PCB, which simplifies wiring and reduces assembly complexity. The top cover is designed to be LEGO-compatible, allowing users to attach additional components or accessories, fostering creativity and experimentation. As shown in Figure 5, the plastic parts of the robot are fewer and easier to assemble in only 7 steps. This modular design not only makes assembly and maintenance easier but also enables users to customize the robot for specific projects or learning objectives.

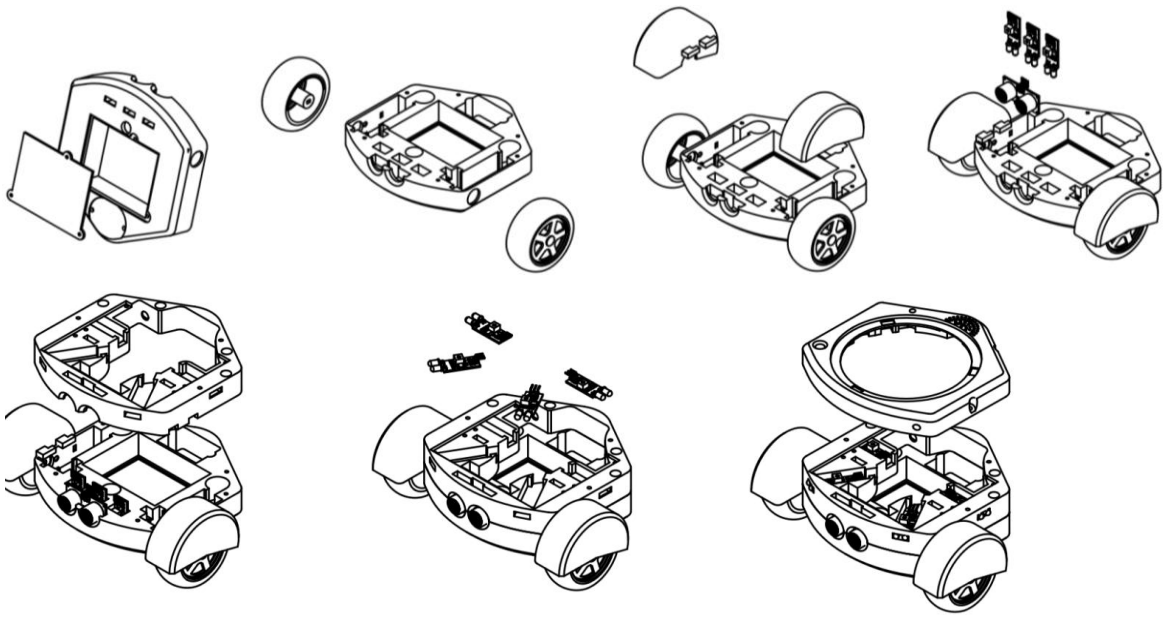


Figure 5. FOSSBot assembly in 7 steps

The integrated PCB is a key feature of FOSSBot 2.0's modularity, as it consolidates multiple circuits into a single board, reducing the number of cables and connections. This not only simplifies the assembly process but also improves the robot's reliability by minimizing potential points of failure. The PCB includes JST clips for easy sensor attachment and replacement, making it simple for users to upgrade or modify the robot's

Co-funded by
the European Union

capabilities. Additionally, the I2C connectivity allows for the integration of additional sensors and peripherals, enabling advanced projects such as computer vision or environmental monitoring. This modular hardware design ensures that FOSSBot 2.0 can grow with the user's skills and interests, supporting a wide range of educational activities and applications.

The software architecture of FOSSBot 2.0 is equally modular, building on a core Python library that controls the robot's main operations, including movement, sensor data collection, and actuator control. This library provides a high level of abstraction, making it easy for users to command the robot's hardware without needing to understand the underlying electronics. On top of this core library, the modular software stack includes additional layers that support the robot's programming interfaces. The administration GUI, built using Python Flask, allows users to control the robot and modify default parameters through a user-friendly web interface. The Blockly-based programming interface provides a visual, drag-and-drop environment for creating programs, which are then converted into Python code and executed by the core library. This modular software stack ensures that FOSSBot 2.0 is accessible to users of all skill levels, from beginners using no-coding or block-based modes to advanced users writing custom Python scripts. By combining modular hardware and software, FOSSBot 2.0 offers unparalleled flexibility and customizability, making it a versatile tool for STEM education.

3.5. Alignment with Educational Standards

FOSSBot 2.0 is meticulously designed to align with national and international educational standards, ensuring seamless integration into formal and informal STEM curricula. For example, in Spain, the robot aligns with Royal Decree 217/2022, which mandates the inclusion of "computational thinking, automation, and robotics" in secondary education. FOSSBot's programming modes, from no-coding to Python scripting, directly support the development of computational thinking and problem-solving skills outlined in this standard. Similarly, in France, the robot aligns with the Digital Strategy for Education 2023-2027, which emphasizes hands-on digital literacy and project-based learning. FOSSBot's modular design and sensor-driven activities mirror the strategy's focus on fostering technical understanding and creativity through interactive experimentation.

Globally, FOSSBot 2.0 aligns with the EU's DigComp Framework, which identifies digital competence as a key skill for lifelong learning. The robot's Blockly-based coding interface and Raspberry Pi integration address competencies such as "computational thinking" (DigComp 2.2) and "digital content creation" (DigComp 3.3). Additionally, FOSSBot supports NGSS (Next Generation Science Standards) in science and engineering, particularly practices like "analyzing data" (NGSS Practice 4) and "designing solutions" (NGSS Practice 6), through sensor-based experiments and project-based challenges. For instance, students can use the robot's ultrasonic sensor to collect environmental data, analyze it with Python scripts, and design solutions to real-world problems, directly tying into cross-disciplinary STEM objectives.

Co-funded by
the European Union

4. Development Process

The development of FOSSBot 2.0 was a collaborative and iterative process, driven by user feedback, desk research, and a commitment to open-source principles. This section outlines the key stages of the development process, from initial conceptualization to overcoming challenges, and highlights how the project evolved to meet its goals of accessibility, usability, and educational relevance.

4.1. Conceptualization and Ideation

The development of FOSSBot 2.0 began with the **conceptualization and ideation** phase, where the team aimed to build upon the existing open-source FOSSBot to create a more advanced and user-friendly educational robot. The initial concept was to take the original FOSSBot, which was already a versatile tool for STEM education, and enhance it to meet the specific goals of the S.T.E.P.S. project. This included addressing limitations such as **complex assembly**, **long printing times**, and **limited sensor support**, while also incorporating new features like **wireless connectivity**, **modularity**, and **advanced programming capabilities**. The team conducted a thorough review of existing educational robots, both open-source and commercial, to identify best practices and areas for improvement. This research informed the design goals, which focused on creating a robot that was not only affordable and easy to use but also adaptable to diverse educational contexts.

The ideation phase also involved **engaging with educators** to understand their needs and preferences. Through **desk research**, the team identified over 50 best practices in educational robotics across the five participating countries (France, Greece, Poland, Spain, and Turkey). These practices highlighted the importance of **project-based learning**, **interdisciplinary integration**, and **low-cost solutions**, which became key pillars of FOSSBot 2.0's design. Additionally, the team recognized the need to address the **gender gap in STEM education** and ensure that the robot appealed to all students, regardless of gender or technical background. This phase laid the foundation for a robot that would not only meet educational standards but also inspire creativity and innovation in learners.

4.2. Iterative Design and Prototyping

The development of FOSSBot 2.0 followed an **iterative design and prototyping** process, where multiple versions of the robot were created and tested to refine its features and functionality. The first prototypes focused on addressing the **assembly complexity** of the original FOSSBot by introducing a **modular design** with three main parts: the base, middle section, and top cover. This modular approach allowed for easier assembly and maintenance, while also enabling users to customize the robot for specific projects. The team also developed a **new PCB** to simplify wiring and reduce the number of connections, making the robot more reliable and user-friendly.

Co-funded by
the European Union

Throughout the prototyping phase, the team conducted **three training sessions** with educators from the participating countries, using the initial FOSSBot to gather feedback. These sessions employed the **FUFU (Formative Usability Feedback and Usability) analysis** method, which provided valuable insights into the robot's usability and educational value. Educators highlighted the need for **shorter printing times**, **better sensor integration**, and **more intuitive programming interfaces**, which were incorporated into subsequent prototypes. The iterative process also involved testing the robot in real-world classroom settings, where educators and students provided feedback on its performance and usability. This hands-on approach ensured that FOSSBot 2.0 evolved to meet the practical needs of its users.

4.3. Incorporating Feedback from Desk Research

The desk research conducted across the five participating countries played a crucial role in shaping the design and functionality of FOSSBot 2.0. The research identified **10 best practices** from each country, which were analyzed to identify common trends and challenges in educational robotics. One of the key findings was the preference for **low-cost, customizable solutions** over expensive commercial kits, which often lacked flexibility. This insight reinforced the team's commitment to open-source principles and informed the decision to make FOSSBot 2.0's design files and software freely available. Additionally, the research highlighted the importance of **project-based learning** and **interdisciplinary integration**, which were incorporated into the robot's programming modes and sensor suite.

The feedback from educators also revealed a strong demand for **simplified assembly and usage**, as well as **better support for teacher training**. To address these needs, the team developed a **digital twin simulation** in CoppeliaSim, allowing users to program and test the robot in a virtual environment before working with the physical hardware. This feature not only lowered the barrier to entry but also provided a valuable tool for teacher training and professional development. By incorporating feedback from desk research and user testing, FOSSBot 2.0 was able to address the real-world challenges faced by educators and students, ensuring its relevance and effectiveness in diverse educational settings.

4.4. Collaboration with Open-Source Communities

A key aspect of FOSSBot 2.0's development was its **collaboration with open-source communities**, which played a vital role in refining the robot's design and functionality. The team actively engaged with educators, developers, and hobbyists through online platforms, workshops, and training sessions, gathering feedback and suggestions for improvement. In Greece, for example, the team collaborated with a **network of over 500 educators** who were trained using the simulation environment, as well as a dozen educators who received and used the physical robot. This collaboration provided valuable insights into the robot's usability and educational value, which were used to refine its design and features.

Co-funded by
the European Union

The open-source nature of FOSSBot 2.0 also encouraged contributions from the broader community, with developers and educators proposing new features, fixing bugs, and creating educational content. This collaborative approach not only improved the robot's functionality but also fostered a sense of ownership and engagement among users. By leveraging the expertise and creativity of open-source communities, FOSSBot 2.0 was able to evolve into a more robust and versatile tool for STEM education.

4.5. Challenges and Solutions

The development of FOSSBot 2.0 was not without its challenges. One of the main obstacles was the **preference of educators for popular commercial robots**, which are often closed-source and difficult to customize. To address this, the team emphasized the **advantages of FOSSBot 2.0**, including its **lower cost**, **greater flexibility**, and **open-source design**, which allows users to modify and expand its capabilities. The team also highlighted the **digital twin simulation**, which provides an easy entry point for users who may be hesitant to adopt a new platform.

Another challenge was ensuring that FOSSBot 2.0 met the **diverse needs of educators and students** across different countries and educational contexts. To overcome this, the team adopted a **user-centered design approach**, incorporating feedback from educators and students throughout the development process. This iterative approach ensured that the robot was not only easy to use but also aligned with educational standards and curricula. By addressing these challenges through collaboration, innovation, and a commitment to open-source principles, the team was able to create a robot that meets the needs of its users and supports the goals of the S.T.E.P.S. project.

Co-funded by
the European Union

5. Technical Design of the new FOSSBot

The technical design of FOSSBot 2.0 significantly improves the original FOSSBot, prioritizing ease of assembly and reliability. Predefined slots are incorporated into the robot structure to clearly indicate component placements, resulting in a more intuitive assembly process. Furthermore, the 3D printing process is optimized to substantially reduce print failures. Components are redesigned to facilitate straightforward attachment and detachment, supported by strategically placed screws to enhance structural robustness.

Extensive material testing is performed to identify the most suitable printing material, ensuring an optimal balance between durability and ease of printability. Additionally, a custom Printed Circuit Board (PCB) is designed to minimize wiring complexity, incorporating integrated sensors and standardized connectors. This approach not only resolves issues identified in the original design but also allows for future expansions and modular enhancements.

5.1. Mechanical Design

In the initial design, the robot consisted of multiple printable parts with a hollow interior, which compromised its robustness. The designated areas for components often resulted in loose fits, requiring hot glue to keep components securely in place. Additionally, the original design made accessing the battery pack and Raspberry Pi challenging once fully assembled. Moreover, the positioning of several components posed risks of accidental short-circuiting during everyday vibrations.

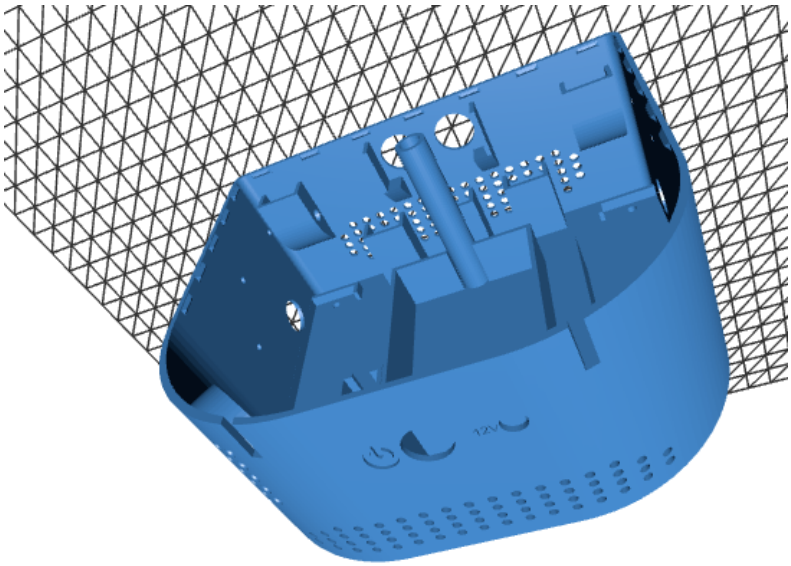


Figure 6. The main body of the original FOSSBot

Co-funded by
the European Union

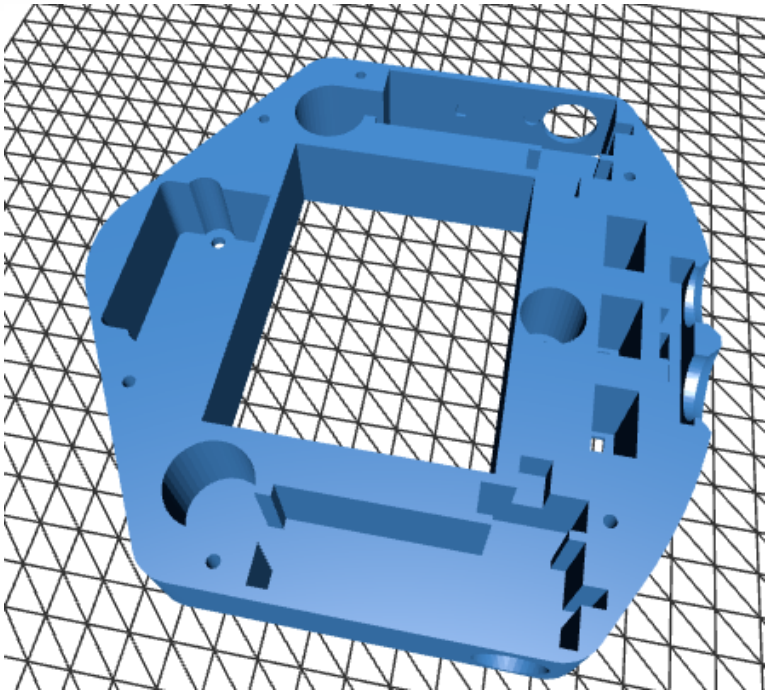


Figure 7. The new design of FOSSBot 2.0 lower body part, with the sockets for placing sensors and motors

In the new design, the robot's main body is divided into three parts. Each part is designed to avoid unnecessary hollow spaces by effectively utilizing 3D printing techniques, such as optimized infill, resulting in increased solidity and robustness. Each component now has clearly designated slots that ensure a tight fit, and the three main body parts are securely joined using screws placed in strategic locations. This "sandwich" approach ensures that all internal components remain firmly in place.

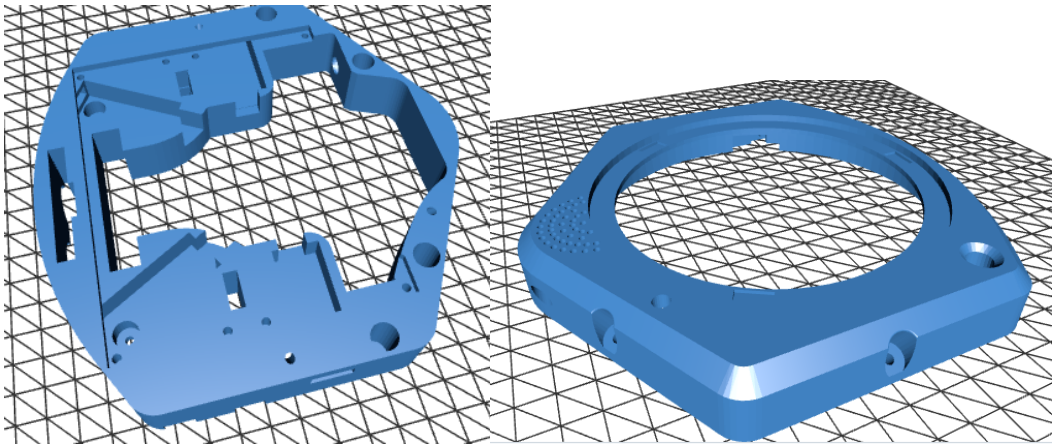


Figure 8. FOSSBot 2.0 middle and top body parts

Significant effort was made to safely placing power-related components. Specifically, high-voltage components are carefully positioned and isolated to prevent short circuits, even under extreme vibrations. The battery pack has been relocated to the robot's bottom and is covered by a specially designed access panel, allowing easy battery

Co-funded by
the European Union

replacement without full disassembly. Additionally, the new design includes a multipurpose magnetic cover featuring a LEGO-compatible surface, providing convenient access to the robot's interior.

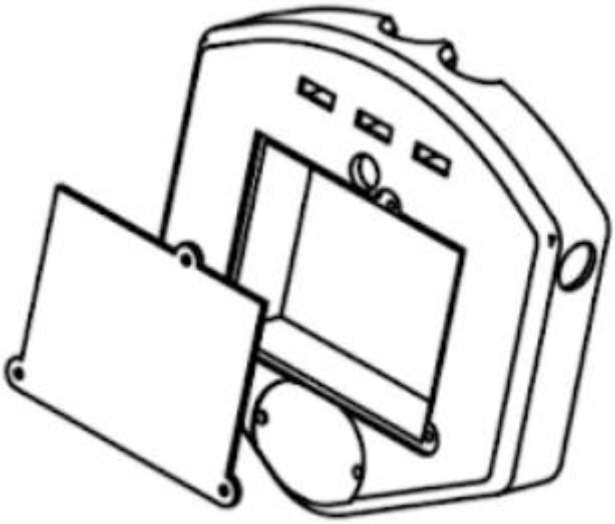


Figure 9. The new batteries slot in the lower part of the FOSSBot 2.0 body

The pen holder component has also been redesigned for improved durability and ease of use. Since we had reports from the users of the original FOSSBot that it easily breaks, the new pen holder is now more robust and removable, ensuring a tighter fit for pencils and allowing easy reprinting if damaged.

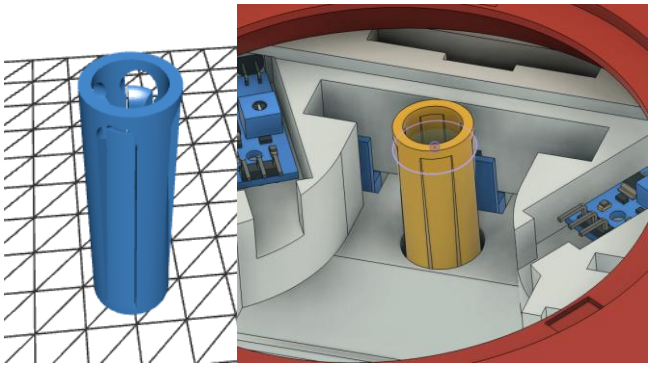


Figure 10. The newly designed, detachable pen holder

Lastly, the Raspberry Pi has been repositioned directly under the magnetic top cover, providing convenient access for camera or screen interface connections, as well as easy access to the micro-SD slot for future firmware updates.

Finally, after conducting research and testing, we determined that the optimal 3D printing material combining durability and minimal print failures is PET-G, commonly used in plastic water bottles. PET-G offers excellent durability, flexibility without cracking, resistance to UV light, sunlight exposure, and good thermal stability after

Co-funded by
the European Union

printing. While the choice of printing material ultimately rests with the end-user, PET-G is highly recommended for optimal results.

The key elements of the new design can be summarized as follows:	
	● Three-part modular chassis for improved robustness. ● Intuitive slots and screw placement for easy assembly. ● Optimized 3D printing to reduce print failures. ● PET-G material selection for durability and flexibility. ● Battery pack repositioned with accessible cover for easy replacement. ● Magnetic LEGO-compatible top cover for easy internal access. ● Redesigned removable pen holder for enhanced durability and usability. ● Safe isolation of high-voltage components to avoid short circuits.

5.2. Electronics Design

In the new robot design, we introduced a custom PCB to simplify the assembly process significantly. In the original version, electronic components were connected using numerous wires, which often overwhelmed users. The large number of wires also made it challenging to follow instructions clearly, and any disconnections were difficult to troubleshoot.

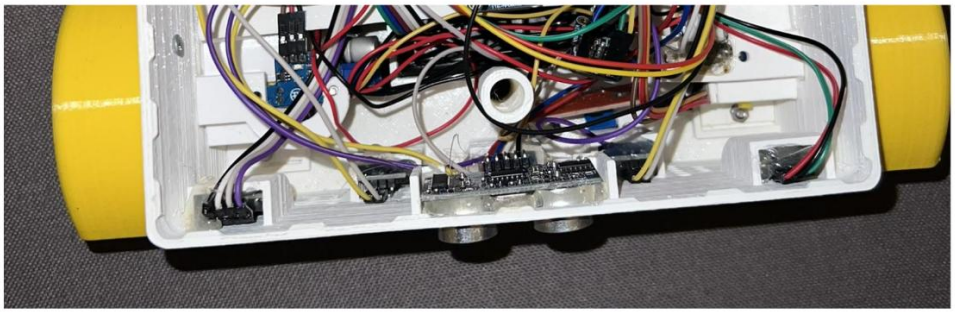


Figure 11. A part of the wiring of the original FOSSBot

The PCB that has been designed in terms of S.T.E.P.S. minimizes the need for multiple small PCBs that previously hosted simple components like LEDs or photoresistors. By integrating these components directly into the main PCB, we reduced wiring complexity and lowered overall costs. The PCB now utilizes JST connectors, creating a standardized

Co-funded by
the European Union

and secure method for attaching components to the robot. Additionally, the PCB is specifically designed to allow easy access to the Raspberry Pi from the robot's top. This design enhances user convenience by facilitating connections to the Raspberry Pi camera and screen interfaces, enabling further sensor expansions.

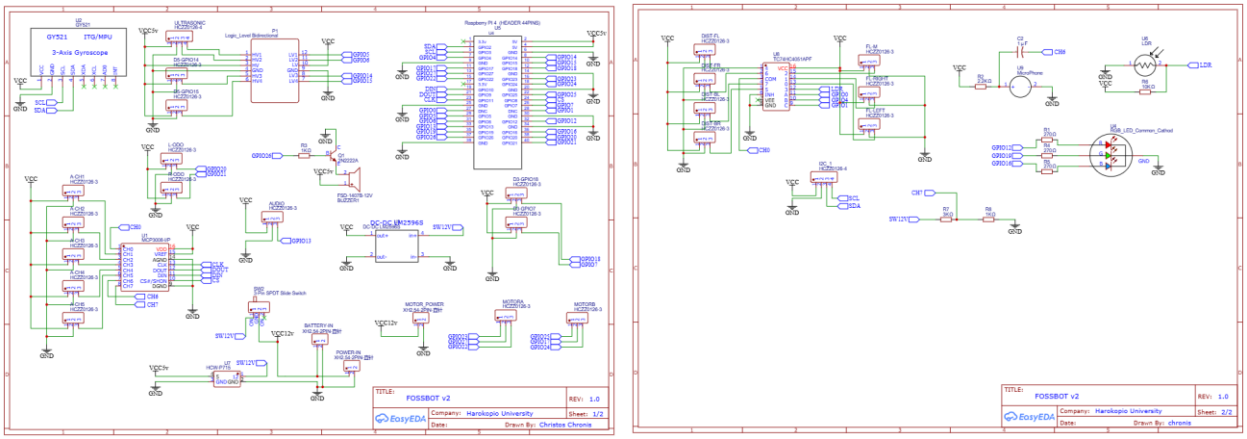


Figure 12. The schematics of the PCB of FOSSBot 2.0

The new PCB incorporates additional integrated components, such as a multiplexer, providing the robot with more digital and analog pins to accommodate a broader range of sensors. The I2C interface is exposed via a JST connector, allowing the easy connection of I2C-based electronic components like small LCD screens or PWM controllers. Furthermore, the robot now supports not only the original Li-Ion battery pack but also any 12V battery.

Integrated directly into the PCB are components such as a photoresistor, RGB LED, buzzer, noise sensor, battery sensor, and I2C port. It includes four digital component ports and five additional analog component ports for extra sensors, as well as four additional IR distance sensors for perimeter object detection. The Raspberry Pi interfaces for camera and screen access are clearly exposed for easy connectivity. The PCB retains compatibility with the original robot's ultrasonic sensor, audio output, three IR distance sensors, and motor driver however, these components now connect via standardized JST connectors.

The key elements of the new PCB designs:	
	● Custom PCB to simplify assembly and reduce wiring complexity. ● Standardized JST connectors for reliable component connections. ● Integrated multiplexer expanding available pins for additional sensors. ● Compatibility with various battery types (Li-Ion and 12V).

Co-funded by
the European Union

	● Clearly exposed Raspberry Pi interfaces for easy camera/screen integration. ● Built-in sensors and components (photoresistor, RGB LED, buzzer, noise sensor, battery sensor). ● Expanded sensor support including additional IR distance sensors and I^2C-based expansions.

5.3 Software Architecture and Connectivity

The software architecture of FOSSBot 2.0 has been significantly enhanced to improve usability, flexibility, and performance while maintaining compatibility with the existing main stack. These improvements focus on simplifying the user experience, extending functionality, and ensuring seamless integration between the simulation environment and the physical robot. Key updates include the removal of the Jupyter notebook option, the extension of the main robot control library, the addition of new Blockly blocks, the development of a browser-based CoppeliaSim plugin, and the implementation of thorough testing to ensure code portability.

Simplified Programming Interface

To make the platform more accessible to users with varying levels of programming expertise, the Jupyter notebook option has been removed. Instead, the focus has been placed on visual no-code programming using Blockly. This decision aligns with the goal of making FOSSBot 2.0 more user-friendly, particularly for educational settings where students may not have prior coding experience. The Blockly interface has been expanded with new blocks to support the additional sensors and components introduced in the updated hardware design. These blocks enable users to easily program advanced functionalities, such as object detection, sound recognition, and environmental monitoring, without writing a single line of code.

Extended Robot Control Library

The main robot control library has been extended to support the new sensor components integrated into the custom PCB, such as the photoresistor, RGB LED, buzzer, noise sensor, and additional IR distance sensors. This extension ensures that all hardware components are fully accessible through the software interface, enabling users to leverage the full capabilities of the robot. The library now includes pre-built functions for common tasks, such as reading sensor data, controlling motors, and managing I2C-based devices, reducing the complexity of programming and allowing users to focus on higher-level logic.

Co-funded by
the European Union

Browser-Based CoppeliaSim Plugin

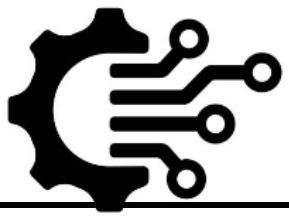
To address the challenges of installing and using the CoppeliaSim robot simulation environment, a browser-based plugin has been developed. This plugin eliminates the need for complex local installations and ensures that users can access the simulation environment directly from their web browsers. The plugin is fully integrated with the Blockly programming interface, allowing users to test their code in a simulated environment before deploying it to the physical robot. This approach not only simplifies the setup process but also enhances the learning experience by providing immediate feedback and reducing the risk of errors when transitioning from simulation to real-world execution.

Enhanced Code Portability

A major focus of the software improvements has been ensuring the portability of code between the simulated environment and the actual robot. To achieve this, extensive testing has been conducted to validate that programs written for the simulation environment behave identically when executed on the physical robot. This testing includes edge cases, such as varying sensor inputs, motor control precision, and environmental conditions, to ensure robustness and reliability. As a result, users can confidently develop and test their programs in the simulation environment, knowing that they will perform as expected on the real robot.

Multilingual User Interface Support

To make FOSSBot 2.0 accessible to a global audience, the user interface (UI) now supports six languages: English, Spanish, French, German, Greek, and Italian. This multilingual support extends to all aspects of the software, including the Blockly programming interface, the CoppeliaSim plugin, and the robot control library documentation. The UI has been designed to dynamically switch between languages based on user preferences, ensuring a seamless experience for non-English speakers. This feature is particularly valuable in educational settings, where students and teachers may prefer to interact with the platform in their native language. The translation process involved collaboration with native speakers to ensure accuracy and cultural relevance, further enhancing the usability of the platform.

The key elements Software Architecture improvement are:	
	● Visual No-Code Programming: Exclusive use of Blockly for an intuitive and accessible programming experience. ● Extended Robot Control Library: Support for new sensors and components, with pre-built functions for common tasks.

Co-funded by
the European Union

	● New Blockly Blocks: Addition of blocks for advanced functionalities, such as object detection and environmental monitoring. ● Browser-Based CoppeliaSim Plugin: Simplified access to the simulation environment via a web browser. ● Thorough Testing: Comprehensive validation of code portability between simulation and real-world execution. ● Multilingual UI Support: User interface available in six languages (English, Spanish, French, German, Greek, and Italian) to cater to a global audience.

These software architecture improvements ensure that FOSSBot 2.0 remains a versatile and user-friendly platform for robotics education, while also providing the tools and flexibility needed for more advanced applications. By streamlining the programming interface, extending hardware support, and enhancing simulation capabilities, the updated software architecture empowers users to explore the full potential of the FOSSBot platform.

6. Key Features and Innovations

6.1. Simplified Assembly Process

The assembly process of FOSSBot 2.0 has been designed to be intuitive and educational, making it suitable for users of all ages, including students and educators. The robot's 3D-printed parts are modular and feature predefined slots, ensuring that components fit securely without the need for additional tools or adhesives. This design not only simplifies the assembly process but also turns it into a hands-on learning activity, where users can understand the relationship between mechanical design and functionality. Additionally, the use of fewer materials and optimized 3D printing techniques has significantly reduced printing time and material waste, making the robot more environmentally friendly and cost-effective.

The introduction of a custom PCB board has further streamlined the assembly process by minimizing the number of cables required. In the previous version, the extensive wiring was a common source of confusion and errors during assembly. The new PCB integrates most electronic components and uses standardized JST connectors, reducing wiring complexity and making it easier for users to connect sensors, motors, and other components. This simplification ensures that even users with limited technical expertise

Co-funded by
the European Union

can assemble the robot efficiently, fostering a sense of accomplishment and encouraging further exploration.

6.2. Enhanced Durability and Usability

FOSSBot 2.0 has been designed with durability and usability as key priorities. The 3D-printed plastic parts are now more robust, thanks to the use of optimized infill patterns and the recommended PET-G material, which offers excellent durability, flexibility, and resistance to wear and tear. Each printing material has been carefully evaluated to ensure it strikes the right balance between strength and ease of printing, making the robot suitable for long-term use in educational environments.

Moreover, the modular design of the robot ensures that sensors and electronic components can be easily detached and reused in other projects. This not only extends the lifespan of the components but also encourages creativity and experimentation. For example, students can repurpose the robot's sensors for other STEM projects, such as environmental monitoring or interactive art installations. This reusability aligns with the principles of sustainability and cost-effectiveness, making FOSSBot 2.0 a versatile tool for both learning and innovation.

6.3. Support for Multiple Programming Levels

FOSSBot 2.0 caters to users with varying levels of programming expertise by offering a modular programming environment with two modes: Kindergarten and Blockly. The Kindergarten mode is designed for young learners and provides a simple, icon-based interface that introduces basic programming concepts through playful interactions. The Blockly mode offers a visual, drag-and-drop programming interface, making it ideal for beginners and intermediate users who want to create more complex programs without writing code.

By supporting multiple programming levels and a no-code UI FOSSBot 2.0 ensures that it can grow with the user, providing a continuous learning pathway from basic concepts to advanced robotics programming.

6.4. Expandable and Customizable Modules

FOSSBot 2.0 is designed to be highly expandable and customizable, making it suitable for a wide range of educational applications and age groups. The robot's modular architecture allows educators to tailor its functionality to specific lessons or experiments. For example, additional sensors, such as temperature or humidity sensors, can be easily integrated to support science experiments, while actuators like servos can be added for more advanced robotics projects.

This flexibility extends to the robot's software as well. The accompanying application can be customized to include specific programming challenges or educational content,

Co-funded by
the European Union

making it adaptable to different curricula and learning objectives. Whether used in a primary school classroom or a university robotics lab, FOSSBot 2.0 can be configured to meet the needs of diverse educational settings, ensuring its relevance across multiple subjects and age groups.

6.5. Cost-Effective and Open-Source Design

One of the core principles of FOSSBot 2.0 is its commitment to being cost-effective and open-source. All design files, including 3D models, PCB schematics, and assembly instructions, are freely available online, allowing anyone to build, modify, and improve the robot. This open-source approach not only reduces the cost of ownership but also fosters a community of users who can share knowledge, collaborate on projects, and contribute to the platform's ongoing development.

In addition to the design files, detailed instructions on how to construct and program the robot are provided, ensuring that even users with limited technical expertise can get started quickly. If any electronic components break or need to be replaced, they can be easily sourced from online electronics or robotics stores, further reducing maintenance costs. By combining affordability with accessibility, FOSSBot 2.0 democratizes robotics education, making it available to schools, hobbyists, and educators worldwide.

7. Comparison with FOSSBot

FOSSBot 2.0 represents a significant evolution from its predecessor, incorporating design, educational, and usability improvements while maintaining its cost-effective and open-source principles. This section highlights the key differences between FOSSBot and FOSSBot 2.0, emphasizing the advancements that make the new version more durable, accessible, and versatile for educational purposes.

7.1. Design and Structural Improvements

FOSSBot 2.0 features a more modern and durable design compared to the original FOSSBot. The new modular design for its 3D-printed plastic components minimizes assembly time and simplifies maintenance. Each part is designed with predefined slots and screw placements, ensuring a secure fit and reducing the need for additional adhesives. This modularity not only makes the robot easier to assemble but also allows for quick disassembly and repair, which is particularly useful in educational settings.
Additionally, FOSSBot 2.0 has been optimized to reduce 3D printing time and material usage. The use of advanced infill patterns and the recommended PET-G material ensures that the robot is both lightweight and robust. The overall size of the robot has also been reduced, making it more compact and portable without compromising its functionality. These design improvements make FOSSBot 2.0 more user-friendly and suitable for a wider range of applications.

Co-funded by
the European Union

7.2. Enhanced Educational Features

FOSSBot 2.0 introduces several features that enhance its educational value. The platform now supports multiple programming levels, catering to users of all skill levels, from kindergarten students to advanced programmers. The modular programming environment includes three modes: Kindergarten (icon-based), Blockly (visual programming), and Monaco (text-based coding), ensuring that the robot can grow with the user's skills.
The top panel of FOSSBot 2.0 is LEGO-compatible, allowing users to attach custom builds and expand the robot's capabilities creatively. The integration of CoppeliaSim as a browser-based plugin provides a seamless simulation experience, enabling users to test their programs in a virtual environment before deploying them to the physical robot. Furthermore, the inclusion of I2C connectivity allows for the easy integration of additional sensors and peripherals, making the robot a versatile tool for teaching robotics, programming, and STEM concepts.

7.3. Usability and Accessibility Comparison

FOSSBot 2.0 significantly improves usability and accessibility compared to its predecessor. The integration of a Raspberry Pi as a web server enables users to control the robot remotely via a web interface, enhancing its flexibility in classroom settings. Wi-Fi connectivity has been introduced, allowing users to connect to the robot wirelessly, eliminating the need for physical connections and simplifying the setup process.
The new custom PCB integrates multiple circuits, reducing the number of cables and connections required. This not only simplifies assembly but also makes troubleshooting easier. The use of JST connectors ensures that sensors and components can be attached and replaced quickly, further enhancing the robot's usability. Additionally, the top panel is now attached with magnets, providing easy access to internal components and making maintenance more straightforward.

7.4. Cost and Manufacturing Efficiency

FOSSBot 2.0 maintains the cost-effective and open-source principles of the original FOSSBot while introducing manufacturing efficiencies that reduce overall costs. The optimized 3D printing process minimizes material usage and printing time, lowering production costs. The custom PCB reduces the need for additional wiring and connectors, further cutting down on expenses.
All design files, including 3D models, PCB schematics, and assembly instructions, remain freely available online, ensuring that the robot is accessible to a wide audience. The use of standardized electronic components means that replacements can be easily sourced from online stores, reducing long-term maintenance costs. These improvements make FOSSBot 2.0 an even more affordable and sustainable option for schools, educators, and hobbyists.

Co-funded by
the European Union

Table 1. Comparison between FOSSBot and FOSSBot 2.0

Feature	FOSSBot	FOSSBot 2.0
Design	Single-part chassis	Modular 3-part chassis
Assembly	Complex wiring, hot glue needed	Simplified wiring, screw-based assembly
3D Printing Time (with the default printing speed)	59 hours	Reduced to 51 hours
Material Usage	380 gr	Reduced to 343 gr
Size	Larger	Smaller and more compact
Programming Levels	Limited	3 modes (Kindergarten, Blockly, Monaco)
LEGO Compatibility	No	Yes
Simulation	Local CoppeliaSim installation	Browser-based CoppeliaSim plugin
Connectivity	Wired	Wi-Fi and web server integration
PCB Design	Multiple small PCBs	Single custom PCB with JST connectors
Top Panel Access	Screw-based	Magnetic attachment
Cost	Low	Even lower due to manufacturing optimizations
Open-Source Availability	Yes	Yes

Co-funded by the European Union

FOSSBot 2.0 builds on the strengths of its predecessor while addressing its limitations, resulting in a more durable, accessible, and versatile platform for robotics education. These improvements ensure that FOSSBot 2.0 remains a cost-effective and open-source solution for educators and learners worldwide.

8. Conclusion

FOSSBot 2.0 marks a transformative advancement in educational robotics, building on the foundation of its predecessor while introducing innovative features that enhance usability, durability, and accessibility. This concluding section summarizes the key achievements of FOSSBot 2.0, highlights its potential impact on educational robotics, and issues a call to action for educators and developers to adopt and contribute to this open-source platform.

8.1. Summary of Achievements

FOSSBot 2.0 has achieved significant milestones that set it apart as a leading tool for robotics education. The robot's modular 3D-printed design simplifies assembly and maintenance, while optimized printing techniques reduce material usage and production time. The introduction of a custom PCB minimizes wiring complexity, enhances reliability, and supports a wide range of sensors and components, making the robot more user-friendly and versatile.

The platform's support for multiple programming levels—Kindergarten (icon-based), Blockly (visual programming), and Monaco (text-based coding)—ensures that it caters to users of all skill levels, from young learners to advanced programmers. The integration of a browser-based CoppeliaSim plugin and Wi-Fi connectivity enables seamless transitions between simulation and real-world execution, enhancing the learning experience. Additionally, the multilingual user interface (supporting six languages) and LEGO-compatible top panel expand its appeal and adaptability in diverse educational settings.

FOSSBot 2.0 also emphasizes sustainability and reusability, with durable 3D-printed parts and components that can be repurposed for other projects. Its open-source design, with all files and instructions freely available online, ensures that the robot remains accessible and affordable for schools, educators, and hobbyists worldwide.

8.2. Impact on Educational Robotics

FOSSBot 2.0 has the potential to revolutionize robotics education by providing a versatile, cost-effective, and accessible platform for teaching STEM concepts. Its

Co-funded by
the European Union

modular and expandable design makes it suitable for a wide range of applications, from introductory programming exercises to advanced robotics projects. By supporting multiple programming levels and offering a seamless simulation environment, FOSSBot 2.0 empowers students to learn at their own pace, experiment with new ideas, and develop critical problem-solving skills.

The robot's emphasis on durability and reusability ensures that it can withstand the demands of classroom use while promoting sustainability. Its open-source nature fosters a collaborative community of educators, developers, and students who can share knowledge, develop new features, and adapt the platform to meet evolving educational needs. By lowering the barriers to entry for robotics education, FOSSBot 2.0 has the potential to inspire the next generation of innovators, engineers, and scientists.

8.3. Call to Action for Educators and Developers

FOSSBot 2.0 is more than just a robot—it is a powerful tool for empowering learners and fostering a deeper understanding of robotics, programming, and STEM concepts. Educators are encouraged to integrate FOSSBot 2.0 into their curricula, leveraging its versatility to teach a wide range of subjects, from computer science and engineering to environmental science and creative design. By providing students with hands-on experience in robotics, educators can inspire curiosity, creativity, and a passion for learning.

Developers and hobbyists are invited to contribute to the ongoing development of FOSSBot 2.0. Whether by designing new sensors, creating additional Blockly blocks, improving the simulation environment, or translating the user interface into more languages, there are countless opportunities to enhance the platform and expand its capabilities. The open-source nature of FOSSBot 2.0 ensures that it will continue to evolve, driven by the collective efforts of a global community.

In conclusion, FOSSBot 2.0 represents a significant step forward in making robotics education accessible, engaging, and impactful. By embracing this platform, educators and developers can play a pivotal role in shaping the future of STEM education and inspiring the innovators of tomorrow. Together, we can unlock the full potential of robotics as a tool for learning, creativity, and innovation.

Co-funded by the European Union

References

Chronis, C., & Varlamis, I. (2022). Fossbot: An open source and open design educational robot. *Electronics*, *11*(16), 2606.

Mamatnabiyev, Z., Chronis, C., Varlamis, I., Himeur, Y., & Zhaparov, M. (2024). A holistic approach to use educational robots for supporting computer science courses. *Computers*, *13*(4), 102.

M. A. Merino-Fernández, A. I. Cuesta, A. Alonso-Centeno, L. A. Mínguez, I. Varlamis, C. Sofianopoulou, S. Aykuş, Ch. Rousoulioti, A. Vrantzas, I. Siakavaras, G. Papaioannou, N. Pischos, K. Burcke, T. Angelopoulos, and J. Ortiz-Revilla. Theoretical and practical coherence of integrated STEM education and educational robotics: review and analysis of good practices in Europe. In S. Papadakis & G. Lampropoulos (Eds.), Social Robots and Artificial Intelligence in Education: Integrating AI in K-12 and Higher Education. Palgrave Macmillan.

Co-funded by
the European Union