REGULAR PAPER

sBotics – Gamified Framework for Educational Robotics

Lucas Moura do Nascimento[1] · Davi Souto Neri[2] · Thiago do Nascimento Ferreira[2] ·
Francinaldo de Almeida Pereira[3] · Erika Akemi Yanaguibashi Albuquerque[3] · Luiz Marcos Garcia Gonçalves[3] (ORCID) ·
Sarah Thomaz de Lima Sá[2]

Abstract

This paper proposes a learning framework for Educational Robotics named sBotics, which includes a complete environment for teaching and programming skills acquisition designed for both teachers and K-12 students. Our framework has been developed using a gamified approach with the system and simulated environment developed in the Unity game engine. The main novelty of this platform is its ease-of-use combined with the flexibility to create a variety of scenarios with endless learning potential, in contrast to our evaluations where no alternatives with such characteristics were found for the K-12 range that we are targeting. Also as a contribution of our proposal, robot programs are treated as games that are affected by a disturbance model, which acts in the robotic system and environment variables. This model is introduced in order to approximate what happens in a real robot programming platform. Besides, it is possible for the user, throughout its use, to code in three levels of abstraction: its overly intuitive native programming language called R-Educ, BlockEduc (R-Educ version of Blockly), and C#. Programs can be compiled and interpreted by virtual robots executing any given command. As for teachers, the framework API offers tools that can be employed in the assembly and customization of the learning setup. The whole platform has been built as a tool dedicated to spreading the worlds of Robotics and Programming among youngsters, as well as making them more affordable to everyone. It has been validated by our experiments and is currently being used during the novel Coronavirus pandemic by the official RoboCupJunior Rescue trials in Brazil, currently with more than a thousand competing teams (about 5 thousands students).

Keywords Robots · Learning and programming framework · Educational robotics

1 Introduction

Educational Robotics can be drawn as a learning methodology focused on stimulating students' desire for learning new technologies and subjects [4]. This methodology is also known to attract students' interest with the use of investigation by inspiring their creativity, developing a scientific method through constructive feedback, stimulating logical thinking, improving motor skills, and many other methods [1, 4]. Educational Robotics methodology is traditionally applied through the use of kits, bought specifically for this

kind of activity [26, 27], or else by using electronic and mechanical components available in the market and even from scratch in a more cheap paradigm [4].

Nonetheless, another way to spread and apply educational robotics is through the use of simulated learning frameworks or environments, which, besides having the basic teaching capabilities, are software systems that are capable of reproducing real-world behaviors of a given system, replicating phenomena and sensations without physical implications. Nowadays, with the Covid-19 pandemic, this type of software has been gaining extreme importance, allowing kids to keep previous knowledge while gaining more abilities in programming.

Therefore, this paper aims to propose sBotics, an educational robotics environment with a gamified simulated learning framework, introducing the methods employed and results obtained with its application in Robocup challenges and classrooms in Brazil. It has been developed using the Unity game engine and C# to offer solutions to some problems that will be further mentioned, generating a

✉ Lucas Moura do Nascimento
  lucas.moura@ufrn.edu.br

Extended author information available on the last page of the article.

friendly and gamified teaching and learning environment for its users. With this tool, students are able to customize and program virtual robots and test their knowledge through the use of simulations. Additionally, students are capable of editing and sharing the virtual environment in their desired manner.

Although Brazil has numerous investments in technology, the workforce for this area needs to be increased with incentives to qualify laborers, in order to match many job opportunities, new technologies, techniques, and products in the country. Another application is the insertion of more students in this field (STEAM) without the need of importing kits for the assembly of educational robots, which discourages public and private schools to use this powerful teaching and learning methodology, underexposing students to this area [2, 4].

Regarding the difficulties of Educational Robotics' insertion in schools, this simulator helps to assist students in the development of their knowledge, without the need for high-cost equipment such as the aforementioned setups. It is capable of supplying the user's needs without depending on physical components. However, if a student or teacher already possesses robotics kits, the simulation is still useful due to its similarities to real-world scenarios, meaning that it is possible to test different types of problem-solving strategies without damaging the robots physically while also accelerating the algorithm's development cycle.

Besides, the gamified simulator framework proposed has an extremely attractive interface. Further, as the main contribution, the main goal of gamification is to not foster the existing and shared feeling in the robotics development community that if the simulation succeeded once it is doomed to succeed in every other attempt [11, 12]. Breaking with this premise, we propose an implementation that includes an environment randomization with the generation of more realistic objects and scenarios. This allows real-world problems and real-world unexpected issues to be present, demanding from the users greater attention in the writing of algorithms and preparing them for the problems and difficulties that will be faced when handling real robots in real situations.

Despite being developed mainly for the educational robotics use case, our proposal also contributes for applications in the several domains of Robotics as human-robot interaction, human-machine interfaces and interaction, and also on social and service robotics, which is also the case here. People and researchers from these topic-areas would benefit from our gamified framework and also from having an online interface, carrying out all programming and testing through the web, which previously demanded real robots. So, this paper provides a cutting edge technology that has been developed and applied to build, design, model and test complex engineering and autonomous systems. Actually, a complete special issue in Robotics in Education has been published at the Journal of Intelligent & Robotic Systems [13]. As such, we verified its applicability with students from Brazilian Robotics Olympiad (BRO) [3] and *RoboCup Brasil* (about 1200 participants) that have used it in 2020, during the pandemic time, in order to build and program their virtual robots, and submit it for competition, which ran online. The *W-Educ* site running at Natalnet Associate Labs' server (provided by the Federal University of Rio Grande do Norte Computing Division) [14] has had more than 40 thousand program compilations, by the previously-mentioned virtual participants of BRO.

Hence, the sBotics platform allows the complete simulation of every stage of the Brazilian trials of RoboCupJunior Rescue Line A, which has been the practical component of the Brazilian Robotics Olympiad [3] for over a decade. This competition is included in the Brazilian selectives for the World RoboCupJunior as an incentive for schools to invest in the Educational Robotics methodology. In this work case, the use of computers for simulation instead of robotics kits may foster educational institutions to adopt said methodology, as most schools already have computer laboratories and almost no additional structural costs would be needed. Due to the pandemic of the new coronavirus, sBotics was selected and is being officially used as an alternative tool so that the practical component of the Brazilian Robotics Olympiad could successfully take place in 2020, in a simulated manner (and through the Internet). We noticed that by producing a realistic simulation environment with random disturbances in all of the environment variables, including sensor errors, every trial of rescue would be different from the previous one, even with the same setup maintenance. This approximates this model from the real competition, where the students have severe problems due to light and other factors, mainly after training in a controlled environment. Results demonstrating the main impressions of users in the run-up to the 2020 Brazilian Robotics Olympiad will also be presented in this paper.

Actually, the work and results reported in this paper have been achieved after the previous version of this work [5]. The simulators' first pre-release was completed after several months of development. After that initial work was done and presented at the 2019 Latin American Robotics Symposium, the development team collected feedback from users to provide a lot more novelties, depicted here. Just recently, in April 2020, with the pandemic in course, it was suggested by *RoboCup Brasil* Trustees to be used as the official simulator for RoboCupJunior regionals in Brazil for that year. We note that it has even been displayed as a powerful tool in other robotics conventions in Brazil besides BRO, such as the Brazilian National Robotics Exhibition. So the current version of the work is a further improvement of the

first one in several aspects, mainly in the disturbance model, which is introduced to approximate the programming of what happens in a real robot programming environment. So this work improves the other one [5] and proposes this tool as a more general framework that can be used elsewhere for teaching robotics or making robotics virtual tournaments. The case of RoboCupJunior at the BRO is described here as a use case, nonetheless, it can be used for other purposes, such as training and learning robotics through Educational Robotics. Actually, a Latin American network on Educational Robotics (FOCORE) intends to use this framework in a broader project for bringing Educational Robotics through the Internet to excluded individuals from the region [6].

2 Background Theory

Despite the visible expansion in the usage of technologies in society's daily life, mainly in the educational field, this growth happens unhurriedly. Although many schools have introduced new technologies, many teaching and learning habits remain unchanged. Classes in science and technology laboratories follow a strict pattern, with predetermined classes where students follow a linear approach to learning, instead of exploring concepts [29]. In counter-position, according to Miranda [30], there is an interest to insert new Information and Communication Technology (I.C.T.) in the academic field, aiming to improve the learning environment in the most productive way for students. Mainly, technology has been used in virtual classrooms as a way for teaching [31]. Hence our greater proposal should be a little discussed regarding its pedagogical aspects in light of these technology-assisted learning. In fact, on previous work we provided a critical reflection on the contributions of educational robotic, either as an active methodology for different purposes [4, 26] or helping as a technology-based method in the teaching-learning process in public schools [32]. Besides, Educational Robotics, which is one of these technologies used for learning has recently expanded to a whole new field of research in Science. So a brief introduction to this pretty new field with its main theories is necessary here.

2.1 Issues on Educational Robotics

Educational Robotics and its related methodologies have originated from Seymour Papert's work. He is a researcher from the Massachusetts Institute of Technology (MIT), who has been a pioneer in the creation of technologies for the use of robotics for educational purposes [33]. Papert created a programming language named LOGO, at first used to program geometric shapes of a turtle on a computer screen. He then built a mechanical turtle to allow a more realistic interpretation for the children. From his paradigm, Educational Robotics has evolved and been settled as a complete methodology aiming to present challenges, constantly, to students giving incentives to reflect and formulate ideas on how to solve such obstacles. According to Miranda [30], there are several ways to apply educational robotics in schools, and a proposal is that a robotics class can be divided into four moments:

1. The problematization: a moment in which the challenges are presented to the students. The problem and the knowledge to be worked on are exposed, which may involve multidisciplinary or social themes, compatible with the level of knowledge of the group of students.
2. Exploring potential solutions: step at which students reflect possible solutions to the problem, sharing their ideas with other students, and discussing the best way to solve the challenge.
3. Development of theoretical solutions: after preparing the solution proposal, this moment is to put into practice what was discussed, by assembling robots and coding algorithms for the created prototype.
4. Analysis of results: it is the moment when it is evaluated whether the solution they developed solves the problem presented. In this stage, the student reflects, alone or in a group, if he has reached the objectives or if it is necessary to return to the second stage for a new interaction.

Throughout the search for solutions, tutees test their practical knowledge, providing logical thinking and theoretical comprehension stimuli, denoting the protagonism of students in the formation of information [27]. Actually, Zilli [40] reports that educational robotics develops, in students, skills such as logical reasoning, interpersonal relationships, use of concepts learned in different areas of knowledge, research, understanding, critical capacity, problem-solving schematics through mistakes and successes, representation, and communication, among others.

2.2 Programming Environments for Educational Robotics

The use of virtual environments is a reality in several fields, including virtual laboratories for education in science, technology, and engineering [35, 36]. Mainly, the use of this technology has been applied to assist students in the visualization of concepts related to mechatronics and to provide immediate graphical feedback during the learning process [37]. Modular interactive teaching packages called Virtual Learning System (VLS) have been proposed to be used by people with little prior computer science knowledge in the field of Mechanical and

Manufacturing Engineering [37], and Mechatronics [36]. Learning evaluation is generally done using laboratory reports, questionnaires, and quizzes, such as kahoots [15]. Nevertheless, visual interaction is a key in these systems and is still considered a challenge nowadays, because of the necessity of an online, real-time system, to get full visualization and realism in virtual reality interactivity.

Hence, we notice that, in these contexts, educational robotics goes beyond the real environment, standing out from the hardware components. In other words, a robot must be visualized not only mechanically, with its components originating from a robotics kit, but one also should be able to see its presence in software. A robot is generally coupled with a programmable logic controller, which allows the editing of its commands by coding algorithms [27].

Thus, there is a need of using programs that assist in the coding of the robots' algorithms, and this software, which can be called the programming environment, allows commands to be written (or structured) and compiled for the device's native machine language. Besides, these programming applications must also enable the compiled codes to be sent to the assembled robot, which must be able to execute the commands [19].

In the Educational Robotics context, there are two types of categories for the programming environments. Web environments have a functional architecture that in general follows the client-server model and the computational cost is requested from the server and not from the local machine, allowing any device connected to the web via a browser to be used to program robots. The second category is the desktop environments, in which one needs to install the software on the local machine [19].

On the basis of the platform proposed in this work we make use of W-Educ [19], which is a complete web environment whose goal is to assist the teaching-learning process of Educational Robotics activities. Namely, it allows several robotic platforms to be programmed with the use of a single language, called R-Educ [41]. The latest version of this language is the one currently in use in our solution. Through R-Educ, our users are able to write programs in Portuguese, that are later translated, via W-Educ, into C# code that is internally responsible for running the simulations.

2.3 Simulated Learning Frameworks for Robotics

Simulators are computational systems that mimic the behavior of some real event with which users can interact [42]. Nevertheless, such behaviors are simplified when compared to the real events and these simplifications are acceptable or even desirable depending on the specifics of the system being considered [16]. They have the advantage of replacing the need for some physical elements required to perform some desired activity, as well as aiding the learning process by reducing obstacles present in real-world situations that obstruct experiments and tests. Once we are confident that the simulation results accurately and precisely reproduce the outcomes of analogous experiments with real hardware, exploring the parameter space can be significantly easier and faster than performing similar experiments in reality.

A simulator applied to the field of robotics is a software program capable of virtually reproducing the version of one or more real robots with their models and behaviors [1, 38]. These simulations allow the user to have contact in real-time with the virtualized environment, and to have the possibility to perform actions similar to those that could be done in a real space [25]. It commonly provides development flexibility and extensive testing capability of the algorithm under different environments and many robot configurations [12], facilitating the implementation and testing of specific situations, boosting progress, and accelerating the algorithm development cycle [17].

With the use of simulators for robotics, events and physical limitations that happen when using real robots can be discarded. According to Pedrosa [1], such factors are:

- Easy testing of new algorithms, as there will be no time wasted collecting the robot from the test area and subsequent connection to a computer to change the code;
- Prevent damage to the actual robot components at the time of the tests, since they will not be repeatedly used;
- They usually are not as expensive as real robots;
- The batteries of the real robots are preserved since they are not used in the tests;
- Ease in the development and testing of new robot models;
- Different components can be easily added or removed for testing;
- Ease the tests of several robots, as in a real environment the test group may need to have multiple prototypes;
- Ease in the creation of the environment that will be used by the robot, since in the simulation equipment parts can be easily added, while in the real environment they need to be transported and organized.

We notice that the use of simulators, despite all its advantages, does not provide students with the chance of developing motor coordination and stimulating creativity necessary for the moment of physical assembly. Although not always offering the best simulation of reality, efforts are needed to create more accurate models and behaviors for simulators [1].

3 State of the Art

A large number of works can be found in the literature describing simulators for Robotics [21–24, 28]. Out of them, we selected the works that resemble the sBotics simulator main characteristics to compose the state of the art, that is, we focused on specific simulator projects for Educational Robotics. Some of which are associated with the team's experience in Educational Robotics, which served as the basis for determining the presence of some features and characteristics on this project. Furthermore, all of these selected simulators are in the class of deterministic simulation models, where there is no variation in the environment caused by the simulator [18]. In this way, they might be able to do a good job of simulating what is expected, however they fail to simulate the unexpected variables of the real world [12]. We notice that the better the model of the simulated environment the fewer changes are needed to migrate the structure of the algorithm from the simulated robot to the real robot.

3.1 S-Educ

S-Educ is a simulated environment specially developed for teaching in educational robotics [25]. It is part of the group of projects developed by the Natalnet Laboratories at UFRN, in the Northeast region of Brazil, that aim to improve teaching through educational robotics. This tool enables the assembly of the virtual robot, offering options for construction, such as actuators, sensors, and various parts. As well as robots on different bases, like a car-type robot with wheels or tracks.

In addition to the simulation environment, it has a programming environment enabling visual or textual programming. Another feature of S-Educ is the possibility to send the program developed and tested in the virtual robot to a real robot. The programming of a robot is done by using the RoboEduc programming environment [26]. Although they work independently, there is a partnership between both development teams.

The S-Educ project started with the proposal of simulating environments in two dimensions, however, the evolution of the system has also made possible more realistic three-dimensional simulations. S-Educ was developed aiming to be simple and easy to use, as it had mainly as it had mainly children as its target audience. Even so, it is also capable of serving more complex activities, as it allows multiple robots. It has also been used as a tool for classes on programming languages taught by their authors [25].

3.2 Open Roberta

Open Roberta Lab is a programming environment that enables children and teenagers or any other age group to program different types of robot or micro-controller systems (e.g. Lego Mindstorms, Arduino, and many others) [9]. A major goal of the Open Roberta project is to work on a software platform dissemination for a scalable, education-friendly, cloud-based, and open source.

The Open Roberta laboratory uses the approach of visual programming, making it more accessible, especially for beginners with no experience, to learn how to code. As a cloud-based programming environment, no installation is needed as it only requires a web browser. As of version 1.3.0, Open Roberta Lab started offering a simulation environment for the codes to be programmed in it. The simulation takes place in a two-dimensional environment, making it possible to program a robot with wheels and two motors. Besides, the available robot has ultrasonic, touch, and color sensors. The return values of each sensor can be verified through codes and are presented for viewing on a virtual display.

3.3 USARSim

USARSim - Urban Search and Rescue Simulation [12] was created both as a research and an education tool. It is, in essence, a robotics simulator for search and rescue missions. Initially, it was focused on simulating wheeled robots, but with the constant growth of its userbase, it has several options as aquatic robots, humanoids, and more customization options through components for the assembly of virtual robots, such as actuators and sensors.

By using this simulator, the user can select ready-made scenarios or assemble his own, besides also allowing the program developed in the simulator to be sent to a real robot. The focus was initially on competitions, such as the RoboCup Rescue Simulation League.

3.4 Robot Virtual Worlds

Robot Virtual Worlds is a simulator that allows students to learn how to code. It simulates two and three-dimensional environments with models of commercially better-known robots, such as Vex, Lego, and Tetrix [10]. The language that is used to program these robots is RobotC, which is also capable of being compiled and sent, without changes by the corresponding real robots. This tool, unfortunately, is not freely available to the public, requiring a paid subscription for its use.

3.5 Virtual Laboratory for Robotics (VLR)

The Virtual Laboratory for Robotics (VLR) [34, 35] possesses all the necessary features of a virtual laboratory: user interface, simulator, and visualization. It was developed to prove the feasibility of the concept and make a step towards full e-learning in technical disciplines. Mechatronics and Robotics are understood as a symbiosis of Mechanical and Electrical Engineering.

The tool uses laboratory exercises increasing the complexity of a potential e-learning system, which is used as basis for developing the software-based laboratory exercises. The tool substitutes the physical lab by emulating its full dynamics. This includes mathematical modeling of differential equations to calculate dynamics and also providing data that would be measured on a physical system, coming out with a complete simulated system.

4 The sBotics Proposal

The aforementioned state of the art simulators have unique characteristics. However, as a common feature all of them allow the programming of one or more real robots. Nonetheless, most of them are built for the training of programming itself without the need of a real robot, not attempting on making the environment variables more realistic (physically based). Thus, running a program

execution in simulation twice and with the same parameters values produces exactly the same simulated robot motion. Notice that this would not be exactly true in a real robot, due to problems as slippery ground and others.

On the other hand, the simulator proposed in this work does not allow programming real robots in itself. That is, by default the platform works only with gamified robots that exist solely for the developed platform. However, the use of real robots can be accomplished through the use of some other tool. Furthermore, as a contribution of this work, sBotics reproduces near real-world problems in its arenas and robots by randomizing errors in the environment and sensors variables a little differently for each execution. This makes the results for each running different from the previous ones even using the same environment and robot variables values in different runs.

Notice that this always happens in a real environment where the user faces the problem of having the same program producing different results due to the environment and sensor noise, which produces systematic and non systematic errors [39]. So this feature approximates the user of the real running, and, once there, he/she will have fewer issues to be solved since he/she has already dealt with most of the problems in the simulator. Besides, as said above, the programming carried out in the simulator in the languages BlockEduc [41] and R-Educ can be used in real robots, using the W-Educ tool, which are by definition outside this proposal context as they are previous developments
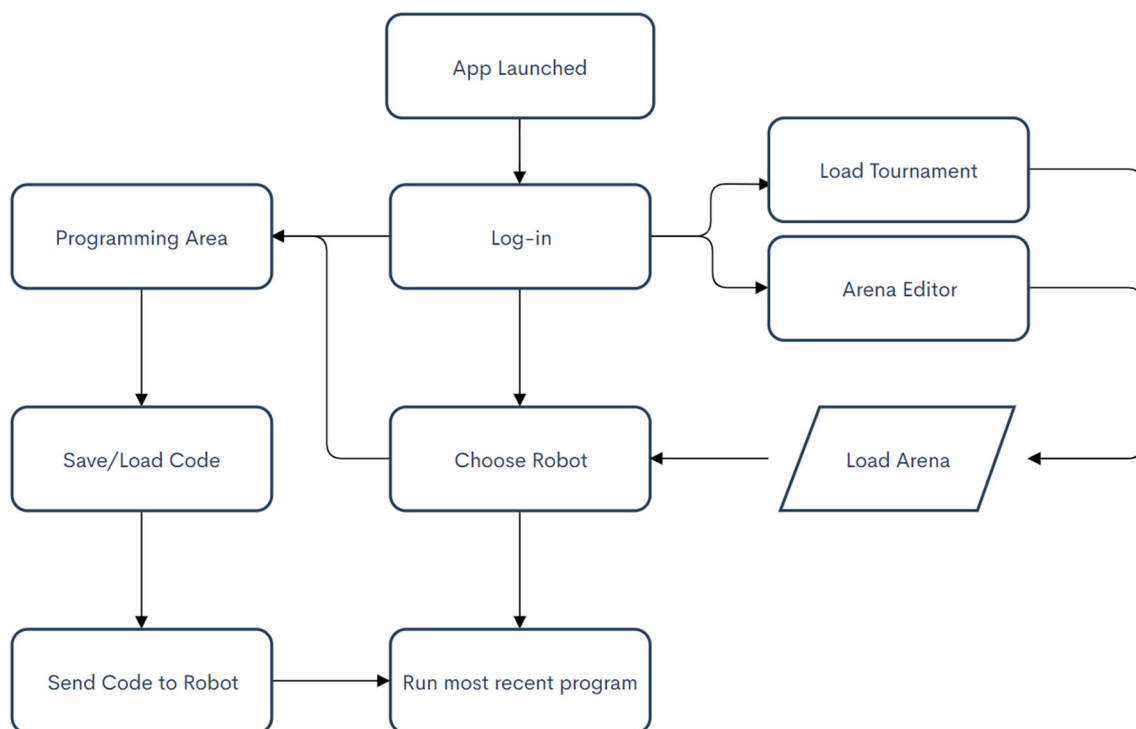

Fig. 1 Flowchart of the platform's usability

[19]. Nevertheless, these tools can be used together, in the same computer facilitating tests in real robots. The details of our proposal will be presented in the next paragraphs, introduced by Fig. 1, that shows all of the possible user interactions within the simulator and can be used to better illustrate its usability.

There is a variety of ways to create a simulator that demonstrates some desired motion behavior. Some of them are free and other ones come as packages of some authoring tools. In this work, we first need to create an environment. Junaid Kayani (https://readwrite.com/author/junaid-kayani/) compiled seven types of high-level software tools that can be used to create realistic 3D outside environments: Vue d'Esprit, Terragen, DEM Earth, World Machine, Large 3D Terrain Generator (L3DT), TerraRay, and Bryce. Another branch of low and high-level tools, at least some 20 of them (https://mashable.com/2008/07/10/tools-for-3d-creation/), exists that is suggested for creating 3D graphics and environments. However, we are interested not only in the environment and robots' construction itself but also in controlling the animation of the created environments and robots with commands programmed by the students and with the possibility of causing disturbances on both. This suggests that using a tool based on low-level computer graphics techniques, for gaming, would be a good idea. Actually, in the approach employed here, we make use of a game engine, as this kind of tool offers much more resources and tutorials for developers. The chosen engine is the Unity3D, one of the most used in the development of 3D animation tools and games [7].

For the communication with the W-Educ platform, an API (Application Programming Interface) is used [8], with the side-effect of requiring an internet connection for programming. However, considering that requests to the API only occur in relevant instances, this connection does not necessarily have to be reliable or quick, and it can still be accessed in remote communities across the country.

Lastly, for user programming, we have agreed that a compiler would have to inject the written code while the application is running. This is a more challenging task than it was primarily considered to be, which is solved in this work through an authentication workflow after opening the simulator, to be discussed later. Therefore, with the methods and the tools chosen, the development of the program to implement the devised architecture could start.

With the above ideas in mind, a system architecture was devised for sBotics that can be seen in Fig. 2. It has three main modules: the *Scenario Editing and Generation* that will be further detailed, the *Programming Interface*, and the *Testing Area* that will be explained next. These modules interact with each other seamlessly through the graphical interface giving the impression that the whole application is a single entity.

4.1 Testing Area

The robot selection and construction is possible in the testing area, where it is allowed to change the robot actuators, the robot itself, and the environment by adding checkpoints to the arena. Here it is also possible to execute the program and adjust the initial and current positions of the robot and components in the environment. An example of the main differences between each possible robot configuration is shown in Table 1.
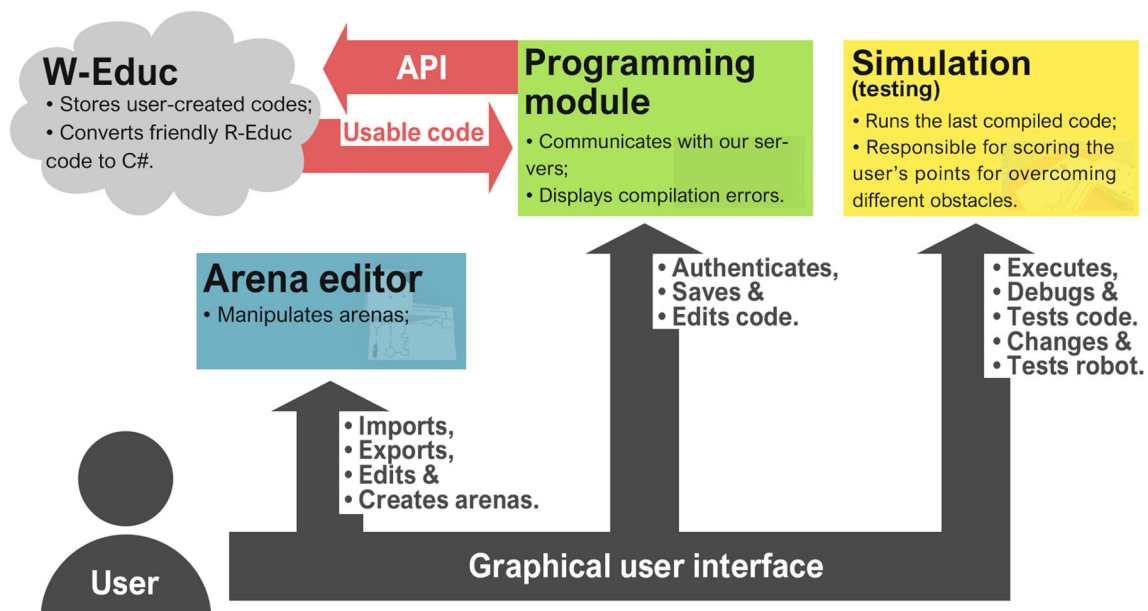

Fig. 2 Each section with its functions and interactions

Table 1 Comparison between robots

Robot	Sensors (input)			Output		Sensor refresh rate
–	Color	Ultrasonic	Touch	LED	Buzzer	–
	3	3	1	1	2	48ms
	6	4	1	1	2	92ms
	5	3	1	1	2	76ms
	3	2	1	1	3	32ms
	6	3	1	1	1	76ms

The difference in each robot's mass is important to the user, to be present in a challenge, because in order to have accuracy and precision in a machine one has to sacrifice speed, and vice-versa, in general. With this simple way for robot selection, code execution, and programming experience, paired with a complete arena editing module, it is possible to emulate many robotic competitions and also being a powerful tool for teaching and learning in schools and colleges alike.

4.1.1 Sensors

The robots in the simulator have virtual components that work as input or output to user-written code. These segments vary from simple LED lights for outputting data to complex ultrasonic sensors, capable of capturing the distance of any given object in its immediate periphery. The behavior of the sensors in the platform is efficient, light, and easy to understand, as it uses simple physical concepts to produce the desired real-world operation. The first category of sensors to be discussed are the most used, light-capturing ones. These being the light and color sensors,

very similar due to having related backend mechanics, but very different due to their intended use and return value types.

A color sensor can be set as a single-pixel camera, with the capability of capturing the pixel related to a surface in front of it in the simulated 3D environment to process the perceived hue, checking whether or not it's superior to the color-calibration value. For instance, having a color with the values of Red: 200, Green: 185, and Blue: 26 under the sensor will make it check if each value surpasses the aforementioned calibration value (by default 63), this will return "true" on red and green and return "false" on blue, prompting the user the color yellow as it is graphically the mixture of red and green.

The light sensor, on the other hand, simulates a 5x5 pixel camera, whose values are greyscaled using Rec. 601 Luma [20] and its light value stored and averaged to return as an integer to the user. Using the same example as above, after greyscaling that same pixel the sensor would have to process 24 more pixels to create an average to send to the user the luminance value between 0 (complete darkness) and 100 (absolute brightness). The C# code employed to

create this grayscaling behavior for each pixel can be seen in Listing 1.

```
light = 0;

for (int i = 0; i < 5; i++) {
    for (int j = 0; j < 5; j++) {
        light += 0.299 * image.GetPixel(i, j).r;
        light += 0.587 * image.GetPixel(i, j).g;
        light += 0.114 * image.GetPixel(i, j).b;
    }
}

lightness = 4 * light;
```
Listing 1 Rec. 601 Luma for light sensor

The next sensor in the category of sight is the ultrasonic sensor, capable of detecting objects at a certain distance. Its functionality consists of a series of ray casts, as shown in Fig. 3, in a triangular shape (mimicking the propagation of sound in an open space) that returns a value once one of the rays hits an object. If any given ray goes too far its return is discarded considering that in a real-world scenario simple ultrasonic sensors are not capable of perceiving distances greater than a couple of meters.

The last category of sensors, although fixed regardless of the robot, is crucial for gathering its surroundings. These components are mostly simple data that can have enormous implications in the user's code. This category includes touch sensors capable of perceiving whether or not an object is in contact with said part; a compass capable of returning angular positions of the robot; a heat sensor able to get the temperature in the robot's virtual setting; and an inclination sensor to receive the perceived inclination of the robot. Notice that other sensors could be defined and used as necessary for some new challenges.
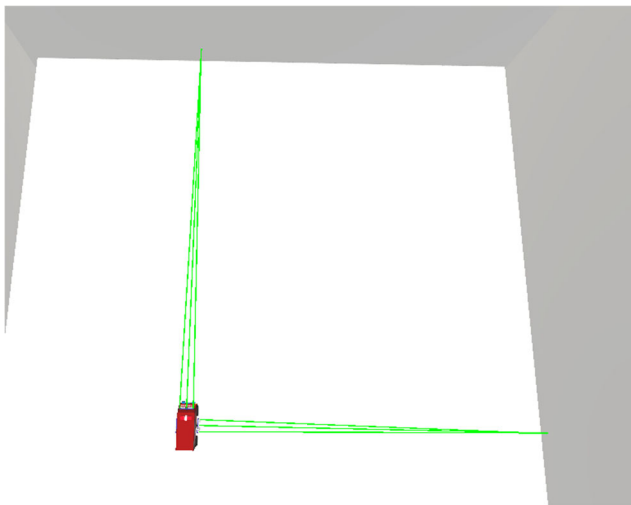

Fig. 3 Ultrasonic's ray casts

4.1.2 Outputs

The next category of elements is the output components. It is possible to encounter in this category motors, buzzers, RGB LED light, and three virtual LCD screens. These output components when programmed by the user can respectively move the robot, play a sound, emit light in any given color, and display some text. Output elements are useful for the programmer to accompany the progress of their program without relying on vague ideas of the code's execution.

By default, each robot has two motors at its bases, and each engine can be manipulated together or independently through its specific functions. In every movement function, it is necessary to determine the force to be used. Forces can vary from -300 to 300 for movements outside the robot axis and from -1000 to 1000 for movements on the rotation axis.

4.2 Creating Arenas and Scenarios

There are many different scenarios in which this simulator can be used regarding educational robotics and tournaments. One example is the usage of the platform to teach geography by displaying a map and demanding a student to go from place A to place B on it (Fig. 4).

Such flexibleness is achieved by an intricate arena creation system, allowing the creation of complex arenas for many different uses.

4.2.1 BRO Arena Generation

This is the main scenario of the simulator and the one which has the most dedicated features. The Brazilian Robotics Olympiad is an annual Brazilian nation-wide robotics tournament for K-12 students based on the RoboCupJunior Rescue Line A [3], and it is very important for hundreds of thousands of young Brazilians all over the country. Although a G20 world power, Brazil is still a country very much influenced by its staggering poverty and inequality rates, which impose challenges for Brazilians mostly in rural parts of the country or in poverty-ridden areas to pursue the field of robotics and participate in the BRO. This simulator intends to provide many Brazilians with that opportunity that would otherwise take years to reach such regions.

The RoboCupJunior Rescue Line A arena consists of a path that the competitor's robot has to follow to find a room with victims (named the rescue area) where the robot has to enter and save. An BRO-type arena is generated by procedurally filling a 6x3 grid with pre-defined tiles, that algorithmically find where to place the next piece until the end of the grid, where the rescue area can be found.

By providing a coordinate for each tile slot in the grid it is possible to define its limits as the zeroth and fifth columns

and the zeroth and second rows. Starting at the coordinates (0,0), the generator has two options: going forward or going to the right. In a coordinate such as (5,0), the algorithm can only go upwards to the final coordinate (5,2), which always faces the rescue area's ramp. In other words, by cleverly storing the last movement by the script and defining its boundaries it is possible to generate new tiles in different directions while always going towards the final coordinates. This is better illustrated in Figs. 5 and 6.

4.2.2 Arena Editor

Every arena can be edited without limitations. That is to say that everything the procedural generator can generate the editor can recreate. This is important for the adaptability of the platform, which can be used in all sorts of robotic

competitions. One can, for instance, easily create a maze for a labyrinth competition (seen in Fig. 7) or a blank arena for image insertion (discussed in the next section).

It's possible with the editor to move a purple cursor throughout the grid and manually choose which tile to fill that grid coordinate. There are advanced settings to set the robots competitors can use, simulated time of day, the number of victims to save, and many others.

Many possibilities come with the editor, such as adding heat-emitting objects (such as the litten candles seen in Fig. 7), markers that reward or punish the user's robot upon contact, maze walls, and many others. It is also important to note that any created arena can be exported as a code and be imported into another computer.

4.2.3 Image Insertion

There is also the possibility of adding images to an arena, which can vary in size, from a single tile to the entire grid. There is also the possibility of adding maps, geometric shapes, artworks, and any picture. Such feature makes the arena useful not only to robotics competitions but also to schools, that can teach many subjects traditionally thought of as incompatible with the field of robotics (Fig. 8).

4.3 Programming Environment

The user's programming interface, shown in Fig. 9 is made to interact with W-Educ, which is another platform responsible for storing and source-to-source compiling codes [19] through an application programming interface (API), thus requiring authentication after opening the simulator. Within the coding tab, it is possible to code in three levels of abstraction: in an overly intuitive programming language (called R-Educ), in BlockEduc (R-Educ version in Blockly [43]), and C#. All languages, except for the latter (which is already in the target language),
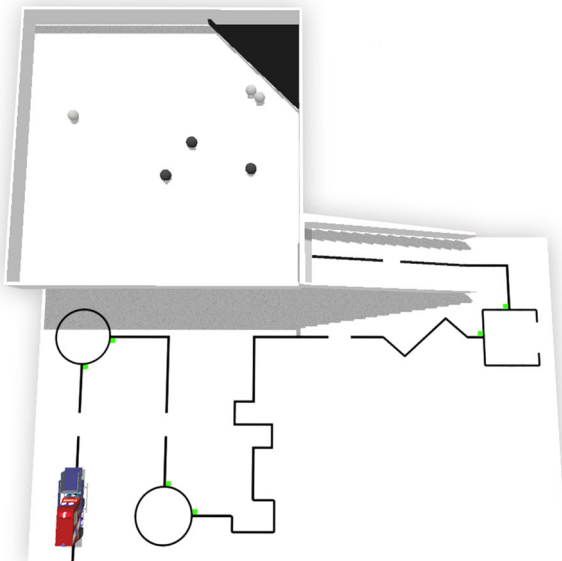

Fig. 5 An BRO-type arena

Fig. 6 Figure 5 with a grid, coordinates and an example tile showing its generation procedure
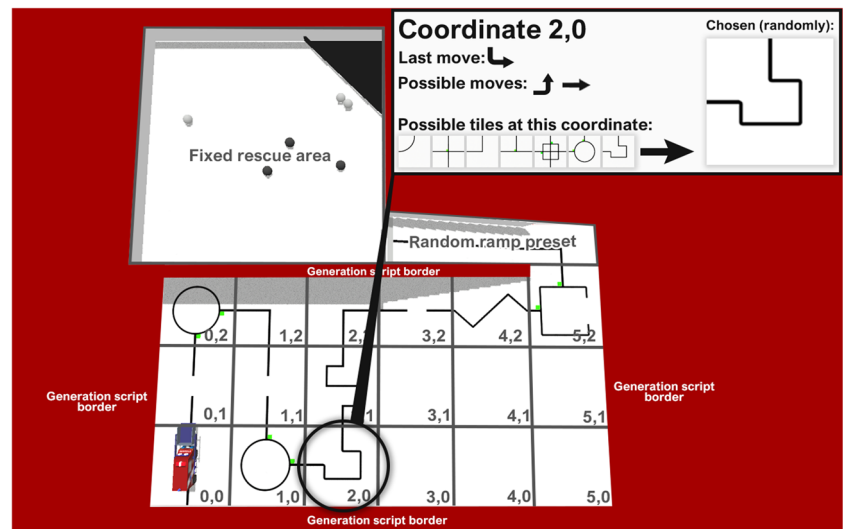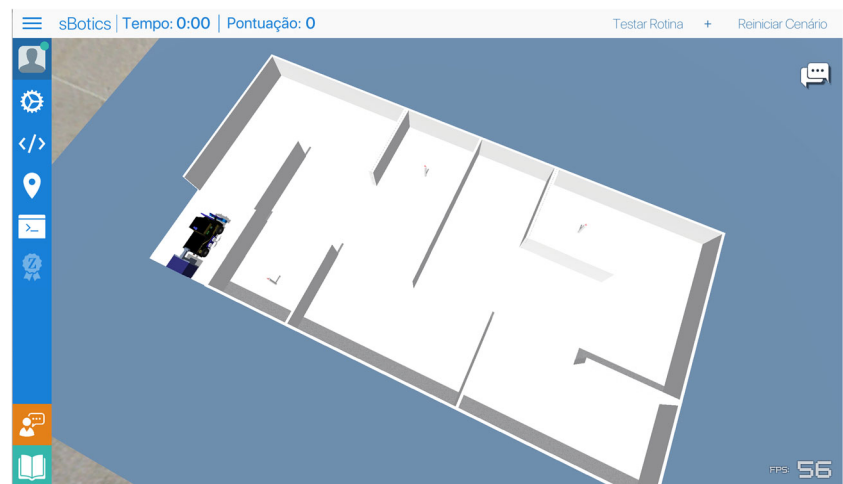

Fig. 7 A labyrinth-like arena

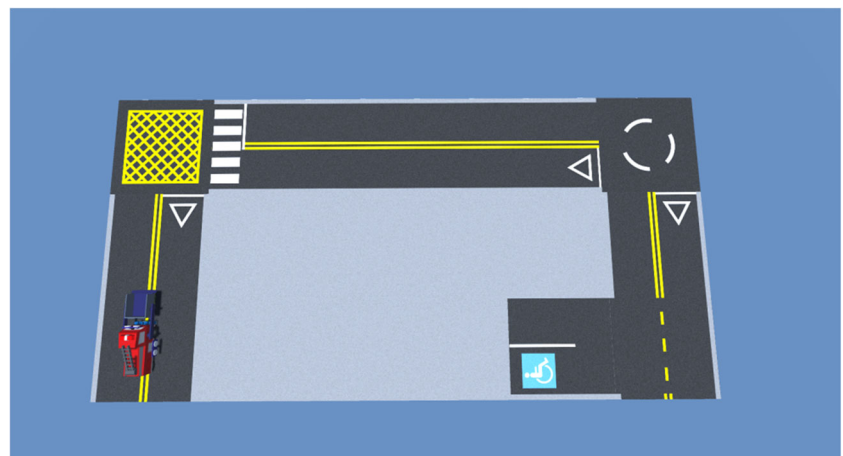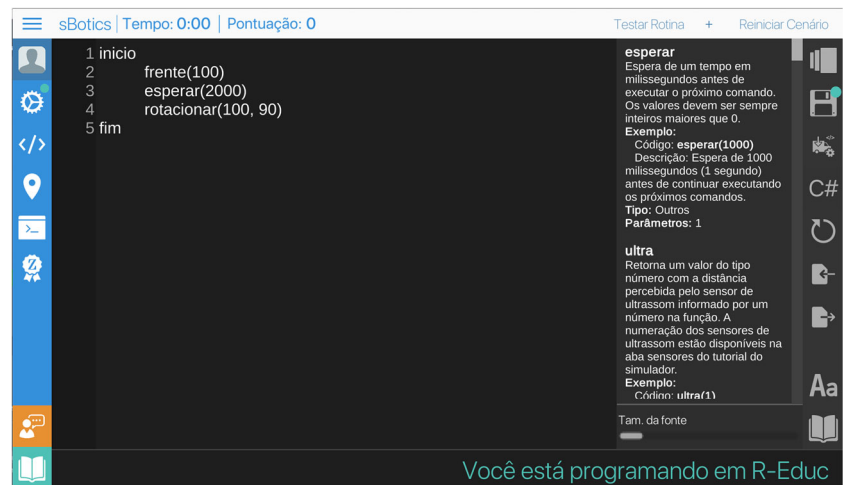**Fig. 8** sBotics being employed to create a virtual road system to teach students traffic laws

are source-to-source compiled in W-Educ to C# to be injected in real-time in the virtual robot using a native C# class called CodeCompiler.

In the programming interface, a guide with the functions/methods documentation for the language chosen for programming is available. One can access all previously saved codes and save new codes. Export files and import edited codes in external editors with automatic updating with each external save. And also compile the code and send the code to the robot.

```
begin
    while (true) do {
        if (light(1) < 20) then {
            right(100)
        } else if (luz(2) < 20) then {
            left(100)
        } else {
            front(100)
        }
    }
end
```
Listing 2 Algorithm to follow a line in R-Educ, origiinally in Portuguese and translated for a better understanding.

Besides, it is possible to convert between the programmed codes from BlockEduc to R-Educ and from R-Educ to C#. This way users can learn to program in R-Educ by converting their written codes in BlockEduc or learning C# from their writings in one of the other available languages. Figure 10, Listings 2, and 3 show the codes to follow the line initially written in BlockEduc and automatically converted by the system to R-Educ and later to C#. Notice that in Algorithm 3 the code generated in C# does not allow the definition of traditional classes, constructors or methods. This language in sBotics allows only the use of

lambda Actions and Functions, since the injection of code at runtime occurs within the scope of another method of the LoaderClass. Despite this, the user has full access to the methods of the BotController class instantiated as bc. These methods correspond to the reading, writing, and movement actions of the robots.

```
while (true) {
    if (bc.lightness((int)(1-1)) < 20) {
        bc.onTF((float)-(100), (float)(100));
    } else if (bc.lightness((int)(2-1)) < 20) {
        bc.onTF((float)(100), (float)-(100));
    } else {
        bc.onTF((float)100, (float)100);
    }
}
```
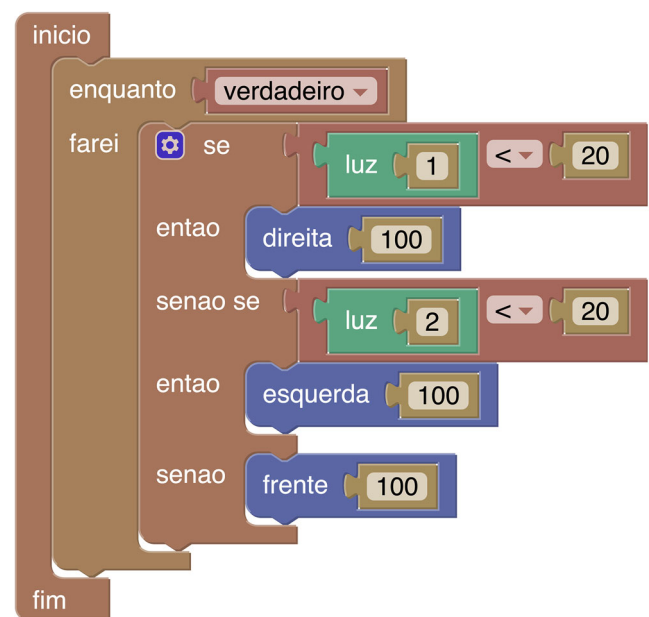**Listing 3** Algorithm to follow line in C#

Fig. 10 Algorithm to follow line in Blockeduc (in Portuguese language however that can be translated)

4.4 Environment Randomization

With environment randomization (ER), we are able to create a variety of simulated environments with randomized properties and train a model that works across all of them. Simulators often struggle to emulate real-world scenarios such as components' defects, assembly nuances, changes in temperature, lighting, and many other circumstances that cause different outcomes, leaving the impression to the average user that simulators are always a perfect environment and not necessarily applicable for situations in a non-simulated setting. This lack of realism creates a dilemma in simulated environments, where it's desirable to achieve acceptable levels of credibility with perfect functioning robots while maintaining mechanisms to randomize difficulties that robots face in the real world.

With this predicament in mind, the application uses several techniques to achieve ER, such as the employment of a noise-like texture in the arena, dropping the robot from a small variable height, using the game engine's physics to drift the robot ever-so-slightly from its path, and so on. Table 2 shows the average values for 100 readings of the distance between an object at the arena coordinates (0,3) and the robot, at 0 and 2 seconds. This in turn causes enough entropy in the perceived distance for the sensors to be considered realistic in our standards. Notice that the initial distance deviates approximately 0.43 distance units since the robot, despite starting automatically, is not spawned exactly in the same position.

4.4.1 Techniques for Randomization

For the implementation of environment randomization, the following techniques are employed:

- Application of a noise-like texture to the arena's surface: the arena is not completely white, simulating a real one that is not perfectly clean. This influences the reading of the robots' light and color sensors, which, just like in a real environment, will vary minimally with each reading made repeatedly in the same place;
- Random robot positioning: the restart direction in the same predetermined position is not always the same, the robot is not positioned directly in the arena but a few distance units away from the ground, suffering uncontrolled physical effects in the fall;

$$direction = presupposedDirection \pm physicalInfluence$$

- Robot rotation functions employ approximations in order to mimic motor clearance;

$$rotation = presupposedRotation \pm randomRange$$

$$randomRange \in \mathbb{R}, randomRange \in [-1, 1]$$

- Wait functions deviate slightly in order to simulate batteries at different charges;
- Every robot has a minimum variable weight defined by the formula:

$$calcWeight = \frac{(15 - (weight * randomRange))}{10}$$

$$randomRange \in \mathbb{R}, randomRange \in [0.95, 1.02]$$

Where *weight* is the fixed robot's mass.

4.5 Controversy on Environment Randomization

Some of the users have negatively reported about the environment randomization, as in general the expected behavior obtained from other traditional simulators for a run is exactly the same for the same set of variables. That is, if some code produces a result, it is supposed to repeat the same one upon a restart if the system and environment variables do not change. Notice that this may be true in a computer system however it does not happen in the real world due to sensor and other errors. This disdain is found amongst the simulator's student community, that like in the real world, have to test their robot multiple times and create a program that foresees different situations and adversities. However, we noticed that the same feeling is not shared by educators, as this realism feature is very useful for creating easily transferable codes from the platform to real similar-working robots. Actually, this feature makes the tool a little more close to reality.

Even with the negative feedback from a few students on the userbase, the appeal of keeping environment randomization still outweighs the valid criticism of not being simulator-like. Not only is the platform not simulating a specific robot for it to be precise (resembling more a game that intends to emulate robotics), but every obstacle created by ER can be circumvented with better-written code, and the depth it adds to the platform is important, as it allows

Table 2 Evaluation of the robot's position

Reading Order	Average	Standard deviation	Estimate reading after 2 seconds
First reading	173.4349	0.4294428973	173.4349 ± 0.4294428973
Second reading	101.9591	0.9449986318	101.9591 ± 0.9449986318

users to write more intelligent programs that better adapt to several varieties of obstacles.

5 Experiments and Results

In order to validate our proposal and demonstrate the usefulness of the framework, we divide the experiments and results Section into two main parts. At first, we discuss preliminary experiments and results obtained with the previous version [5]. Then the second round of experiments shows the numbers on the usability of the system by students and competitors and its applicability on the RoboCupJunior Brasil, inside the Brazilian Robotics Olympiad (www.obr.org.br).

5.1 sBotics I - The Beta Version

The beta version of sBotics was released after two years of development, even though it had not reached the level of excellence planned for its first release, because it was a need. This in turn was not an obstacle for its initial validation as a useful tool in robotic events in Brazil, such as the Brazilian National Robotics Exhibition Fair (MNR) and the Brazilian Robotics Olympiad (BRO) itself. The BRO has two modalities, theory and practice. In 2019, it had over 204 thousand direct participants from all Brazilian states with more than 5000 teams competing in the practical modality in the country. sBotics was designed mainly for the final short-course for the theoretical modality of BRO. This is a practical programming course that happens in parallel with the the practical BRO modality, which is the national RoboCupJunior finals.

The first release paper on sBotics framework was presented at the Workshop of Robotics in Education [5], where an overview of its functioning and the experimental setup for its use in the above short course on Educational Robotics, for teenagers, is described. Basically, the platform starts with the simulation, with a top bar where it is possible to execute programs and a sidebar where settings, programming and robot placement are enabled. Using the computer keyboard and mouse the user can manipulate the camera's angle and position, moving around the arena to familiarise itself with the idea of a virtual setting. One of the tabs in the sidebar is the settings, where the user can manipulate the time and format of the arena, their preferred robot and actuator. Each robot offers different types of sensors for the user to manage at their own will. For the programming tab, authentication is required for the integration with the W-Educ platform. After performing all the procedures, the user is presented with a black programming canvas, where it is possible to experiment with the several available functions in the R-Educ

Fig. 11 Simulator being used to instruct the best scoring students of the practical component of the BRO

programming language, offered by the aforementioned platform. After the programmer finishes its code, it is possible to compile and execute the program, similar to a real robot.
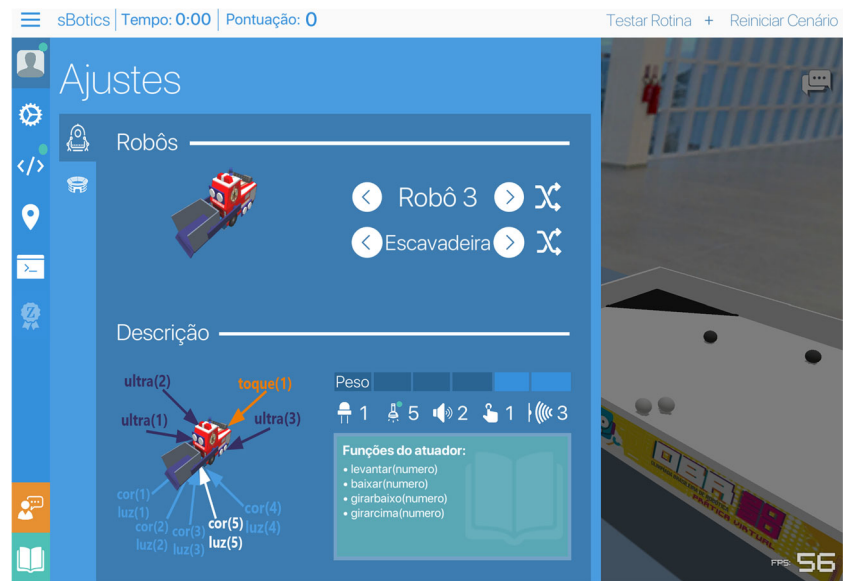
This platform has other possible applications such as the training of judges in the BRO and courses related to the subjects of robotics or programming logic. As said above, in the latest national round of the BRO, the simulator was used to train the best scoring students of the practical modality of the tournament (see Fig. 11).

Besides being limited, the first version version also had a restricted number of functions and it has been substantially upgraded since there in order to reach the current level of usability, and being finally released and used by RoboCupJunior Brazil as pointed next. All of the screens and the algorithms were redone, and the performance enhanced, also becoming a desktop multiplatform system. Nonetheless, we consider that the results of the experiments carried out with this first version were quite satisfactory and drove us to give a continuity on the project.

5.2 sBotics II - Current Version

With the Covid-19 pandemics course, our initial proposal of sBotics was suggested by Robocup Brazil Trustees to be used as the official simulator in order to making it possible to run RoboCupJunior regionals in Brazil in 2020. At this point, as noticed above, we remark that this competition is one of the biggest RoboCupJunior trials of the world with thousands of teams and participants so it provides a good benchmark for testing our tools. It takes place in two phases, one being regionals and another one the finals on Brazil.

For this, we remodeled all of the simulator interfaces following the formal theory on user experience (UX) metrics [44] in order to improve the user experience and to make it more attractive to children and teenagers. Visual

Fig. 12 Robots selection

experience of this enhancement can be seen in Figs. 12 and 13. The simulated arenas have been placed in a virtual copy of a real environment, which is at the Natal Convention Center, overlooking a postcard from the city of Natal/RN/Brazil (Fig. 14). The goal of this is to give a greater idea of immersion in an environment of competition. This can be easy changed for further competitions. In addition, we have created more features for the robots. Actually, the BotController class responsible for the entire operation of the robot comes up from 31 methods in the first release to 127 fully available, nowadays, to the programmer. With constant communication with teachers that are using the tool we listened to their needs and incorporated all possible ones, such as block programming [43] in BlockEduc, a translated block language. The current

version of sBotics can be found at https://weduc.natalnet.br/sbotics/.

With the official launch of the simulator by RoboCupJunior regionals in Brazil, we went from approximately 170 users with active accounts to more than 4,500 accounts. In addition, more than 18,400 clicks were made on the download link of the simulator. According to user reports, many of these accounts are being shared between students in the same class or members of the same competition team for code sharing. Currently, we have a daily average of 10,000 compilations in our servers.

Additionally, for the sake of testing the environment previously to the official stages of the regionals, we have planned and executed 2 warmups with it. These pre-competitions took place on two consecutive Sundays,

Fig. 13 Arena editor

Fig. 14 sBotics main Scene

with registrations taking place on the previous Friday and Saturday of each one. Table 3 shows the number of participants and their engagement on the warmups.

At the end of the first warmup, several participants reported the need for other ones to occur before the official events in order to familiarize them with the new tool. In addition, errors in the scores were reported by them, in very specific situations in which they missed a large number of tests, to be found and corrected. One of the participants reported that:

> The competition system is super dynamic and fluid. (Competitor A)

Another participant said:

> Although it was run virtually it felt like I was in the real competition, the emotion was the same, I was shaking a lot. (Competitor B)

The official regional phase of the competition will initially have 27 state selective events and then the best players will be classified for the national finals. Each of these events will have 3 seats in the final event. Unlike the warm-up event that had a maximum of 147 teams registered, the official event has so far 1162 teams with 3700 students from all over Brazil, and will run virtually.

The use of sBotics has not been limited to competitions with social distance and the interruption of face-to-face classes in schools across the country. The tool is being used to teach programming, control, engine operation, sensors, and other general Educational Robotics applications.

Table 3 Warmup engagement numbers for Brazilian Robotics Olympics

Round	Registered	Competed	Valid scores
1st Warm-up	133	87	84
2nd Warm-up	147	103	102

Among the videos listed on youtube, more than 100 videos with tutorials, courses, and applications of the tool can be found in the classroom for various purposes.

A simple search on "youtube sBotics videos" returns several results on the production of independent videos and tutorials about the use of sBotics by the community, as this one: https://www.youtube.com/watch?v=RWHLKTfAz3g. Another example of more complete tutorial can be found at the independent youtube channel (https://www.youtube.com/channel/UCoeQ3SkVxFAYR0vrIUUpPtw), starting with the number 1 video: https://www.youtube.com/watch?v=qBixpxM2_Mk). The last one is provided by *The Mecatrônica Project*, an independent user. We believe that these several videos plus the current users are the best validation for our work, and we are happy to help Brazilian Robotics Olympiad community to keep competitions up during this hard Covid-19 pandemic time.

6 Conclusion

In this paper, a new Educational Robotics environment with a gamified simulated framework is proposed to fulfill the existing gap between students that have no access to robots and to also diversify a regular educational robotics learning environment. To do that, the Unity3D game development engine is used in combination with the W-Educ environment [19] for programming through the Internet and view the results immediately in a 3D environment shown in a desktop computer.

The RoboCupJunior, which is the practical modality of the Brazilian Robotics Olympiad, is one of the greatest national competitions on the world, with more than 5 thousand teams and about 200 thousand overall (including theoretical modality) participating on it. So the testbed is really hard and the developed framework has passe it up to date.

As a huge contribution to STEAM, the developments of this work provide a new way to expand educational robotics, taking into account the easy access of sBotics when comparing with robotics kits especially on current social distance times. Besides, with sBotics schools with low financial conditions can insert the educational robotics methodology without the need for extra expenses. Since most schools have already computers in their facilities, several public ones in Brazil are already using it.

And even for those who already have materials for mounting or making their robots, it would be a great help in order to save the use of items that can be damaged or for saving energy and other physical resources, in addition to helping with the practicality of tests with robots. For these schools, the use of physical robots would be in the final stage of a robotics class, since the presented

work could replace most of the steps of the lesson in class.

Besides, through the simulator, teachers of robotics will be able to do something that was previously unfeasible for their students, due to the need for physical components, which are homework. Teachers can build a simulated environment, use markers to indicate students' successes, and share the arena with their class to perform the activity at home.

To this end, we believe that the sBotics framework is polished and ready to be used by schools and individuals, to help diffusion of educational robotics in an ever-connected society. It is available at https://weduc.natalnet.br/sbotics/. However, this is not the end of the works on sBotics, since there are still many things for our development team to accomplish, such as enhancing the block-based programming, offline development, user-created robots and an automatic scoring system to gamify even more the platform.

Although the tool is already quite complete, considering that most requests from the community have already been developed, we indicate the need to insert more robot models with different sensor and motor positions. To this end, a module for assembling and modeling virtual robots could be integrated to simulate the completion of the 3rd stage of an educational robotics class, which is the moment of building robots.

Our system downloading data show that it has already been downloaded from several countries in Latin America and also from outside as the United States. Thus, the possibility of using it in different languages such as English and Spanish is also one of our next goals, expanding its use and taking more results as the ones obtained in Brazil for other places.

Another aspect that has to be expanded upon is the impact of the presented platform on humans. As an autonomous and intelligent system (A/IS), it has to safeguard throughout its life cycle human well-being at the individual, population, and societal levels [45]. As a platform developed for education, human interaction optimization and its positive impact on society have been two of the many driving factors for development. Thus, the platform is well suited and its use is secure for literate humans of any age, although further research can be made to explore all possible interactions. Beyond that, compliance with Brazilian data regulations such as LGPD [46, 47] (General Law for Personal Data Protection, in Portuguese) is a priority measure taken by the development team of W-Educ, which currently handles personal user data for the platform. The robot-human interactnion aspects are important, not only for this work but for every platform, although they are not the focus of this paper. This paper has proposed a working framework, whose main focus is the realism of the sensors and environment feedback to the robot actions, which is generally neglected in traditional robotic simulators. Nevertheless, the high-level aspects and related architectures, are drawn as a future study, with which our team will have time to study now.

Besides, as said at the end of the State of the Art Section, the programming carried out in the simulator that uses the languages BlockEduc [41] and R-Educ can be used in real robots only by using a different tool, the W-Educ [19], which is left outside this proposal. So the integration of all of these tools in order to be used together in the same computer and environment is another future work to be done, thus facilitating tests in real robots.

Data Availability No data was produced for (or from) this study. Nevertheless, the sBotics is available at https://weduc.natalnet.br/sbotics/ and the W-Educ interface is at https://weduc.docs.apiary.io/.

Declarations

Conflict of Interests There are no conflicts of interests nor competing interests for this work.

References

1. Pedrosa, E.F.: Simulated environment for robotic soccer agents. Master's Dissertation, University of Aveiro, Portugal (2010)

2. Becker, G.L.: Development of a simulator for an autonomous vehicle (in Portuguese) Final Course Conclusion Work in Computer Science. Universidade Federal de Minas Gerais, Belo Horizonte, Brazil (2010)

3. Aroca, R., Pazelli, T., Tonidandel, F., CA Filho, A., Simoes, A., Colombini, E., Burlamaqui, A., Gonçalves, L.: Brazilian Robotics Olympiad: A successful paradigm for science and technology dissemination. Int. J. Adv. Robot. Syst. **13**, 1–8 (2016). https://doi.org/10.1177/1729881416658166

4. Aroca, R.V., et al.: Increasing students' interest with low-cost cellbots. IEEE Trans. Educ. **56**(1), 3–8 (2013). https://doi.org/10.1109/TE.2012.2214782

5. do Nascimento, L.M., Neri, D.S., do Nascimento, T., Yanaguibashi, E.A., Sá, S., Gonçalves, L.M.: sBotics: simulation applied for the practical component of the brazilian robotics olympiad. In: 2019 Latin American Robotics Symposium (LARS), 2019 Brazilian Symposium on Robotics (SBR) and 2019 Workshop on Robotics in Education (WRE), Rio Grande, Brazil, pp. 487–491 (2019). https://doi.org/10.1109/LARS-SBR-WRE48964.2019.00092

6. Fernandes, C.C., et al.: Demystifying educational robotics with FOCORE: from very low cost software and hardware technologies to the development of new methodologies and curriculum for continuing teacher education and teaching of brazilian basic education students. In: 2020 Latin American Robotics Symposium (LARS), 2020 Brazilian Symposium on Robotics (SBR) and 2020 Workshop on Robotics in Education (WRE), Natal, Brazil, pp. 1–6 (2020). https://doi.org/10.1109/LARS/SBR/WRE51543.2020.9307032

7. Unity 3D (web site). https://unity.com/. Last accessed Oct, 11th 2020

8. W-Educ API (web site). https://weduc.docs.apiary.io/. Last accessed 2020, Oct, 11th 2020

9. Open Roberta (web site). https://lab.open-roberta.org/. Last accessed Oct, 11th, 2020

10. Robot Virtual Worlds (web site). http://www.robotvirtualworlds.com/. Last accessed Sep, 23rd 2020.

11. Brooks, R., Matarić, M.: Real Robots, Real Learning Problems. RobotLearning. Kluwer Academic Press, Dordrecht (1993)

12. Balakirsky, S., Carpin, S., Dimitoglou, G., Balaguer, B.: From simulation to real robots with predictable results: methods and examples. In: Madhavan, R., Tunstel, E., Messina, E. (eds.) Performance evaluation and benchmarking of intelligent systems. Springer, Boston (2009). https://doi.org/10.1007/978-1-4419-0492-8_6

13. Curto-Diego, B., Moreno-Rodilla, V., García-Peñalvo, F.J. (eds.): Special issue: robotics in education. J. Intell. Robot. Syst. (2016) Vol. 81, Issue 1. ISSN: 0921-0296. https://link.springer.com/journal/10846/81/1/page/1, Accessed on 13-1-2021

14. sBotics site at Natalnet Associate Labs, Federal University of Rio Grande do Norte. https://weduc.natalnet.br/sbotics/ (accessed on January, 13, 2021)

15. Kahoot for Schools - Quizzes and Challenges for teaching interactive lessons. https://kahoot.com/schools-u/ (accessed on January, 13, 2021)

16. Cianci, C.M., Raemy, X., Pugh, J., Martinoli, A.: Communication in a swarm of miniature robots: the e-puck as an educational tool for swarm robotics Şahin, E., Spears, W.M., Winfield, A.F.T. (eds.), vol. 14433. Springer, Berlin. https://doi.org/10.1007/978-3-540-71541-2_7 (2007)

17. de Lima, P.V.S.G., et al.: Improving early robotics education using a line-following robot simulator. In: 2018 Latin American Robotic Symposium, 2018 Brazilian Symposium on Robotics (SBR) and 2018 Workshop on Robotics in Education (WRE), Joao Pessoa, pp. 547–553 (2018). https://doi.org/10.1109/LARS/SBR/WRE.2018.00101

18. Chebotar, Y., et al.: Closing the sim-to-real loop: adapting simulation randomization with real world experience. In: 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, pp. 8973–8979 (2019). https://doi.org/10.1109/ICRA.2019.8793789

19. Thomaz, S., Fernandes, C., Pitta, R., Torres, V., Gonçalves, L.M.: Web-based configurable and multiplatform development environment for educational robotics. In: 2013 16th International Conference on Advanced Robotics (ICAR), Montevideo, pp. 1–8 (2013). https://doi.org/10.1109/ICAR.2013.6766509

20. Gaunt, R.: Color spaces in digital video. United States: N. p. Web (1997)

21. Ajwad, S.A., Asim, N., Islam, R.U., Iqbal, J.: Role and review of educational robotic platforms in preparing engineers for industry. Maejo Int. J. Sci. Technol. Chiang Mai **11**(1), 17–34 (2017)

22. Teixeira, J.V., Hounsell, M.S.: Educational robotic simulators: a systematic literature review. Nuevas Ideas en Informática Educativa TISE (2015)

23. Koenig, N., Howard, A.: Design and use paradigms for Gazebo, an open-source multi-robot simulator. In: 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No. 04CH37566), Sendai, vol. 3, pp. 2149–2154 (2004). https://doi.org/10.1109/IROS.2004.1389727

24. Costa, V., Rossetti, R.J.F., Sousa, A.: Autonomous driving simulator for educational purposes. In: 2016 11th Iberian Conference on Information Systems and Technologies (CISTI), Las Palmas, pp. 1–5 (2016). https://doi.org/10.1109/CISTI.2016.7521461

25. Fernandes, C.C.: S-Educ Um Simulador de Ambiente de Robótica Educacional em Plataforma Virtual. 2013. 83 f. Master's Dissertation, Universidade Federal do Rio Grande do Norte, Brazil (2013)

26. Barrios-Aranibar, D., et al.: RoboEduc: a software for teaching robotics to technological excluded children using lego prototypes. In: 2006 IEEE 3rd Latin American Robotics Symposium, Santiago, pp. 193–199 (2006). https://doi.org/10.1109/LARS.2006.334332

27. Thomaz, S., et al.: RoboEduc: A pedagogical tool to support educational robotics. In: 2009 39th IEEE Frontiers in Education Conference, San Antonio, TX, pp. 1–6 (2009). https://doi.org/10.1109/FIE.2009.5350439

28. Michel, O.: Cyberbotics Ltd. WebotsTM: professional mobile robot simulation. Int. J. Adv. Robot. Syst. (2004)

29. CAMPOS, Rodrigo, F.: Robótica Educacional no Brasil: Robótica Educacional no Brasil Questões em Aberto, Desafios e Perspectivas Futuras. RIAEE, Sã,o Paulo, vol. 12, no. 4, pp. 2108–2121 (2017)

30. Miranda, L.C.D.: RobôFacil: Especificação e Implementação de Artefatos de Hardware e Software de Baixo Custo para um um Kit de Robótica Educacional. Master's dissertation, Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brazil, vol. 1, no. 1, pp. 1–124 (2006)

31. Olszewska, J.I.: The virtual classroom: a new cyber physical system. In: Proceedings of IEEE World Symposium on Applied Machine Intelligence and Informatics, pp. 1–6 (2021)

32. de Olivera, D.S., dos Santos Garcia, L.T., Gonçalves, L.M.G.: Políticas de formação continuada de professores: inovação para uso da robótica como recurso pedagógico. Revista Linhas **20**(43), 102–131 (2019). https://doi.org/10.5965/1984723820432019102

33. PAPERT, S.: Constructionism A New Opportunity for Elementary Science Education. A proposal to the National Science Foundation Massachusetts Institute of Technology (1986)

34. Potkonjak, V., Vukobratović, M., Jovanović, K., Medenica, M.: Virtual Mechatronic / Robotic laboratory – A step further in distance learning. Comput. Educ. **55**(2), 465–475 (2010). https://doi.org/10.1016/j.compedu.2010.02.010

35. Potkonjak, V., Gardner, M., Callaghan, V., Mattila, P., Guetl, C., Petrović, V.M., Jovanović, K.: Virtual laboratories for education in science, technology, and engineering: A review, vol. 95, pp. 309–327 (2016). https://doi.org/10.1016/j.compedu.2016.02.002

36. Petrović, V.M., Nikolić, B., Jovanović, K., Potkonjak, V.: Development of virtual laboratory for mechatronic systems. In: Advances in robot design and intelligent control (Proceedings of the 25th Conference on Robotics in Alpe-Adria-Danube Region - RAAD 2016), pp. 622–630. Springer (2016)

37. Hashemipour, M., Manesh, H.F., Bal, M.: A modular virtual reality system for engineering laboratory education. Comput. Appl. Eng. Educ. **19**, 305–314 (2011). https://doi.org/10.1002/cae.20312

38. Jaramillo-Botero, A., Matta-Gomez, A., Correa-caicedo, J.F., Perea-Castro, W.: ROBOMOSP. IEEE Robot. Autom. Mag. **13**(4), 62–73 (2006). https://doi.org/10.1109/MRA.2006.250572

39. Feng, L., Borenstein, J.: Correction of systematic odometry errors in mobile robots. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, Pittsburgh, Pennsylvania, USA, p. 3569 (1995). https://doi.org/10.1109/IROS.1995.525942

40. Zilli, S.D.R.: A Robótica Educacional no Ensino Fundamental: Perspectivas e Prática. Master's Dissertation, Florianópolis, Brazil, vol. 1, no. 1, pp. 1–89 (2004)

41. Francinaldo, A., Yanaguibashi, E.A., Silva, S.R., Sa, S.T.L., Gonçalves, L.M.G.: BlockEduc - a tool for visual programming of robots on dynamic environment W-Educ, vol. 1 (2017)

42. Jonassen, D.H.: Computers in the classroom: mindtools for critical thinking, vol. 43, pp. 24–32. Prentice-Hall, Columbus (1996)

43. Pasternak, E., Fenichel, R., Marshall, A.N.: Tips for creating a block language with blockly. In: IEEE Blocks and Beyond Workshop (B&B), Raleigh, NC, vol. 2017, pp. 21–24 (2017). https://doi.org/10.1109/BLOCKS.2017.8120404

44. Seffah, A., Donyaee, M., Kline, R.B., et al.: Usability measurement and metrics: A consolidated model. Softw. Qual. J. **14**, 159–178 (2006). https://doi.org/10.1007/s11219-006-7600-8

45. 7010-2020 - IEEE Recommended Practice for Assessing the Impact of Autonomous and Intelligent Systems on Human Well-Being. Available from https://doi.org/10.1109/IEEESTD.2020.9084219 accesses on January, 30th, 2021

46. LGPD, General Law for Personal Data Protection (web site in Brazilian Portuguese). Available at http://www.planalto.gov.br/ccivil_03/_ato2015-2018/2018/lei/L13709.htm. Accessed on January, 3rd 2021

47. English Translation of the General Law for Personal Data Protection (LGPD, in Portuguese). Available from https://www.lgpdbrasil.com.br/wp-content/uploads/2019/06/LGPD-english-version.pdf. Accessed on January, 3rd 2021

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Lucas Moura do Nascimento has a technician degree in Computers and Internet from the Federal Institute of Education, Science and Technology of Rio Grande do Norte (IFRN, 2018). He is currently pursuing the Bachelor Degree in Information Technology at the Federal University of Rio Grande do Norte. His main interests are in Games and Web Development, Educational Robotics, Robotic Perception, and Simulation Tools, including developments with PHP, SQL, Ruby on Rails, Javascript, C# and Unity.

Davi Souto Neri is currently an undergraduate student pursuing the degree of Technology in Systems Analysis and Development at the Federal Institute of Education, Science and Technology of Rio Grande do Norte (IFRN). His main interests are in Web Development, Digital Games, and Educational Robotics. He has experience with PHP, SQL, Ruby on Rails, Javascript, C# and Unity.

Thiago do Nascimento Ferreira is currently an undergraduate student in Systems Analysis and Development at the Federal Institute of Education, Science and Technology of Rio Grande do Norte (IFRN). His main interests are in Programming Languages, Digital Games, and Web Services. He has experience in rogramming Languages.

Francinaldo de Almeida Pereira holds a a Bachelor Degree in Science and Technology from Federal University of Rio Grande do Norte (2017) and a Bachelor Degree in Computer Engineering also from Federal University of Rio Grande do Norte (2019). He is currently an M.Sc. student at the Graduate Program in Electrical and Computer Engineering from Federal University of Rio Grande do Norte. His main interests are in Web Development and Educational Robotics.

Erika Akemi Yanaguibashi Albuquerque has two Bachelor Degrees in Science and Technology from the Federal University of Rio Grande do Norte (2017) and in Biomedical Engineering also from the Federal University of Rio Grande do Norte (2019). She is currently an M.Sc. student at the Graduate Program in Electrical and Computer Engineering of UFRN (PPgEEC-UFRN) in the area of Robotics in Education. She is also a researcher at the Natalnet Associate Labs from UFRN and has experience in the area of Educational Robotics and Medical Image Processing.

Luiz Marcos Garcia Gonçalves holds a Doctorate in Systems and Computer Engineering (1999) from COPPE-UFRJ, Brazil, including a two years PhD stage at the Laboratory for Perceptual Robotics of UMASS, Amherst, MA, USA. He is Full Professor at the Computer Engineering Department of UFRN, Brazil. He has done researches in the several aspects of Graphics Processing including fields as Robotics Vision (main interest), Computer Graphics, GIS, Geometric Modelling, Animation, Image Processing, Computer Vision, and also has several works on Robotics in Education.

Sarah Thomaz de Lima Sá has a Bachelor Degree in Computer Engineering from the Federal University of Rio Grande do Norte (UFRN, 2011), and a M.Sc. (2013) and a Ph.D. (2016) in Electrical and Computer Engineering also from UFRN. She is Associate Professor at the Federal Institute of Education, Science and Technology of Rio Grande do Norte. She has been the national co-chair for Brazilian Robotics Olympics and its state representative in Rio Grande do Norte, and also a member of the Special Committee in Robotics of Brazilian Computer Society. Her main interests are in Educational Robotics, Systems Development, Embedded Systems and Information Technologies.

Affiliations

Lucas Moura do Nascimento[1] · Davi Souto Neri[2] · Thiago do Nascimento Ferreira[2] ·
Francinaldo de Almeida Pereira[3] · Erika Akemi Yanaguibashi Albuquerque[3] · Luiz Marcos Garcia Gonçalves[3] ⓘ **·**
Sarah Thomaz de Lima Sá[2]

Davi Souto Neri
davisoutoneri@gmail.com

Thiago do Nascimento Ferreira
thiagonascim@yahoo.com

Francinaldo de Almeida Pereira
falmeida.dev@gmail.com

Erika Akemi Yanaguibashi Albuquerque
erikayanaguibashi@gmail.com

Luiz Marcos Garcia Gonçalves
lmarcos@dca.ufrn.br

Sarah Thomaz de Lima Sá
sarah.sa@ifrn.edu.br

[1]   Metrópole Digital Institute, Universidade Federal do Rio
      Grande do Norte, Natal, RN, Brazil

[2]   Diretoria Acadêmica de Gestão e Tecnologia da Informação,
      Instituto Federal de Educação Tecnolócgica do Rio Grande do
      Norte, Natal, RN, Brazil

[3]   Graduate Program in Computer and Electrical Engineering,
      Universidade Federal do Rio Grande do Norte, Natal, RN, Brazil