sBotics: Simulation applied for the practical component of the Brazilian Robotics Olympiad

1st Lucas M. do Nascimento
Instituto Metrópole Digital
UFRN
Natal, Brazil
lucas.moura@ufrn.edu.br

2nd Davi S. Neri
Diretoria Acadêmica de Gestão e Tecnologia da Informação
IFRN
Natal, Brazil
davisoutoneri@gmail.com

3rd Thiago do Nascimento
Diretoria Acadêmica de Gestão e Tecnologia da Informação
IFRN
Natal, Brazil
thiagonascim@yahoo.com

4th Erika A. Yanaguibashi
Departamento de Engenharia de Computação e Automação
UFRN
Natal, Brazil
erikayanaguibashi@gmail.com

5th Sarah T. L. Sá
Diretoria Acadêmica de Gestão e Tecnologia da Informação
IFRN
Natal, Brazil
sarinhaxp@gmail.com

6th Luiz Gonçalves
Departamento de Engenharia de Computação e Automação
UFRN
Natal, Brazil
lmarcos@dca.ufrn.br

***Abstract*—The idea that will be presented in this paper endeavours the use of a web robotics simulation platform throughout Brazil, particularly the simulation of the first two categories of its national robotics competition, the Brazilian Robotics Olympiad (OBR). On this tournament, contestants use robotic kits (such as LEGOs EV3 and Arduino-based kits) to simulate the rescue of a victim in a disastrous situation. The idea of this project is to offer an alternative for those who seek to test their knowledge through other means not needing physical resources, or does not have access to robotic kits normally used by participants in robotics tournaments, a problem predominant in Brazils diverse and unequal society.**

***Index Terms*—Democratisation of Robotics, Simulation, Educational Robotics**

I. Introduction

In the past 40 years, humanity witnessed an unprecedented shift, becoming more connected than ever in ways few people could ever predict. However, while in many places this shift happened to most people, it is not the case in developing nations, such as Brazil, particularly in rural and poor communities where public schools lack funding and most students do not have access to fast and reliable internet.

Despite the fact that Brazil has a national robotics competition, the Brazilian Robotics Olympiad (Olimpíada Brasileira de Robótica, or OBR), less than half of Brazil's municipalities have participated in the tournament, leaving several Brazilians, especially those in poor communities out of the OBR [5].

In regards to this problem, our team developed a web robotics simulation platform, which not only is a cheap solution for the many underfunded public Brazilian schools but a powerful medium to bring more young creative minds into the field of robotics.

Usually, the adopted model for robot-related activities and development consists of a joint group of developers and builders, which are responsible for the planning and elaboration of sets of tests related to the assembly and programming of the developed machine, observing and analysing the results of each one. The ideal setting would be the conduction of these analyses in a perfect robot, which can be complicated when more than one is necessary for its execution [1].

The usage of physical robots imposes many restrictions on the developers, such as the potential damage to the device's components, the limited nature of a real-world environment, the distance between group members and the machine cost itself.

To diminish cost and research time in robot development, multiple pieces of research design proprietary tools to promote an accessible and uncomplicated ways of testing ideas, theories and programs with robots without depending physically on a machine (with the sole exception of the emulation device itself). The simplest way of achieving this is with the usage of computational simulators [3] [2]. Simulators are virtual environments that emulate real-life phenomena, in which users can manipulate, explore and experiment [4].

In some experiments with physical robots, the tests become very extensive, making its repetition necessary. In other cases, constant modifications of the machine are needed, such as different types of sensors and chassis. This continual adaptation of the robot would demand excessive time and batteries for

978-1-7281-4268-5/19/$31.00 ©2019 IEEE
DOI 10.1109/LARS-SBR-WRE48964.2019.00092

the device.

A robotic simulation is not only able to model scenery that resembles real-world situations but emulate real robot movement and behaviour, providing its users with the freedom to interact and perceive the machine in real-time on a virtual setting while performing similar activities that are possible with a real device.

With simulation, the traditional aforementioned development group could focus more on the virtual environment, as every action can be easily adapted to the real world, resulting in less testing with physical components and more efficient time and resource consumption.

There are potentially thousands of citizens that can be reached by OBR [5], and with the lower cost of simulation, the democratisation of educational robotics may be more achievable than ever.

This technology has been implemented in robotic competitions before, such as the RoboCup [9], which has four simulated categories where participants compete in a virtual setting:

- RoboCup Soccer 2D Simulation League;
- RoboCup Soccer 3D Simulation League;
- Rescue Simulation League;
- RoboCup Junior - CoSpace Demo Challenge.

II. Related works

Several works can be found in the topic of simulators for Robotics. The ones that are most closer to our proposal are the Open Roberta and Gazebo. Open Roberta Lab is a programming environment that enables children, adolescents, or any other age group to program different types of robot or micro-controller systems (e.g. Lego Mindstorms, Arduino, and many others) [12].

Gazebo is an open source simulator framework developed by the University of Southern California as a high-fidelity platform for simulating outdoor environments under various conditions. Despite its original purpose, most of its users simulate indoor settings [13].

Table I presents a comparison between Open Roberta, Gazebo and the simulator proposed in this paper. In comparison to the others, the simulator introduced in this paper is the one that best fits the needs of the OBR, offering multi-language programming, being very intuitive and providing user assistance inside the application.

III. Platform Design and Implementation

For the creation of a simulation platform suited for the Brazilian scenario, some elements are required to appease students. The first of which is the digital convenience of the platform, meaning that the average Brazilian could have access to it by using their preferred device. Nonetheless, smartphones are impractical for this type of simulation, due to their lack of processing power, leaving computers as the favoured option. Since many stores in Brazil often sell more affordable machines with pre-installed Linux and are not capable of handling significant amounts of data, a web solution seemed more appropriate. The second requirement of the platform is the accessibility of the programming language used. Since only around 5% of Brazilians have some deep knowledge of the English language [6], many options that in this circumstance would be deemed as "easy to learn" in other nations (such as python) are too challenging for the average non-English proficient Brazilian. For that reason, a partnership with the W-Educ platform was established, providing easy-to-learn Portuguese scripting that could be easily convertible to other languages [7].

TABLE I
Similar platforms and frameworks

Table Head	**Similar simulators and frameworks**		
	O.B.R. Sim	***Open Roberta***	***Gazebo***
Intuitive usage and programming language	yes	yes	no
Works on lower end Computers	yes	yes	yes
Offers help to the user	yes	yes	yes
Can simulate the RoboCup Junior Rescue	yes	yes	yes
Offer support for multiple types of sensors	yes	yes	yes
Multilingual programming	yes	yes	no
3D	yes	no	yes

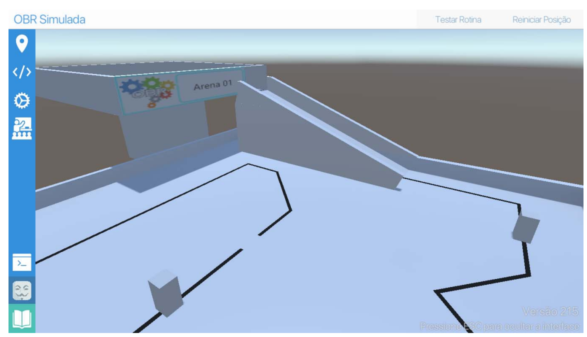

Fig. 1. Starting screen of the platform.

The third and last requirement for the simulator is to have an intuitive user interface, capable of commanding multiple functions in an uncomplicated manner. A well-thought interface and simulation can captivate young minds to experiment with the program while showing to their parents and the elderly the importance of the tasks it is trying to emulate. For this reason, the preset robots designed for the platform have significance to society, such as automated ambulances and police vehicles.

A. Used tools

There are many methods to create a virtual 3D environment, but the approach employed for the development of this tool was the usage of a Game Engine, as it offers many resources and different tutorials for its usage. The engine chosen was Unity3D, one of the most used in third-dimensional development [10].

As for the communication with the W-Educ platform, an API (application programming interface) was used [11], having the side-effect of requiring an internet connection for the programming. However, considering that requests to the API only occur in relevant instances, this connection not necessarily has to be reliable or quick, meaning it can still be accessed in remote communities.

Lastly, for the user programming, it was concluded that a compiler would have to inject the written code while the application was running, a task more challenging as it was primarily considered to be. Then, with the methods and the tools set, the development of the program could commence.

IV. The sBotics simulator

The simulator can be subdivided in three different programs, that are constantly communicating with each other throughout its execution. The first of which are the robots, arenas and the many other scripts developed by our team, which are handled by the client. The arenas use a procedural generation algorithm (which will be further described in the ”Arena generation” subsection), that is able to generate all possible Robocup Rescue A arenas, while the robots are preset by the platform. The second program is the C# Compiler, a compiler that runs alongside the application to compile and inject code into the robot (this will be further explained in the ”Step by step usage of the platform” subsection). And lastly the W-Educ platform, which runs on the server, and is able to convert R-Educ code into C#.

A. User interface

When the platform is started the first thing that the user is presented with is an arena with the first preset robot (see Fig. 8 for reference). By the left of the screen is a control panel, where from the top to bottom the user is able to:

- Log-in in the W-Educ platform to be able to use the coding panel;
- Open a settings panel where the user can change the robot, generate new arenas in a predetermined difficulty, visualise how heavy the desired robot is, select the actuator of the robot, determine the number of victims in the arena, import and export arenas the circuits of the arena, and many other minor things;
- Go to a coding panel which the user can create, open, save and delete programs in R-Educ to be compiled and injected into the robot later (this panel will be further explained in the ”Step by step usage of the platform” subsection);
- Manually move the robot to a position in the arena (this was made so the user does not have to write code to move manually the robot to that certain position, making easier to test how their code interacts with certain parts of that specific arena);
- Open a virtual console screen where the user can view debug code written in their program, such as what each sensor’s value is, or its rotation;
- Open a panel where an arena can be manually generated by the user, with custom obstacles and images. The icon for this panel is a arena since it’s its desired demographic. With this panel, it is possible to generate custom maps such as the map of Brazil with lines for the robot to follow.
- View the R-Educ language tutorial and all the robot’s functions.
- View the simulation timer, for the user to see how long their robot can perform the scheduled tasks.
- View the points made by the user in the simulation.

At the top-bar the user is able to reset the arena and the robot’s position (the ”Reiniciar Posição” button in Fig. 8), as well as start the simulation (the ”Testar Rotina” button), which injects the converted R-Educ code into the robot.

B. Using sBotics

The first action the user has to take when opening the system is to choose a robot that fits their strategy, as well as determine the difficulty for the arena for it to be generated. After this, the user has to authenticate in the W-Educ platform (through the log-in panel) to be able to use the coding screen. This process of communication is done using the W-Educ API [11], and returns to the platform the hash to be used as authentication in further API requests by the platform.

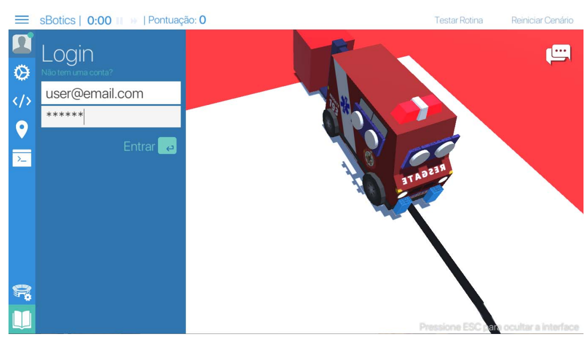

Fig. 2. Login screen of the platform.

Inside the coding panel, the user is presented with many options, such as programming directly in C# or in R-Educ, editing your previously created programs and changing the size of the font. After choosing all of the desired settings, the user can code their desired logic to be applied to the robot, and in the end ”compile” the program using the W-Educ application.

This compiling procedure is not the actual C# compiling that takes place in the code injection, but rather a code ”translation”, where a request is sent to the W-Educ API to convert the R-Educ code to a usable C# code to be later compiled and injected to the robot for it to perform its designated actions.

After choosing the arena, robot, authenticating, writing the code and ”compiling”, the user is finally able to test their own program. After clicking in the button to ’start routine’, the code will be injected into the platform using the aforementioned built-in C# Compiler. Here the user is able to debug any

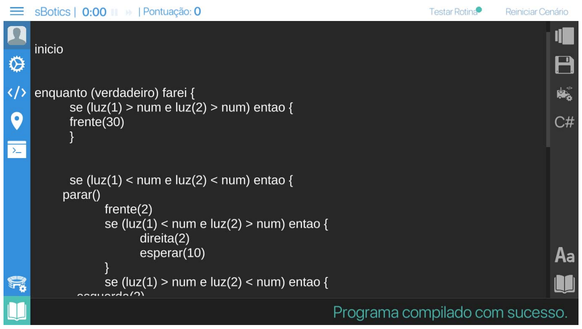

Fig. 3. Programming screen of the platform.

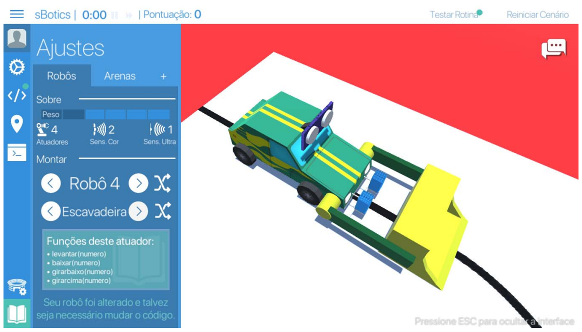

Fig. 4. Robot settings panel of the platform.

problem they might have in the robot just by looking at the Virtual Console screen, as well as resetting the arena and robot's position at any time.

C. Arena Generation

The arena generation algorithm works by generating lines in a 3x6 virtual table comprised of tiles, each tile representing one possible real-life Rescue A arena obstacle. After the arena's generation the user is given a code which contain its information, as well as other manually configurable settings, such as: The type of ramp used, arena checkpoints for the robots, whether or not the user has to follow the lines or use their own defined checkpoints (separate from those generated or configured by the arena). This code can be exported or imported for many educational uses, such as maths teachers creating arenas for their students to create shapes in the simulated environment or other subjects such as geography, where custom arenas can be built simulating real-world maps encouraging exploration and easy learning.

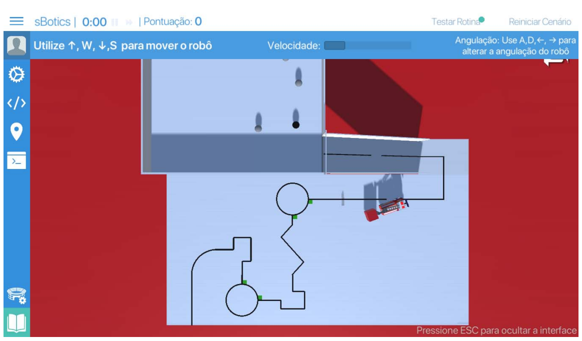

Fig. 5. Change robot's position panel of the platform.

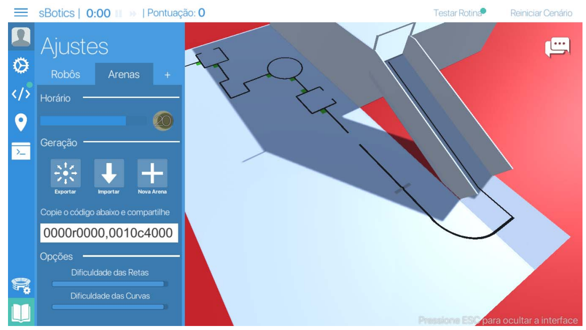

Fig. 6. Arena settings panel of the platform.

For the Rescue A (non-custom) arenas teachers are able to view how well their students are performing, to see how much their techniques have to improve and to stimulate competition among students (that many teachers already use nowadays with platforms such as "Kahoot!"). Through the aforementioned "panel where an arena can be manually generated by the user", it is possible to manipulate such settings without having to manually edit the arena's code, since the editor is simple to use and learn, and it is designed for less tech-savvy people, such as faculty staff or parents.

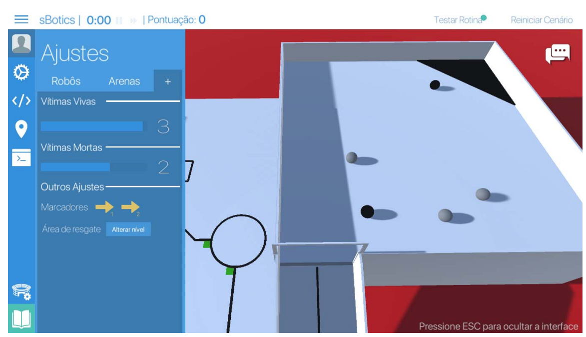

Fig. 7. Arena components settings panel of the platform.

V. Results and Experiments

The first version of the application was completed after several months of development. However, it still has not reached the level of perfection planned for its official first release, though it has provided sufficient material to present itself as a powerful tool in robotic conventions in Brazil, such as the M.N.R. (Brazilian National Robotics Exhibition) and the OBR itself.

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 (see

Fig. 8. Arena's editor screen of the platform.

Fig. 8). 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 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.

A. *Different applications*

This platform has many different possible applications, such as the training of judges in the OBR and courses related to the subjects of robotics or programming logic. In the latest national round of the OBR, the simulator was used to train the best scoring students of the practical component of the tournament (see Fig. 9).

Fig. 9. Simulator being used to instruct the best scoring students of the practical component of the OBR.

VI. Conclusion

In this paper a simulation tool is presented to fulfil the existing gap between students that have no access to robots and the competitions in the Brazilian Robotics Olympiad. To do that, the Unity3D game development engine is used in combination with the W-Educ environment for programming through the Internet and viewing the results immediately in a 3D environment shown in the screen.

After more than one year of development, the simulator is very polished and ready to be used by schools and individuals, to help diffuse educational robotics in an ever-connected society. However, this is not the end of sBotics, since there are still many things our development team has yet to accomplish, such as block-based programming, offline development, user-created robots and a punctuation system to gamify even more the platform and appeal to children.

References

[1] Pedrosa, Eurico F. (2010), Simulated environment for robotic soccer agents, Dissertação de mestrado, Universidade de Aveiro, Portugal
[2] Becker, Guilherme L. (2010), Desenvolvimento de um simulador para um veculo autônomo, Trabalho de Conclusão de Curso, UFMG, Belo Horizonte/MG.
[3] Obst, Oliver & Markus Rollmann (2004), Spark - a generic simulator for physical multi- agent simulations, Lecture Notes in Artificial Intelligence 3187, 243257.
[4] Jonassen, David (1996), O uso de novas tecnologias na educação a distância e a aprendizagem construtivista, Em Aberto, Brasília, ano 16, n.70, abr/jun.
[5] Aroca, Rafael & Pazelli, Tatiana & Tonidandel, Flavio & CA Filho, Azamor & Simoes, Alexandre & Colombini, Esther & Burlamaqui, Aquiles & Gonçalves, Luiz. (2016). Brazilian Robotics Olympiad: A successful paradigm for science and technology dissemination. International Journal of Advanced Robotic Systems. 13.
[6] British Council (2014), Learning English in Brazil: Understanding the aims and expectations of the Brazilian emerging middle classes
[7] Sá, S. T. L.; Fernandes, C. C. ; Barros, R. P. ; Paiva, V. ; Gonçalves, Luiz M. G. . Web based Configurable and Multiplatform Development Environment for Educational Robotics. In: 16th International Conference on Advanced Robotics (ICAR), 2013, Montevideo. Proceedings of IEEE ICAR 2013, 2013
[8] Yanaguibashi, Erika & Thomaz, Sarah & Gonçalves, Luiz. (2019). Short Course at Brazilian Robotics Olympiad: Forming Competitors: Methods and Applications for Teaching and Learning. 10.1007/978-3-319-97085-1_5.
[9] Robocup, http://www.robocup.org/. Last accessed 24 Mar 2018
[10] Unity, https://unity.com/. Last accessed 04 Apr 2019
[11] W-Educ API, https://weduc.docs.apiary.io/. Last accessed 04 Apr 2019
[12] Open Roberta, https://lab.open-roberta.org/. Last accessed 10 Apr 2019
[13] Gazebo, http://gazebosim.org/. Last accessed 10 Apr 2019