SEVEN publicações acadêmicas

CHAPTER 35

USING GODOT TO CREATE GAMIFIED SIMULATORS

https://doi.org/10.56238/sevened2024.041-035

Daniel de Andrade Moura[1], Sabrina Souza da Silva[2], Lucas Martins[3], Marina Mendes Rezende[4], Guilherme Ferreira da Silva[5], Luciano da Silva Carvalho[6], Carlos Antônio de Souza Galdino Junior[7], José Vinicius Gonçalves[8], Maria Nicole Nascimento Silva[9] and Gabriela Yuri Yoshimoto Matsuo[10].

ABSTRACT

Digital simulators have been studied as auxiliary tools in education, especially in physics classes, with the aim of engaging students' interest and facilitating the understanding of the content. In this sense, this article investigates the use of the *Godot Engine,* a tool used in the scope of the Visual Physics project of the Federal Institute of Education, Science and Technology of São Paulo (São Paulo campus), for the development of gamified physics simulators; a process that proved to be challenging, since the team needed to be trained in the use of the *software*. It is intended, therefore, to analyze the problems involving the adaptation of classical physics formulas for use within the engine, in addition to presenting the solutions found so that the simulator could meet the conditions of gamification and simulation of reality.

Keywords: Simulators. Education. Physics. Godot.

[1] Professor of Physics
Federal Institute of São Paulo, São Paulo, SP, Brazil
dmoura@ifsp.edu.br
[2] Physics Student
Federal Institute of São Paulo, São Paulo, SP, Brazil
silva.sabrina@aluno.ifsp.edu.br
[3] Undergraduate in Physics
Federal Institute of São Paulo, São Paulo, SP, Brazil
18martins.lucas@gmail.com
[4] Undergraduate student in Control and Automation Engineering
Federal Institute of São Paulo, São Paulo, SP, Brazil
marina.rezende@aluno.ifsp.edu.br
[5] Undergraduate in Chemistry
Federal Institute of São Paulo, São Paulo, SP, Brazil
guilherme.ferreira3@aluno.ifsp.edu.br
[6] Graduating in Electronic Engineering
Federal Institute of São Paulo, São Paulo, SP
luciano.carvalho@aluno.ifsp.edu.br
[7] Undergraduate in Physics
Federal Institute of São Paulo, São Paulo, SP, Brazil
carlos.galdino@aluno.ifsp.edu.br
[8] Undergraduate in Physics
Federal Institute of São Paulo, São Paulo, SP, Brazil
jose.vinicius@aluno.ifsp.edu.br
[9] Bachelor's Degree in Civil Engineering
Federal Institute of São Paulo, São Paulo, SP, Brazil
maria.nicole@aluno.ifsp.edu.br
[10] Bachelor's Degree in Mechanical Engineering
Federal Institute of São Paulo, São Paulo, SP, Brazil
gabimatsuo22@gmail.com

INTRODUCTION

Simulators can be defined as artificial models that allow the partial or total reproduction of tasks, and can be used in various areas, especially in the educational environment, as exemplified by their applications in medicine and aviation (FILHO and SCARPELIN, 2007). It is in basic education, however, that they have started to be used more frequently, especially during the pandemic period (DA SILVA, et.al., 2021). For Ribeiro (2024), for example, the integration of the well-known Phet platform in high school has had very positive effects compared to lectures, since it has ensured greater participation, improved performance, and also encouraged student learning.

This is because, as pointed out by Lopes, Souza, and Gomes (2023), simulators have the ability to stimulate students' interest in the content, which allows them to explore complex concepts in a more interactive and engaging way. In this way, they have proven to be auxiliary tools in the teaching of sciences such as Physics, which is often seen as difficult by students (MOREIRA, 2021).

For the creation of these simulators or simulations, tools are needed that allow this development. One possibility is Geogebra, a mathematics software that allows, through its resources, the development of simulations involving mathematical concepts (BARBOSA, 2013), as well as computer simulators for the teaching of natural sciences (ARAUJO, BRACHO, 2020)

In this sense, the use of game development engines for the manufacture of gamified computer simulators is explored, since these engines are tools that allow the performance of common tasks related to games, having mechanisms that facilitate these creations, which makes it possible to create simulations of the real world (Paul, Goon, Bhattacharya; 2012). Thus, as it is a game engine, it is evident that the use of the platform enables the creation of simulators with game aspects, such as a scoring system. Thus, users are provided with an educational experience of physical phenomena while ensuring the interactivity inherent to gamification.

Some examples of game engines include: Unity, Unreal, Game Maker and etc., which have well-known simulators, such as Universe Sandbox, Microsoft Flight Simulator, Train Sim World, Assetto Corsa and others. Among the various options available, Godot appears as an open source tool, which can run on various operating systems, such as Windows and Linux, and allowing the creation of games on platforms from mobile devices to computers, in addition to having a basic programming language (GDScript), which is characterized by the clarity of the codes, useful to beginners in the programming area (SILVA, YEPES; 2018). However, despite its characteristics, its productions are very little

widespread and the possible simulators produced in the software practically unknown so far.

In view of a scarce production in the literature in Portuguese on the subject, there is a need for a deepening of the theoretical and practical nature of the use of Godot for the development of digital simulators, aiming to help the teaching of physics. Thus, in order to offer a contribution to the theme, this text brings an analysis of the software from the making of several prototypes.

OBJECTIVES

This article aims to report the practice of developing gamified simulators for teaching physics through Godot Engine 3.5.5, highlighting difficulties and successes to assist the making of other simulators. In addition, it is also intended to expand the available literature on the subject.

METHODOLOGY

Godot Engine was chosen, a priori, by indication of the project's advisor. One of the reasons for this was the lack of infrastructure of the IFSP - SPO with regard to the outdated computers in the school, in such a way that there was a need to opt for software whose hardware demand for processing was lower. Furthermore, because it is an open source application - that is, open source for users - there are no impediments to the licensing of the games or simulators programmed in it.

Thus, at first, the team training stage continued, where some Udemy courses (online school of preparatory courses in technology) were chosen, namely: "How to create 2D platform games with the Godot Engine", whose duration was 36 hours; and "Master 3D Game Development on Godot 4.0 (2023)", with a duration of 13 hours. In this sense, the training allowed to introduce and familiarize the team members with the programming language used by the engine (GDScript) and its interface. In this way, the mechanics introduced in the prototypes developed were based only on the skills acquired in the courses taken - involving, for example, scoring systems, enemy insertion, life point programming for the character and the like - in addition to the experience obtained in the process.

Next, we started to develop prototypes of various themes, totaling 4 initial simulators, which could be made based on already created and open source projects available within the platform itself, addressing topics related to kinematics. After that, it was started from a

demand, within the scope of the Visual Physics project, for the manufacture of 3 gamified simulators that would meet the context of a specific class.

RESULTS AND DISCUSSION

Initially, 4 prototypes were created simultaneously. One of them was the "Block Game" (https://jos3v1n1.itch.io/jogo-do-bloco[11]), in which the user controls the "*player*" character who pushes a block at rest, making it travel a distance according to the player's linear momentum. For this, two *kinematic body nodes were* edited, which allow the creation of mobile bodies via programming; in addition to a scenario created using *tilemaps* - a node intended to store and position objects for the construction of scenarios. The gravity of the character and the block was also configured with a constant called "GRAVITY" and the character's movement was mapped by the "A" and "D" keys. Thus, the collision regions of the two bodies were created with "*CollisionShape*" nodes, allowing interaction between them and the surface, and the surface collision area was defined through the *tilemap*.

Another prototype, which still has no title, was based on a 2D platformer (https://jos3v1n1.itch.io/2d-plataformer) with enemies with a similar programming to the previous one. In it, two rulers were implemented and a clock was created - through a node called "*timer*" - in order to make it possible to calculate the speed of characters and projectiles.

With the two previous projects, it is evident that applying the laws of motion physics in the *game engine* used is a complex task, because many times identical codes can present different behaviors in different projects, which makes programming difficult. This led to the creation of several versions for each simulator, as well as the temporary abandonment of the second prototype. In addition, another major obstacle was the conversion of real units to the programming language, which would be essential for the creation of simulators consistent with reality.

Named "Maquinando Trens" (https://jos3v1n1.itch.io/maquinando-trens), this prototype was based on a train game, in which the user can control the speed of a locomotive with wagons in 6 different phases, from an isometric perspective. It is also possible to verify quantities related to the train, such as its mass and speed, in addition to the frictional force between the rail and the train; and air resistance. In this simulator, the scenery and texture of the train were made and the variables were programmed to be displayed on the screen.

[11] This and the others are on the itch.io platform - a free game upload platform - and do not require a login. It is recommended to use the Google Chrome browser on your computer for access.

Despite being finalized, the prototype presented difficulties in understanding the mechanisms used by the author of the project, especially with regard to the interaction between the tracks and the train, since, instead of a "*kinematic body*" or another body knot, the author used more complex tools to limit the movement of the train on the rail.

Finally, entitled "Cannon Game"(https://jos3v1n1.itch.io/jogo-do-canhao), the last prototype of this half-moment process initially had a cannon and some targets. However, for the same reasons as the prototype "Maquinando trens", a second version of it was made, which did not use an *open source project*. Thus, the new version has a mobile cannon and fixed targets at different heights, and it is up to the user to configure the variables mass of the cannon, mass of the projectile launched, gravity and angle of departure of the projectile. Despite this, both work on the concept of oblique throw, seeking to simulate the quantities involved in it.

Thus, with the last two projects it became evident that editing pre-existing projects is a task that, despite advancing the programming of basic procedures, hinders the development of simulators, given the need to understand the logic of the original author. Therefore, for the following projects, it was decided to make them authorally.

Thus, the second stage - which started from a demand - consisted of the realization of 3 simulators, also namely: The prototype "Bimetallic Blades" (https://jos3v1n1.itch.io/laminas-bimetalicas) was the highest performance simulator, having a drawer that contains 4 different materials that can be chosen by the user's click to compose the blade, which must then be directed to a space in an electrical circuit, where it undergoes dilation. These mechanisms were created through "*Button"* nodes, which when clicked emit signals and display the material referring to the button as part of the blade. In this sense, victory is achieved when the blade with the greatest deformation is chosen.

During the process of making this simulator, it was noticed that the mouse mapping tool used to drag the blade presented inconsistencies in tracking the accuracy of the movement, an error that is even more visible in other screen proportions. Due to this, there is a need, in future versions, to abandon this mechanic and use a more simplified one.

Now with "Linear Expansion on Rails" (https://jos3v1n1.itch.io/trem-dilatacao), the relationships between temperature, time, coefficient of rail expansion, length of the rail segment, length of the path and the spacing between rails - a method formerly used to avoid expansion - were programmed in this simulator. While the last 4 variables must be determined by the user, the temperature varies according to values compiled from 2014 to 2024 (data referring to Rio de Janeiro), through "arrays" and "dictionaries", which allow the storage of several values and their access in a programmed manner.

Named Electron Game (https://jos3v1n1.itch.io/jogo-do-eletron), this focused on the exploration of electric current based on the Drude Model. To this end, we sought to make atoms referring to the wire material and free electrons, in such a way that an interaction between them should occur. However, there was difficulty in programming these collisions, since the electron, according to the model, must be repelled in a coherent way at the angle of collision, an obstacle that has not yet been solved.

As an alternative, a simulator on calorimetry in the kitchen was proposed, entitled "Physics in the Kitchen" (https://jos3v1n1.itch.io/simulador-estados-da-materia), also focusing on the states of matter. This prototype was created in a similar way to the drawer of the simulator "Bimetallic blades", and allows the user to select kitchen items, such as microwave and stove, to melt the ice, given the power of the appliances - from which the time needed for the fusion to occur is determined.

Thus, despite several problems attributed to Godot, it is understood that the platform has, in fact, great potential, as it allowed the manufacture of simulators entirely in its software, in addition to one of the on-demand prototypes having already been finalized. Furthermore, it is noteworthy that the performance of the *engine* remained constant even with larger projects and with greater hardware demand.

FINAL CONSIDERATIONS/CONCLUSION

The creation of this and other simulators through the Godot Engine involved a series of challenges. At first, one of the criticisms arises around the use of the Engine's own language, GDScript, whose syntax lacks some simpler constructions, as occurs in other languages, such as *delay functions* - for controlling the time of actions - and randomization.

These challenges can be partly attributed to the training stage, since the courses used focused too much on game development, rather than simulators, which require the adaptation of real-world physics to the digital world. As an indication, the commands executed by the course teachers already presented errors when they did not behave in the same way when they were transcribed, which delayed the progress of the courses.

Thus, it was noted that for the group to develop the simulators, it was necessary to learn constantly and arduously, based on trial and error, showing that the chosen courses were not enough for the objective. Therefore, it follows that the most appropriate would be courses focused on learning programming in GDScript and, if possible, in physics simulators. However, despite the difficulties, it was possible to create prototypes of considerable performance. Therefore, it is believed that with greater experience with the

platform, *Godot* can be an interesting game engine for the development of outstanding simulators for the educational scenario.

REFERENCES

1. Barbosa, S. M. (n.d.). O software Geogebra e as possibilidades do trabalho com animação. Revista do Instituto GeoGebra Internacional de São Paulo. Disponível em https://revistas.pucsp.br/IGISP/article/view/12843/12201 (Acesso em 29 de julho de 2024).

2. Da Silva, A. S. G., Deosti, L., De Carvalho, H. A. P., & Carvalho, H. A. P. de. (2021). Educação em tempos de pandemia: Uma experiência no ensino de física. UFPR. Disponível em https://compartilha.ufpr.br/wp-content/uploads/2021/11/9-educacao-em-tempos-de-pandemia-1.pdf (Acesso em 20 de julho de 2024).

3. Da Silva, V. P., & Yepes, I. (n.d.). Desenvolvimento de jogos na plataforma Godot. Disponível em http://anais.eati.info:8080/index.php/2019/article/view/282/294 (Acesso em 29 de julho de 2024).

4. Gutiérrez-Araujo, R. E., & Castillo-Bracho, L. A. (n.d.). Simuladores com o software GeoGebra como objetos de aprendizagem para o ensino da física. Disponível em http://www.scielo.org.co/scielo.php?pid=S0121-38142020000100201&script=sci_arttext (Acesso em 29 de julho de 2024).

5. Lopes, J. S., Silva, A. G. da S., & De Souza, G. F. de S. (n.d.). Ensino de Física com uso de simuladores virtuais: Potencial de utilização em sala de aula. Disponível em https://www2.ifrn.edu.br/ojs/index.php/HOLOS/article/view/14365 (Acesso em 26 de julho de 2024).

6. Moreira, M. A. (2021, junho 30). Desafios no ensino da física. Revista Brasileira de Ensino de Física. Disponível em https://www.scielo.br/j/rbef/a/xpwKp5WfMJsfCRNFCxFhqLy/ (Acesso em 29 de julho de 2024).

7. Paul, P. S., Goon, S., & Bhattacharya, A. (n.d.). History and comparative study of modern game engines. Disponível em https://encurtador.com.br/NMNNp (Acesso em 29 de julho de 2024).

8. Pazin Filho, A., & Scarpelini, S. (2007). Simulação: Definição. Medicina (Ribeirão Preto), 40(2), 162–166. Disponível em http://www.revistas.usp.br/rmrp/article/view/312 (Acesso em 29 de julho de 2024).

9. Ribeiro, J. P. M. (n.d.). A integração do laboratório virtual “PHET Interactive Simulations” no ensino de física. Disponível em https://ciet.ufscar.br/submissao/index.php/ciet/article/view/797 (Acesso em 26 de julho de 2024).