Designing Immersive Learning Experiences: A Postmortem Analysis of a VR Game for Teaching Programming

Gustavo Martins Nunes Avellar[1], **Paulo André Pimenta Aragão**[1], **Fernando Henrique Paes Generich**[1], **Pedro Lunkes Villela**[1], **Lucas Oliveira Castro**[1], **Ellen Francine Barbosa**[1]

[1] Instituto de Ciências Matemáticas e de Computação (ICMC-USP)
Universidade de São Paulo – Departamento de Sistemas de Computação
Caixa Postal 668 - CEP 13560-970 – São Carlos (SP), Brasil

{gustavo.avellar, andre.aragao, fernando_gene, pedrolunkesvillela, lucascastro}@usp.br, francine@icmc.usp.br

***Abstract. Introduction**: Teaching programming concepts through immersive environments has demonstrated pedagogical potential, although most existing solutions rely on expensive technologies that remain unaffordable in many educational contexts. **Objective:** This paper presents a postmortem analysis of the design, development, and evaluation of SSPOT-VR, an affordable, collaborative VR mobile game developed to support the teaching and learning of programming for K-12 students. **Methodology**: We adopted a postmortem analysis approach grounded in the game development literature, supported by user evaluations and continuous reflection throughout iterative development phases. **Results:** The analysis highlights strengths, including multiplayer design, visual storytelling, and accessible deployment, while also identifying challenges related to infrastructure, usability, and scalability. Lessons learned informed design enhancements and future directions, such as cross-platform support and strategies for pedagogical integration.*

***Keywords** Postmortem Analysis, Educational Game Design, Programming Education, Mobile Devices, Virtual Reality.*

1. Introduction

Teaching computer programming has become an essential competency in 21st-century education [Computer Science Teachers Association 2017], increasingly integrated into curricula across different educational levels [Ribeiro et al. 2022]. Learning to program opens academic and professional opportunities and strengthens logical reasoning and problem-solving skills [Luxton-Reilly et al. 2018]. In K-12 contexts, visual programming has emerged as an effective introductory approach, allowing students to create and share interactive applications through a comprehensible and intuitive visual language [Weintrop 2019].

In recent years, the combination of visual programming, specifically block-based programming, with immersive technologies such as Virtual Reality (VR) and Augmented Reality (AR) has been explored to make programming instruction more engaging and interactive [Agbo et al. 2023]. VR, in particular, enables immersive environments that foster active learning and gamified experimentation with computational concepts, facilitating the visualization of abstract ideas while enhancing student engagement and knowledge retention [Stigall e Sharma 2017]. However, most research in this area relies on high-end equipment, such as head-mounted displays (HMDs) connected to powerful computers [Vincur et al. 2017, Segura et al. 2020, Jin et al. 2020].

The use of expensive hardware restricts the adoption of VR solutions in large-scale educational contexts, especially in developing regions with limited infrastructure. Approximately half of the research on VR for supporting the teaching and learning of programming adopts only high-end HMDs and powerful computers [Sukirman et al. 2022]. Beyond investigations on high-end equipment, it is essential to examine how students engage with mobile immersive VR games that rely on more affordable solutions, such as low-cost HMDs and smartphones [Pellas et al. 2021].

To address the limitations imposed by cost in immersive learning environments, previous studies presented the development and evaluation of SSPOT-VR (Space Station for Programming Training in Virtual Reality), a mobile collaborative VR game aimed at supporting the teaching and learning of programming concepts to K-12 students through a visual and interactive digitally created world [Avellar et al. 2024]. The game was designed to operate on low-cost smartphone-powered HMDs, the game integrates immersive VR, storytelling, block-based programming tasks, and real-time collaboration. The primary objective was to deliver a playful and engaging learning experience that remains pedagogically grounded, and scalable to school contexts with limited technological infrastructure.

The development of the game followed an iterative, user-centered design process, incorporating feedback from field deployments involving approximately 170 K-12 students. Aligned with agile design principles, the team engaged in regular reflection cycles to assess and adjust the games's features and overall quality [Murphy-Hill et al. 2014]. As a result, the game has undergone multiple redesign phases that included technical adaptations, interface enhancements, content expansion, and performance improvements. This trajectory produced a broad set of design decisions, insights, and constraints that justify a structured and critical reflection on its evolution.

In this paper, we present a postmortem analysis of the development process, drawing from practices commonly used in software and game development to promote reflective evaluation [Murphy-Hill et al. 2014]. Postmortem analysis is a part of a game's development process and serves to identify mistakes and improve future processes, as well as to recognize team efforts and achievements [Chandler 2014]. It is an important practice throughout longer projects, enabling continuous improvement between development phases rather than exclusively at the end [Chandler 2014]. Inspired by this perspective, we adopted the postmortem analysis to examine the evolution of the educational VR game, emphasizing key design decisions, pedagogical constraints, and lessons learned.

The remainder of this paper is organized as follows: Section 2 presents the background; Section 3 reviews related work; Section 4 describes the methodology; Section 5 provides the postmortem analysis; and Section 6 concludes with final remarks and future directions.

2. Background

Computer Science (CS) and its associated technologies play a central role in contemporary society and the global economy

[Computer Science Teachers Association 2017]. In order to participate fully in a digitally mediated world and to be prepared for 21st-century careers, individuals must develop a solid understanding of fundamental CS concepts and practices [Computer Science Teachers Association 2017].

One promising approach to support CS education is the use of educational games. These environments provide interactive, engaging, and student-centered learning experiences that stimulate critical thinking and effectively facilitate knowledge retention [Zhan et al. 2022]. Considering that games are inherently appealing and designed to sustain attention, they can promote active engagement with learning tasks over extended periods of time. This quality is particularly valuable in educational contexts where maintaining student attention has become increasingly challenging [Carreño-León et al. 2018].

In addition, educational games offer a safe and exploratory space for learning, where students can make mistakes without the negative consequences typically associated with failure in traditional classrooms. This freedom fosters a mindset that encourages experimentation and the development of problem-solving skills, which are critical for success in CS and other 21st-century domains [Denner et al. 2019]. These foundations support the integration of immersive technologies such as VR and AR, opening new possibilities for the design of educational games. When applied to programming education, these technologies provide a unique opportunity to explore complex computational concepts in a hands-on and visually engaging manner [Agbo et al. 2023]. VR and AR can transform theoretical abstractions into immersive experiences, allowing students to visualize and interact with abstract ideas in ways that are often unattainable in conventional learning environments [Rozelma França 2015].

Visual and narrative design are key to creating immersive VR and AR learning experiences. Coherent visual elements, such as color, scale, and spatial layout, enhance the sense of presence, while storytelling structures give context and continuity to actions [Jerald 2016]. In collaborative settings, narrative-driven goals and visual cues also help guide interaction and coordination among learners, reinforcing both cognitive engagement and social connection [Tori e Hounsell 2018].

Particularly in the context of VR, the rise of mobile and low-cost solutions has made immersive experiences more available, especially in educational settings with limited infrastructure [Agbo et al. 2023]. By leveraging affordable smartphones and basic VR HMDs with bluetooth joysticks, mobile VR offers scalable opportunities for game-based learning, eliminating many of the financial and technical barriers associated with high-end devices. These experiences can promote meaningful learning through interaction, experimentation, and shared problem-solving [Sukirman et al. 2022].

As game development increasingly intersects with education, postmortem analysis has emerged as a valuable method for structured reflection. Chandler (2014) frames postmortem analysis as a lightweight and flexible practice embedded throughout development, enabling teams to document what worked, what did not, and why [Chandler 2014]. Widely adopted in the games industry, postmortem analysis supports continuous improvement and knowledge sharing, particularly in iterative and agile workflows [Murphy-Hill et al. 2014]. In this study, we apply postmortem analysis to critically reflect on the evolution of a collaborative mobile VR game designed to teach programming to K-12 students.

3. Related Work

Educational solutions designed to support the teaching and learning of programming can be categorized into five types [Pears et al. 2007]: (i) visualization tools, (ii) automatic assessment tools, (iii) programming support environments, (iv) microworlds, and (v) miscellaneous tools that do not fit into the previous categories. Among these, visualization solutions that adopt visual programming approaches are particularly relevant in educational contexts due to their capacity to reduce the syntactic burden and promote conceptual understanding through intuitive interfaces [Weintrop 2019].

In this context, several educational initiatives have explored the use of immersive VR to support the teaching and learning of programming across different educational levels. A significant portion of these efforts relies on high-end hardware configurations to deliver immersive environments and 3D game-like scenarios.

For example, Cubely [Vincur et al. 2017] and VR-OCKS [Segura et al. 2020] used HTC Vive HMDs and Unity 3D to introduce basic programming concepts through tasks requiring algorithmic thinking and visual logic composition. Similarly, VWorld [Jin et al. 2020], developed for Oculus Quest, focused on promoting creativity and computational thinking by enabling users to create algorithmic behavior for virtual objects using motion and sound commands.

On the other hand, low-cost VR solutions have emerged as alternatives for broader access. Studies such as OOPVR [Tanielu et al. 2019], Imikode [Sunday et al. 2022], and iThinkSmart [Agbo et al. 2022] explored the use of mobile VR combined with basic VR HMDs. These applications were also developed using Unity 3D and focused on supporting the understanding of object-oriented programming concepts, algorithmic thinking, and problem-solving.

Across the reviewed literature, no study was found to adopt a postmortem analysis of the design, development, and evaluation trajectory. This work addresses that gap by providing a structured postmortem analysis of the design and iterative refinement of a collaborative mobile VR game for teaching and learning programming. It aims to contribute to the field of educational VR design and to the documentation of reflective practices that support sustainable and scalable innovation in immersive learning environments.

4. Methodology

4.1. Research Approach

This study adopts a postmortem analysis as its central methodological strategy, which is commonly used in software and game development to assist reflective evaluation. Rather than applying a prescriptive model, we follow the flexible and lightweight approach suggested by Chandler (2014) [Chandler 2014], who considers postmortem analysis to be an integral part of the development process, aiming both to identify mistakes and to recognize team achievements.

Chandler (2014) also recommends conducting postmortems between development phases to accelerate learning and improve responsiveness [Chandler 2014]. Our study is aligned with this view, as the development of the game evolved over multiple cycles, each accompanied by internal documentation and discussions, culminating in the first analysis conducted which is presented in this paper.

4.2. Development Process

The game was initially created during a research project focused on low-cost immersive technologies for educational use. Following the prototype, the project evolved into a cooperative development effort, engaging a multidisciplinary team. Unity 3D [Unity Technologies 2025] and C# was adopted as the primary development technology, targeting Android smartphones in combination with low-cost VR HMDs [Google 2025]. The project followed an iterative development model [Jerald 2016], with each cycle dedicated to refining specific features, such as programming concepts, additional levels, navigation, language support, animation, and multiplayer infrastructure. Activities were organized around short development sprints, regular feedback meetings, and version control practices.

One of the most significant advances was the implementation of multiplayer support, enabling real-time collaboration between users within the VR environment. This transition transformed the game into an online experience, adding a new layer to the development process and expanding its pedagogical value. Additionally, the collaborative gameplay strategy is aligned with contemporary educational practices, reinforcing teamwork, communication, and shared problem-solving [Vincur et al. 2017].

4.3. Evaluation and Redesign

The first version of the SSPOT-VR was evaluated through structured empirical studies involving approximately 170 K-12 students in both remote and classroom settings [Avellar et al. 2024]. The evaluation employed validated models addressing user acceptance, technology adoption, and learning outcomes.

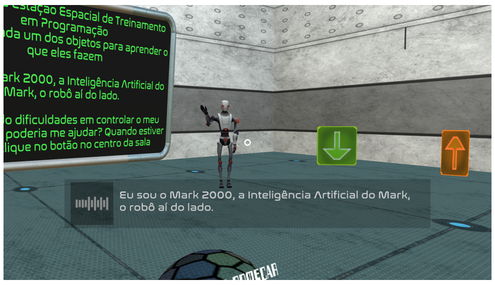
Findings from these studies guided several redesign iterations. Improvements were introduced to enhance visual aesthetics, interaction design, level progression, and accessibility, such as implementing multilingual narration and synchronized subtitles. These changes were implemented through iterative design cycles and guided by structured reflection meetings held after each deployment phase. The redesign efforts directly contribute to the postmortem analysis presented in this study.

5. Lessons Learned from the Design, Development and Evaluation Process

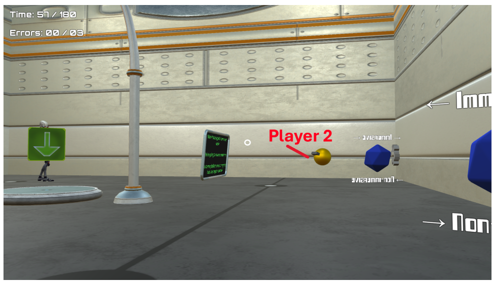
5.1. What Worked Well

One of the most effective design decisions was to adopt low-cost technologies to build an immersive and affordable experience that competes in quality with expensive high-end VR setups. The game operates on standard Android smartphones with gyroscope and accelerometer sensors, and it supports both VR HMDs (such as Google Cardboard or VR Box) and a non-immersive fullscreen mode ensured flexibility and broader adoption. A desktop version for Windows was also developed, enabling cross-platform and cross-device interaction regardless of VR equipment, and allowing both online and offline gameplay.

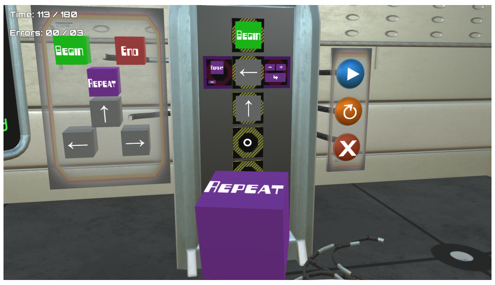
The integration of a visual programming language based on interactive cubes (i.e., block-based programming) proved effective in promoting engagement and logical reasoning. Furthermore, the game's narrative structure, immersive 3D environments, and responsive interaction design contributed to student immersion and motivation. User studies revealed high levels of acceptance, enjoyment, and knowledge retention among K-12 students, who also expressed interest in using the game more regularly in classroom settings [Avellar et al. 2024]. These characteristics positioned SSPOT-VR as a strong alternative to more expensive immersive learning tools, particularly in the context of K-12 education in developing regions. Figure 1 provides a overview of the visual and functional improvements made throughout the game's development.


Eu sou o Mark 2000, a Inteligência Artificial do Mark, o robô ao lado.

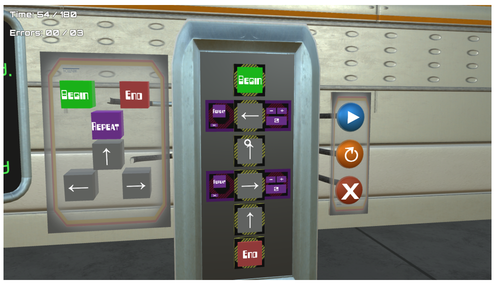
(a) Example of narration and synchronized subtitles, in the tutorial and subsequent levels.



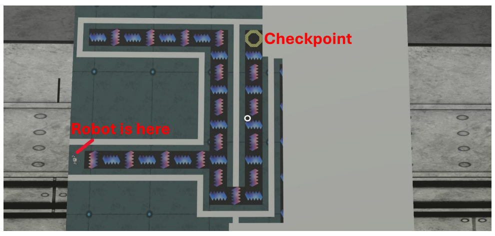
(b) Collaboration in the programming task, where the golden sphere represents player two.



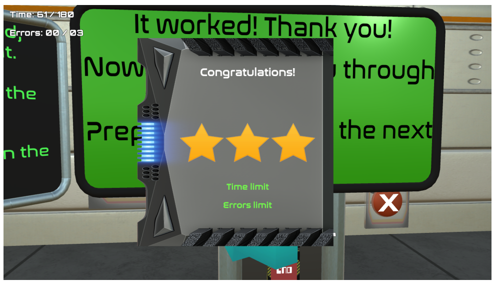
(c) Level designed to introduce and practice the loop concept.



(d) Alternate perspective of the level focused on the loop concept.



(e) Maze level where students work together to program the robot to traverse it.



(f) Gamification elements including score, timer, and fanfare.

Figure 1. New features and improvements in the latest game version.

Among the most impactful additions was the implementation of a multiplayer mode enabling real-time collaboration throughout the entire game, the introduction of new programming concepts (loops), and the maze level. The game's collaborative mode was specifically designed to simulate pair programming dynamics. In this mode, two students interact in real time, assuming complementary roles to collaboratively complete tasks across all levels. For instance, in the maze level, one student observes the environment and communicates the necessary instructions, while the other programs the character's movements using the visual language. The roles are subsequently reversed, ensuring that both participants experience both perspectives.

These features enhanced the pedagogical value of the game through teamwork. Additional enhancements such as synchronized subtitles, multilingual narration (Portuguese and English), and visual refinements also contributed to user experience. Notably, no motion sickness was reported during or after the game sessions, indicating a well-calibrated and comfortable VR design.

The game also integrates classical game design elements [Zhan et al. 2022]. It features a storytelling strategy that provides clear goals and contextualizes each task. Mechanics such as time challenges, visual feedback (e.g., scores, progress indicators), and auditory rewards (e.g., fanfares upon level completion) were incrementally introduced to reinforce student motivation and a sense of accomplishment. Collectively, these elements contribute to sustaining attention, fostering meaningful play, and creating a rewarding learning experience [Chandler 2014]. Table 1 summarizes the progression from the initial to the current version of the game.

Table 1. Evolution of the Game's Features and Educational Scope

Aspect	**Initial Version**	**Current Version**
Number of Levels	Two levels: (i) tutorial and (ii) algorithm composition.	Five levels: (i) tutorial; (ii) algorithm; (iii) loops (one command); (iv) loops (multi-command); (v) maze.
Programming Concepts	Algorithm composition and usage of visual programming language.	All the previous and loops.
Interaction Mode	Single-player only.	Single-player and online multiplayer with real-time collaboration in pairs.
Platform Support	Android (local play only).	Android and Windows, both local and online, with cross-platform support.
VR Mode	VR Box, Cardboard or full-screen mode.	VR Box, Cardboard or full-screen mode.
Accessibility Features	None	Multilingual narration (PT/EN), subtitles, and language toggle

5.2. What Went Wrong and What We Learned

The development of SSPOT-VR was marked by a variety of challenges across technical, pedagogical, and operational dimensions. One of the core technical constraints was ensuring consistent performance on low-end smartphones, many of which had limited processing power or lacked essential sensors such as gyroscope and accelerometer. These hardware limitations affected the immersive experience and, in some cases, prevented the game from being used altogether. Additionally, despite being compatible with Android and Windows, the game remains unavailable on iOS, macOS, and Linux, which limits scalability in schools with diverse or restricted infrastructures.

Another significant challenge was the complexity of implementing real-time multiplayer functionality in a mobile VR environment. Connection stability, latency, and synchronization issues were frequent, particularly in school settings with deficient internet infrastructure. These factors occasionally disrupted gameplay and created inconsistent collaborative experiences between students.

From a usability standpoint, early iterations of the game exhibited issues related to visual clarity and instructional support. Students identified problems with font legibility, color contrast, feedback on interactive objects, and visual guidance. These shortcomings made the process harder for new users and reduced the pedagogical effectiveness of some interactions. On the other hand, while the game was designed for K-12 students, educators reported the need for specific training and support materials to integrate the game effectively into their classrooms. The novelty of VR technology, combined with the need to teach programming, introduced a double layer of complexity for educators unfamiliar with either domain. Moreover, the game's installation process and VR-specific interaction model posed barriers for both students and educators who were used to more conventional educational tools.

The current absence of official app store distribution (e.g., Google Play, App Store, Microsoft Store) further complicates deployment. Similarly, limitations in platform compatibility, such as the inability to run on older devices, unsupported operating systems, or web browsers, remain unresolved. These issues underscore the need for platform-agnostic deployment strategies, including a potential WebGL version to facilitate broader access.

Despite extensive internal testing, occasional bugs have been reported, especially when the game is used on heterogeneous devices in uncontrolled environments. These technical setbacks reinforce the complexity of building stable, scalable VR software that operates seamlessly across platforms and use cases. The development team's reliance on collaborative tools, such as documentation logs, team retrospectives, and structured feedback sessions, has been instrumental in identifying, tracking, and resolving such incidents over time.

Finally, the project's iterative nature has revealed a broader challenge in balancing pedagogical depth with technical feasibility. The demand for additional phases, programming concepts (e.g., conditionals, object-oriented programming, data structures, and so on), and greater narrative complexity must be met without compromising performance or access. Meeting these goals will require continued development and collaboration with educators, designers, and developers to ensure the game remains affordable, scalable, and impactful.

5.3. Insights from Iterative Development and User Feedback

The iterative development approach adopted throughout the project was essential for identifying design flaws, improving usability and user experience, and expanding educational value. Each version of the game was informed by structured evaluations, user feedback, and reflective meetings among team members. These feedback loops enabled rapid adjustments, particularly in visual design, interface layout, and content progression.

User feedback indicated a high level of student engagement, but also highlighted dissatisfaction with the limited scope and short duration of the initial content, which included only a tutorial and a single practical activity focused on algorithm composition. This insight directly led to the expansion of the game, which now includes five progressive stages: (i) a tutorial; (ii) algorithm composition level; (iii) loop structures with only-one instruction level; (iv) loop structures with multiple instructions level, and (v) a maze level. The latter was designed specifically to foster real-time communication between paired students, reflecting the pedagogical value of collaborative problem-solving in VR environments.

Working with K-12 students and educators in both online and in-classroom settings emphasized the necessity of tailoring design to educational realities, including limited infrastructure, equipment variability, and varying levels of familiarity with programming and VR. Feedback from educators also highlighted the need for training and pedagogical support materials, leading to the proposal of a formal instructional guide to facilitate classroom implementation. Another crucial insight derived from practice was the importance of enabling the game to run on multiple devices and configurations. It became evident that support for non-HMD play, desktop modes, and mixed-device multiplayer (e.g., Android and Windows) was essential to achieving broader inclusion. All of these features, initially perceived as auxiliary, emerged as central to the game's educational impact.

Although the technical complexity of VR development, particularly for mobile platforms, presented recurring challenges, the team adopted rigorous testing and feedback review protocols to address bugs, performance bottlenecks, and design inconsistencies before each deployment. The collaborative development model, involving undergraduate and graduate students, enriched the design process by distributing responsibilities across technical, pedagogical, and aesthetic dimensions. This structure supported sustained innovation, including the addition of new levels, mechanics, and accessibility features, while building local capacity in immersive educational technology development.

5.4. Designing Affordable VR Experiences for Learning

The SSPOT-VR project was originally conceived as a response to the widespread unavailability of immersive educational applications that rely on high-cost VR setups and proprietary platforms. In contrast to solutions built on high-end HMDs with powerful tracking and desktop systems, this game was designed from the outset to operate on affordable Android smartphones, using only gyroscope and accelerometer sensors. It can be played with low-cost VR HMDs (such as Google Cardboard or VR Box) or even without a HMDs in fullscreen mobile mode, preserving interactivity through natural sensor orientation.

The project was guided by the goal of delivering a game that, despite operating in a less advanced tracking environment, could offer engagement, visual quality, and pedagogical value comparable to high-end immersive applications. This commitment influenced every aspect of the experience, from the implementation of a visual programming language using interactive cubes to the storytelling strategy and the design of 3D, sci-fi-themed environments.

Cross-device multiplayer (Android and Windows) was implemented to ensure flexibility in diverse educational scenarios. Looking ahead, the development roadmap includes support for iOS, Linux, and Web platforms, allowing the game to run in browser environments. In parallel with the technical infrastructure, multilingual support was added through synchronized narration and subtitle systems in Portuguese and English. This framework is designed to be easily extended to other languages, reinforcing the game's global scalability and educational relevance across different cultural contexts.

The project also embraces the broader goal of inclusion and democratization of digital education. It addresses the real needs of public schools and educational institutions in developing regions. By providing an immersive, collaborative, and story-driven experience that can run on commonly available devices, the game contributes to expanding the reach of programming education and digital literacy.

6. Conclusions and Future Work

This paper presented a postmortem analysis of the design, development, and evaluation of SSPOT-VR, an immersive, collaborative, and low-cost mobile VR game for programming education. We discussed the project's main strengths, challenges, and lessons learned throughout its trajectory. Contributions include the integration of full multiplayer collaboration, the expansion of programming concepts, enhancements in narrative and visual aesthetics, and the addition of accessibility features such as multilingual support and synchronized subtitles.

The design process was informed by empirical evaluations, leading to continuous refinements across pedagogical, visual, and technical dimensions. The game demonstrated its ability to engage students and support knowledge retention, while maintaining usability and affordability [Avellar et al. 2024]. These findings reinforce the value of postmortem analysis as a means to critically evaluate complex educational game projects and support reflective design practice [Chandler 2014].

In line with the planned content progression, future development will focus on expanding the game's scope by introducing new programming concepts, such as conditional statements, object-oriented programming, and basic data structures [Computer Science Teachers Association 2017]. The collaborative mechanics will be further refined, and support for additional platforms, including iOS, Linux, and Web, which will be prioritized to enhance scalability and access. In the long term, the game is envisioned to evolve into an educational metaverse [Mystakidis 2022, Hwang e Chien 2022], incorporating avatar customization and social interaction features, while maintaining its pedagogical integrity and low-cost setup.

To support classroom use, the team also plans to develop materials and a comprehensive instructional guide for educators, such as the Computational Thinking Teaching Resources [CSTA e ISTE 2011]. These resources aim to simplify implementation, address challenges, and empower educators with little prior experience in programming or VR to confidently integrate the game into their practices.

Acknowledgments

This study was financed by the University of São Paulo (USP) and the Brazilian funding agency *Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil* (CAPES) - Finance Code 001.

References

Agbo, F. J., Oyelere, S. S., Suhonen, J., e Tukiainen, M. (2022). Design, development, and evaluation of a virtual reality game-based application to support computational thinking. *Educational technology research and development.*

Agbo, F. J., Oyelere, S. S., Suhonen, J., e Tukiainen, M. (2023). Design, development, and evaluation of a virtual reality game-based application to support computational thinking. *Educational technology research and development*, 71(2):505–537.

Avellar, G. M. N., Fioravanti, M. L., Deus, W. S., Branco, K. R. J. C., e Barbosa, E. F. (2024). SSPOT-VR: An immersive and affordable mobile application for supporting K-12 students in learning programming concepts. *Education and Information Technologies*, 29(13):16411–16439.

Carreño-León, M., Sandoval-Bringas, A., Álvarez-Rodríguez, F., e Camacho-González, Y. (2018). Gamification technique for teaching programming. In *2018 IEEE Global Engineering Education Conference (EDUCON)*, pages 2009–2014.

Chandler, H. (2014). *The Game Production Handbook.* Foundations of game development. Jones & Bartlett Learning.

Computer Science Teachers Association (2017). *K-12 Computer Science Standards.* CSTA.

CSTA e ISTE (2011). Computational Thinking Teacher Resources. Computer Science Teachers Association (CSTA). International Society for Technology in Education (ISTE).

Denner, J., Campe, S., e Werner, L. (2019). Does computer game design and programming benefit children? a meta-synthesis of research. *ACM Trans. Comput. Educ.*, 19(3).

Google (2025). Google VR & AR. URL: https://arvr.google.com/cardboard/.

Hwang, G.-J. e Chien, S.-Y. (2022). Definition, roles, and potential research issues of the metaverse in education: An artificial intelligence perspective. *Computers and Education: Artificial Intelligence*, 3:100082.

Jerald, J. (2016). *The VR Book: Human-Centered Design for Virtual Reality.* Association for Computing Machinery and Morgan &#38; Claypool, New York, NY, USA.

Jin, Q., Liu, Y., Yuan, Y., Yarosh, L., e Rosenberg, E. S. (2020). VWorld: An immersive vr system for learning programming. *Proceedings of the 2020 ACM Interaction Design and Children Conference: Extended Abstracts.*

Luxton-Reilly, A., Simon, Albluwi, I., Becker, B. A., Giannakos, M., Kumar, A. N., Ott, L., Paterson, J., Scott, M. J., Sheard, J., e Szabo, C. (2018). Introductory programming: A systematic literature review. *Proceedings Companion of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education.*

Murphy-Hill, E., Zimmermann, T., e Nagappan, N. (2014). Cowboys, ankle sprains, and keepers of quality: how is video game development different from software development? In *Proceedings of the 36th International Conference on Software Engineering*, ICSE 2014, page 1–11, New York, NY, USA. Association for Computing Machinery.

Mystakidis, S. (2022). Metaverse. *Encyclopedia*, 2(1):486–497.

Pears, A., Seidman, S., Malmi, L., Mannila, L., Adams, E., Bennedsen, J., Devlin, M., e Paterson, J. (2007). A survey of literature on the teaching of introductory programming. *SIGCSE Bull.*, 39(4):204–223.

Pellas, N., Mystakidis, S., e Kazanidis, I. (2021). Immersive virtual reality in K-12 and higher education: A systematic review of the last decade scientific literature. *Virtual Reality Journal*.

Ribeiro, L., Cavalheiro, S., Foss, L., Cruz, M., e França, R. (2022). Proposta para implantação do ensino de computação na educação básica no brasil. In *Anais do XXXIII Simpósio Brasileiro de Informática na Educação*, pages 278–288, Porto Alegre, RS, Brasil. SBC.

Rozelma França, P. T. (2015). Desafios e oportunidades ao ensino do pensamento computacional na educação básica no brasil. *Anais dos Workshops do Congresso Brasileiro de Informática na Educação*, 4(1):1464.

Segura, R. J., del Pino, F. J., Ogáyar, C. J., e Rueda, A. J. (2020). Vr-ocks: A virtual reality game for learning the basic concepts of programming. *Computer Applications in Engineering Education*.

Stigall, J. e Sharma, S. (2017). Virtual reality instructional modules for introductory programming courses. *Proceedings of the 7th IEEE Integrated STEM Education Conference*.

Sukirman, S., Ibharim, L. F. M., Said, C. S., e Murtiyasa, B. (2022). A strategy of learning computational thinking through game based in virtual reality: Systematic review and conceptual framework. *Informatics in Education*.

Sunday, K., Wong, S. Y., Samson, B. O., e Sanusi, I. T. (2022). Investigating the effect of imikode virtual reality game in enhancing object oriented programming concepts among university students in Nigeria. *Education and Information Technologies*.

Tanielu, T., 'Akau'ola, R., Varoy, E., e Giacaman, N. (2019). Combining analogies and virtual reality for active and visual object-oriented programming. *Proceedings of the ACM Conference on Global Computing Education*.

Tori, R. e Hounsell, M. d. S. (2018). *Introdução a realidade virtual e aumentada*. Editora SBC, Porto Alegre (RS).

Unity Technologies (2025). Unity 3D. URL: https://unity.com.

Vincur, J., Konopka, M., Tvarozek, J., Hoang, M., e Navrat, P. (2017). Cubely: Virtual reality block-based programming environment. *Proceedings of the 23rd ACM Symposium on Virtual Reality Software and Technology*.

Weintrop, D. (2019). Block-based programming in computer science education. *Communications of the ACM*.

Zhan, Z., He, L., Tong, Y., Liang, X., Guo, S., e Lan, X. (2022). The effectiveness of gamification in programming education: Evidence from a meta-analysis. *Computers and Education: Artificial Intelligence*, 3:100096.